LOAN DOCUMENT

AD-A239 256

DTIC ACCESSION NUMBER

PHOTOGRAPH THIS SHEET

LEVEL

INVENTORY

WL-TR-91-8023

DOCUMENT IDENTIFICATION

Jul 91

DISTRIBUTION STATEMENT

HANDLE WITH CARE

ACCESSION FOR

NTIS	GRA&I	☑
DTIC	TRAC	☐
UNANNOUNCED		☐
JUSTIFICATION		

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

DTIC DOCUMENT PROCESSING

DATE ACCESSIONED

DATE RETURNED

91

DATE RECEIVED IN DTIC

91-07353

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM 70A JUN 90

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED

LOAN DOCUMENT

WL-TR-91-8023

AD-A239 256

Product Definition Data Interface (PDDI)

Product Specification

McDonnell Aircraft Company
McDonnell Douglas Corporation
P. O. Box 516
St. Louis, MO 63166

July 1991

Final Report

Approved for public release; distribution is unlimited.

MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT LABORATORY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY UNCLASSIFIED	3. DISTRIBUTION / AVAILABILITY OF REPORT APPROVED FOR PUBLIC RELEASE DISTRIBUTION UNLIMITED
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) WL-TR-91-8023
6a. NAME OF PERFORMING ORGANIZATION McDonnell Aircraft Company 6b. OFFICE SYMBOL (If applicable) McAir	7a. NAME OF MONITORING ORGANIZATION Manufacturing Technology Dir. (WL/MTIB) Wright Laboratory
6c. ADDRESS (City, State, and ZIP Code) McDonnell Douglas Corporation P. O. Box 516, St. Louis, MO 53166	7b. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB, OH 45433-6533
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Materials Lab 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-82-C-5036
8c. ADDRESS (City, State, and ZIP Code) Wright Patterson Air Force Base, Ohio 45433-6533	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 78011F; PROJECT NO. 3095; TASK NO. 06; WORK UNIT ACCESSION NO. 29

11. TITLE (Include Security Classification)
PRODUCT DEFINITION DATA INTERFACE (PDDI), Product Specification

12. PERSONAL AUTHOR(S)
(see reverse side)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM _____ TO _____	July 1991	1049

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Product Definition Data; ICAM Architecture; Life Cycle Document; CAD/CAM; Engrg./Mfg. Interface; (continued on back)
1308	0905		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This document is the Product Specification for the Product Definition Data Interface (PDDI) Extensions. This document provides the description of the Computer Program Components (CPCS) deliverable for the PDDI program.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Alan Winn	(513) 255-8787	WL/MTIB

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

12. Personal Author (s):

Altemueller, Jeffrey
Chi, Kelly
Baldridge, Gary
Davis, Lori
Dorr, Phillip
Magnuson, Charles
Melh, Kenneth
Oakes, Janet
Shreve, Edward
Ulmer, Beth
Whelan, Anna

18. Subject Terms:

Needs Analysis Document
System Requirement Document
State-of-the-Art Document
System Specification Document
SS - Draft Standard
System Design Specification
Operators Manual
Users Manual - Access Software
Users Manual - Translator

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Alan R. Winn — 30 May 91

ALAN R. WINN
Project Manager

DATE

FOR THE COMMANDER:

Bruce Rasmussen — 31 May 91

BRUCE A. RASMUSSEN, Chief
Integration Technology Division
Manufacturing Technology Directorate

DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WL/MTIB , WPAFB, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

FOREWORD

This document was produced under Air Force Contract F33615-82-C-5036, Product Definition Data Interface (PDDI). This contract is sponsored by the Air Force Wright Aeronautical Laboratories, Materials Laboratory, Air Force Systems Command, Wright-Patterson, Air Force Base, Ohio.

The program is being administered under the technical direction of Lt. Eric Gunther, ICAM Project Manager. The MCAIR Program Manager is Mr. Jerry Weiss and Mr. Herb Ryan is the Deputy Program Manager.

This document was prepared in accordance with the ICAM Configuration Management Life Cycle Documentation requirements for the Configuration Item.

TABLE OF CONTENTS

		PAGE
Section 1	SCOPE	1-1
1.1	Identification	1-1
1.2	Introduction	1-1
1.3	Other PDDI Documents	1-1
1.4	Approach	1-2
Section 2	REFERENCES	2-1
2.1	Applicable Documents	2-1
2.1.1	Specifications	2-1
2.1.2	Standards	2-1
2.1.3	Other Publications	2-1
2.2	Terms and Abbreviations	2-2
Section 3	SYSTEMS OVERVIEW	3-1
3.1	System Overview	3-1
3.2	PDDI Access Software	3-1
3.3	PDDI Translator	3-3
3.4	System Interfaces	3-3
3.5	System Enviornment	3-5
Appendix A	Glossary	A-1
Appendix B	Translator Software Routines	B-1
Appendix C	Translator Data Dictionary	C-1
Appendix D	Access Software Hierarchy	D-1
Appendix E	Access Software Routines	E-1
Appendix F	Access Software Data Dictionary	F-1
Appendix G	PDDI Data Dictionary (Schema)	G-1

LIST OF ILLUSTRATIONS

Figure 3-1	PDDI Architecture	3-2
Figure 3-2	PDDI Access Software Schema	3-3
Figure 3-3	PDDI Translator Architecture	3-5

SECTION 1

SCOPE

1.1 Identification

This Product Specification manual describes the "As Built" software specifications for the Product Definition Data Interface (PDDI) Project 5601. This project was developed under the Air Force Contract F33516-82-C-5036.

1.2 Introduction

This manual is a reference document for programming personnel who maintain and enhance the PDDI software. The remainder of this document presents the PDDI data in detail at three levels of complexity; the individual data entities (data dictionaries), the routines comprising the software and the relationships between these routines. A glossary is also provided to clarify unique PDDI terms.

Three data dictionaries are presented to provide a complete listing of data within the Translator, the Access Software and the entire PDDI schema.

The software routines are listed for the Translator and the Access Software. Routines are groups of code which perform a specific function. The software hierarchy presents the relationships between these routines.

1.3 Other PDDI Documents

The PDDI Access Software User's Manual provides a guide for application programmers to use the Access Software capabilities. Described in this manual is Access Software Initialization, Entity Creation, Deletion and Manipulation, and List Operations.

The associated PDDI Operator's Manual (OM) provides a guide for PDDI software installation and IBM to VAX conversion procedures. The PDDI Translator User's Manual (UM) provides a guide for use of the PDDI Translator.

The PDDI software was designed to be transportable and has been operated on IBM 43xx and DEC VAX 11/780 computers.

This manual provides IBM software descriptions only.

1.4 Approach

This Product Specification is divided into three (3) main sections: Scope, References, and System Overview. The appendices provide supplemental information.

Section 1 - Scope of this document.

Section 2 - Reference documentation applicable to PDDI and this document.

Section 3 - A PDDI System Overview.

Appendix A - Glossary of Terms used in this document.

Appendix B - Translator Software Routines with Hierarchy Chart.

Appendix C - Translator Data Dictionary.

Appendix D- Access Software Hierarchy.

Appendix E- Access Software Routines.

Appendix F- Access Software Data Dictionary.

Appendix G- PDDI Data Dictionary (Schema).

SECTION 2

REFERENCES

2.1 Applicable Documents

2.1.1 Specification:

DOD-D-1000B	Drawings, Engineering and Associated Lists
MIL-D-5840	Requirements for Data, Engineering and Technical Reproduction

2.1.2 Standards:

ANSI Y14.5	Dimensioning and Tolerancing
ANSI Y14.26M	Digital Representation for Communication of Product Definition Data
ANSI B46.1	Surface Texture (Surface Roughness, Waviness and Lay)
ANSI B92.1	Involute Splines and Inspection
DOD-STD-100C	Engineering Drawing Practices
MIL-STD-9	Screw Thread Conventions and Methods of Specifying
MIL-STD-12	Abbreviations for Use on Drawings, Specifications, Standards and in Technical Documents
IDS150120000C	ICAM Documentation Standards
ANSI/IEEE STD 829	Standards for Software Test Documentation

2.1.3 Other Publications:

CLD150120000	ICAM Document Catalog
FTR110210000U	ICAM Architecture
FTR110232000U	ICAM Architecture Part II, Automated IDEFO Development

Product Definition Data Interface

ITR560130001U	First Interim Technical Report (Period 1 Oct 82 - 31 Dec 82)
ITR560130002U	Second Interim Technical Report (Period 1 Jan 83 - 31 Mar 83)
ITR560130003U	Third Interim Technical Report (Period 1 Apr 83 - 30 Jun 83)
ITR560130004U	Fourth Interim Technical Report (Period 1 Jul 83 - 30 Sep 83)
ITR560130005U	Fifth Interim Technical Report (Period 1 Oct 83 - 1 Dec 83)

ITR560130006U	Sixth Interim Technical Report (Period 1 Jan 84 - 31 Mar 84)
ITR560130007U	Seventh Interim Technical Report (Period 1 Apr 84 - 30 Jun 84)
ITR560130008U	Eighth Interim Technical Report (Period 1 Jul 84 - 30 Sep 84)
ITR560130009U	Ninth Interim Technical Report (Period 1 Oct 84 - 31 Dec 84)
ITR560130010U	Tenth Interim Technical Report (Period 1 Jan 85 - 31 Mar 85)
FTR560130001U	Task I Final Report - System Test Methodology, Volume III
	Technical Operating Report - Product Assurance/Quality Assurance - 15 Oct 84
SD 560130001U	Scoping Document
NAD560130000	Needs Analysis Document
SAD560130000	State-of-the-Art Document
SRD560130000	System Requirement Document
SS 560130100	System Specification Document
SS 560130200	System Specification Document - Draft Standard
SDS560130000	System Design Specification Document
STP560130000	System Test Plan
STR560130000	System Test Report
OM 560130000	Operator's Manual
UM 560130000	User's Manual - Translator
UM 560130001	User's Manual - Access Software

2.2 Terms and Abbreviations

Refer to Appendix A for Glossary.

SECTION 3

SYSTEM OPERATIONS

3.1 System Overview

The purpose of the PDDI Software System is to provide a prototype for the communication of complete Production Definition Data (PDD) between dissimilar CAD/CAM Systems. This system will serve as the information interface between Engineering and Manufacturing functions. It is composed of Access Software, Conceptual Schema, Exchange Format and a Translator. (See Figure 3-1).

The Access Software is a set of callable utility programs that will allow applications to manipulate and query PDD. The Conceptual Schema is a data dictionary that contains the data needed to define a CAD/CAM model. The Exchange Format is a neutral physical sequential format for passing data between dissimilar systems. The PDDI Translator is the software mechanism for passing this data between the Exchange Format and the Working Form of the PDD.

Physical Schemas

The Working Form physical schema is determined through a data dictionary or PASCAL include files. The Exchange Format physical schema is defined by a data dictionary and the specification for the neutral file format.

Software Packages

The software for the system consists of two (2) packages - Access Software and Translator.

3.2 Access Software

The PDDI Access Software package is an integrated set of routines that create and manage an incore Working Form of the PDDI data structure through key access. This Access Software keeps the application independent of the actual physical definition of the Working Form. It also serves as a bridge between existing CAD/CAM systems and the PDDI Exchange Format. The PDDI Access Software reduces the task of writing the Exchange Format by providing the utility functions for initializing the Working Form model, manipulating entities, and maintaining lists.

The PDDI Access Software operates on the data structure of the application and the Working Form, by using either entity or list operations. The entity operations allow the user to create, delete, modify and query entities. List operations manage the lists which are temporary data structures containing references to entities (keys). An application can build and maintain lists specifically for its needs.

The Access Software package allows the structuring of the user data. The entities can be related in user/constituent order. An entity may be related to multiple user entities, creating a network structure in the Working Form. An entity may also contain multiple constituent entities. Figure 3-2 shows the Schema for PDDI Access Software Interface routines.

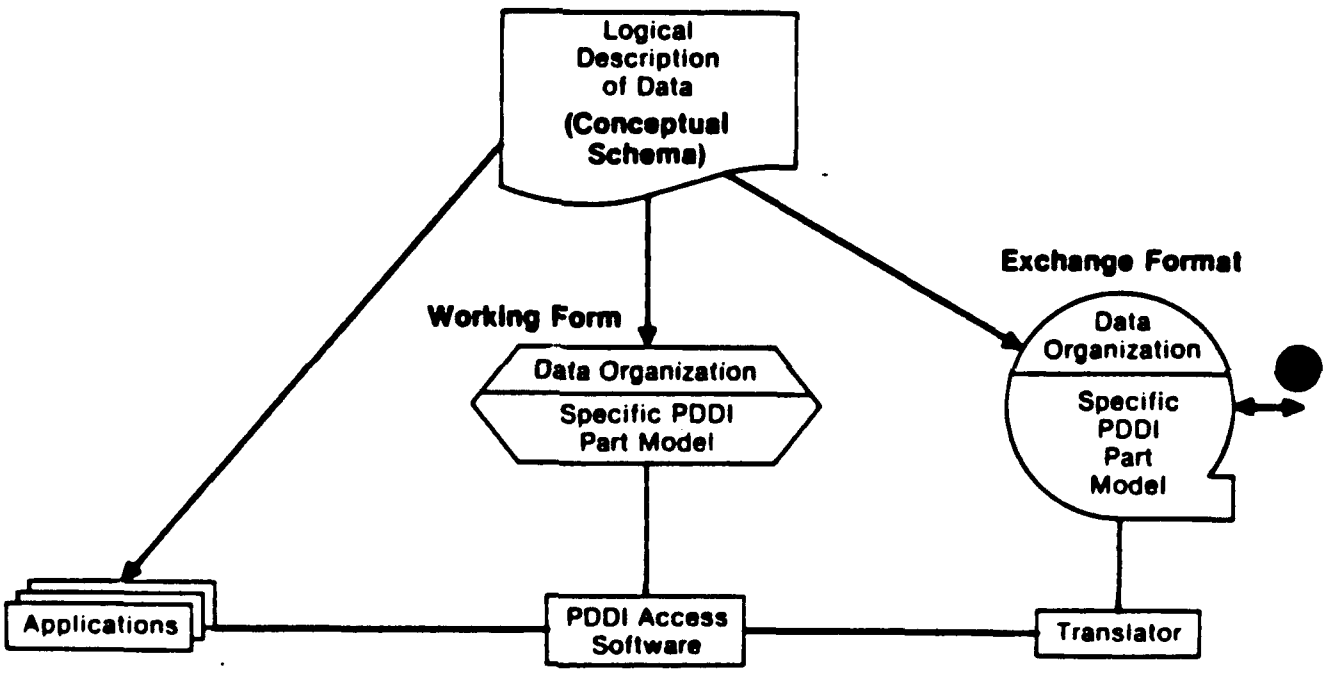


Figure 3-1 PDDI Architecture

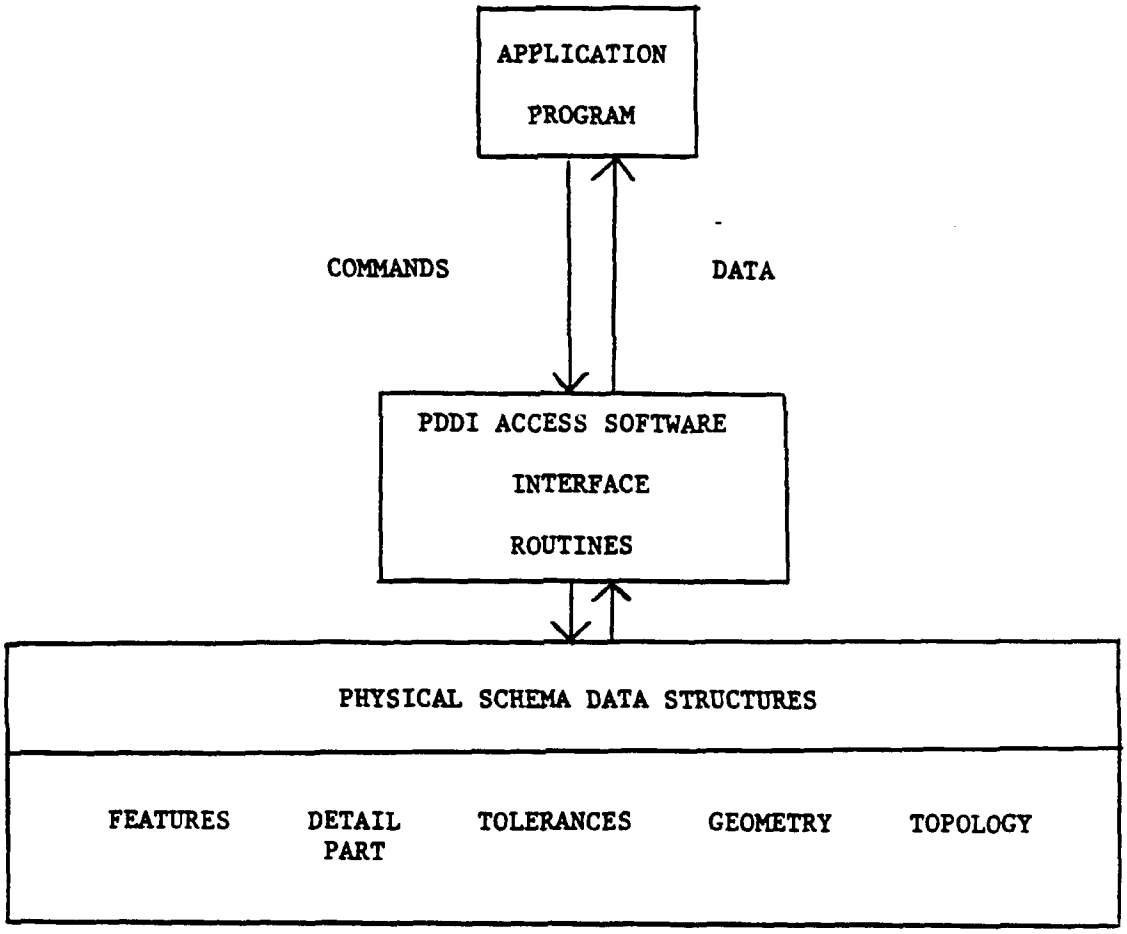


Figure 3-2 Access Software Schema

3.3 Translator

The PDDI Translator is the software package used to transmit the PDD between systems. The Translator consists of three main sub-packages. These sub-packages are: "Run System", "Pre-Processor" and "Post-Processor". (See Figure 3-3).

The Run System is the interface between the user and the "processors". This package provides menus, queries and system responses to the user.

Functions of this package include: Perform system configuration activities, determine files needed by the processors and make them available, and provide messages to aid user interfaces.

Access to the native database is provided for by this package via calls to user-supplied routines. These routines allow data from the native database to be placed into or obtained from the Working Form using calls to the Access Software. The pre-processor or post-processor is then called to perform the desired translation.

The Pre-Processor provides the interface from the Working Form to the Exchange Format.

Working Form entities, in the Working Form physical schema, are accessed via the Access Software. Tables, obtained from the Run System, are then used to map the Working Form entities to the Exchange Format physical schema. The Exchange Format entities are then encoded and placed into the Exchange Format file.

Transfer data is collected during entity processing. This data is encoded and placed into the Exchange Format file.

Error messages or condition codes are sent to the "Run System" to indicate the status of the transfer.

The Post-Processor provides the interface from the Exchange Format to the Working Form.

A set of tables, obtained from the Run System, is used to map the Exchange Format entities to the Working Form physical schema. The Access Software is then used to place these entities into the Working Form.

3.4 System Interfaces

The PDDI software must interface with the computer system on which it is installed, the local (native) CAD/CAM database, the Exchange Format, the Working Form, and the user (application). It does this via the PDDI Access Software, the PDDI Translator and local (native) developed software packages. The left-hand side of Figure 3-4 shows the PDDI development environment.

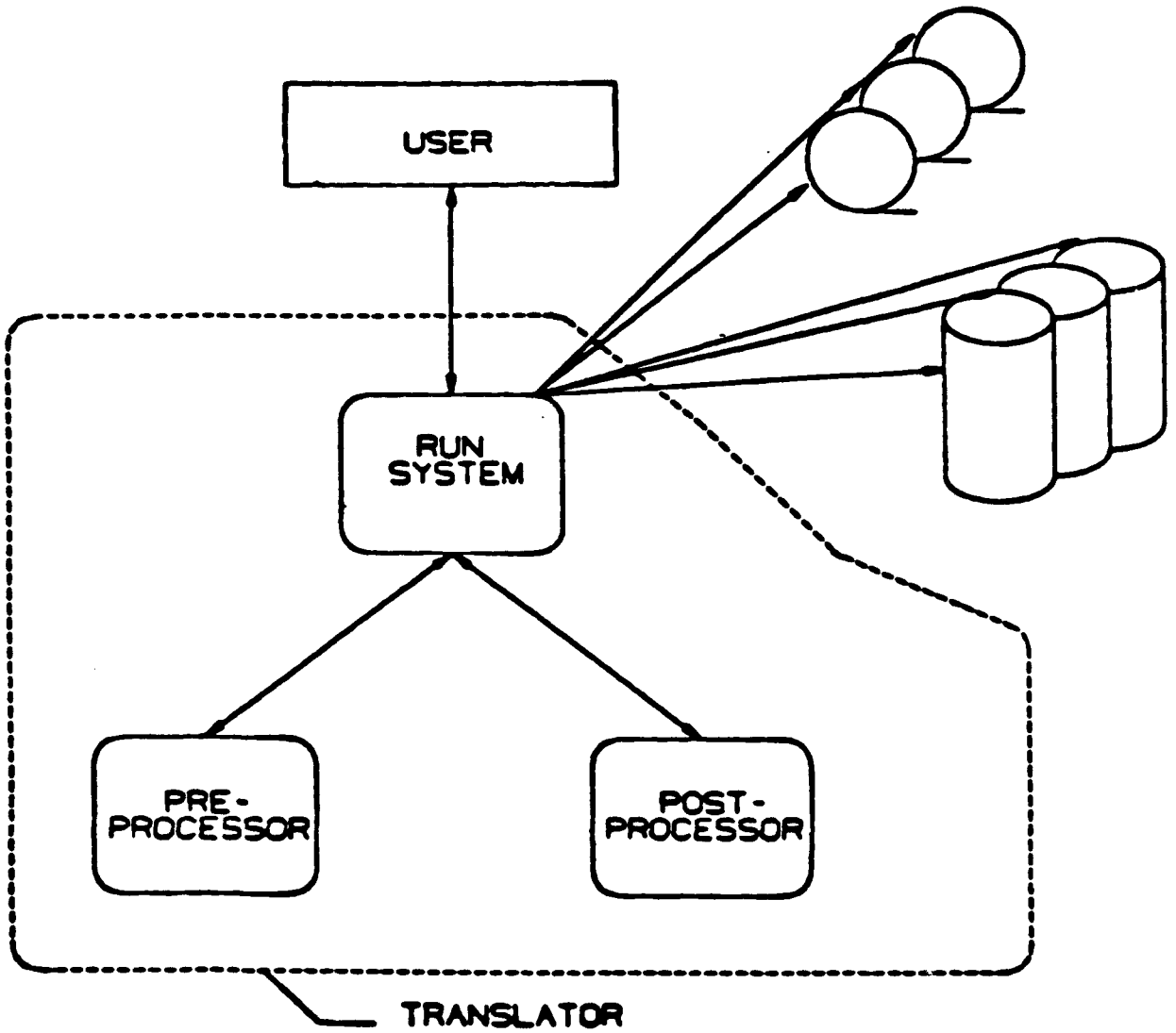


Figure 3-3 PDDI Translator Architecture

3.5 System Environment

The PDDI system was developed in the following computing environment:

Computer/Operating System

IBM 43XX/MVS with TSO and associated tape drives, disk drives and terminals.

DEC VAX 11/780 VMS with associated tape drives, disk drives and terminals.

Storage (Core) Requirements

The development PDDI system required the following core requirements:

Model (large)	.57M	
PDDI Software	.66M	(No Overlay)
Database	3.00M	
	4.23M	

Compilers

IBM-Pascal/VS Release 2.2
DEC-Pascal V3.3, FORTRAN 77 V4.4

Terminals

E&S PS300 (or equivalent for graphics applications)

The PDDI system is transportable to other computing systems. However, appropriate local (native) interfaces must be provided. The right-hand side of Figure 3-4 shows the PDDI commercial demonstration architecture for UNIGRAPHICS and Computervision and United Technologies Research Center (UTRC) Systems.

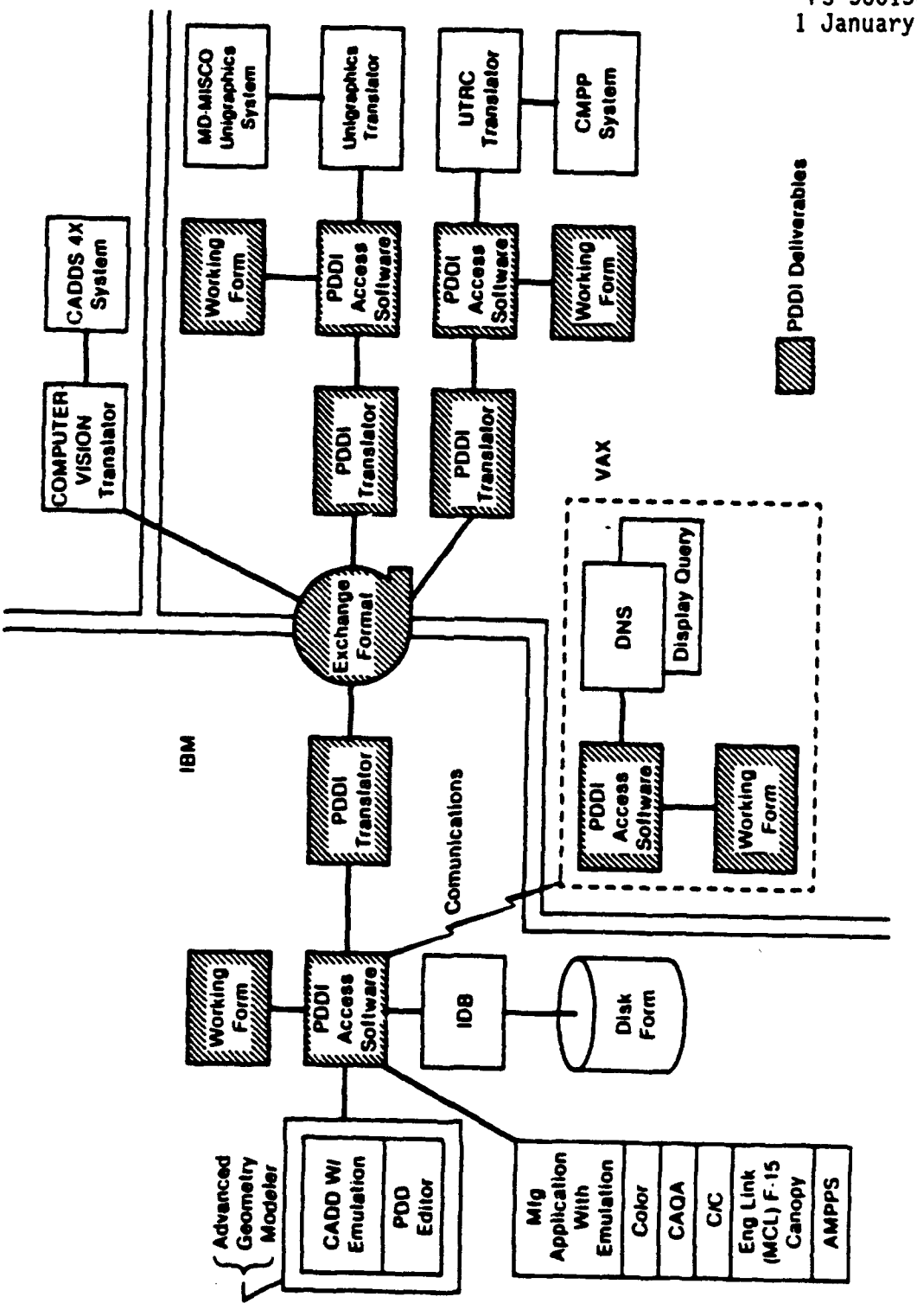


Figure 3-4 PDDI Environment

APPENDIX A

GLOSSARY

ACCESS SOFTWARE - A set of routines for creating, managing and querying an incore Working Form model.

ANSI - American National Standards Institute.

APPLICATION - Refers generically to any software modules which are used in CAD/CAM functions.

APPLICATION REQUEST - A request initiated by an application program, either through batch or interactive processing, which will interrogate the model through the PDDI Access Software to obtain or operate on specific information regarding the model and its components or elements.

APPLICATION REQUESTED DATA - The data which fulfills the application's original request and which is in the proper format and readable by the application.

ASCII - American Standard Code for Information Interchange.

ATTRIBUTE - An item of information about an entity. A key attribute identifies the entity; a role attribute gives a fact about an entity.

CAD/CAM - Computer Aided Design/Computer Aided Manufacturing.

CLASS - A collection of entities that are alike in some manner.

CLIST - IBM Command lists.

CONSTITUENT - A specific instance of an entity that is used in the definition of some other entity.

CONTEXT-FREE GRAMMAR - The syntax of the language gives a precise specification of the data without interpretation of it.

CRB - Constituent Read Block.

CSECT - Control Section, the name sometimes used for a routine in software descriptions.

DOMAIN - The set of values permissible in a given context. A natural domain is the value set native to a given machine architecture; an imposed domain is a specific subset of the natural domain.

DYNAMIC ALLOCATION - The allocation (and deallocation) of memory resources as required by the application. The opposite is static allocation where a fixed size segment of memory is available to the application.

EBCDIC - Extended Binary Coded Decimal Interchange Code (IBM character set).

ENTITY - A collection of facts (attributes) about something of interest.

EXTERNAL REFERENCE - A reference to some quantity of data that exists somewhere outside the scope of the immediate body of information.

FUNCTIONALITY - (1) To show that the configuration item has fulfilled the specified requirements. (2) The receiving and sending systems can operate on the entity in the same manner with the same results within a pre-defined tolerance.

IDB - Integrated Database, MCAIR internal database system.

IIT - Internal Item.

INCLUDE FILE - Pascal source code from another file or library included on the compilation of a Pascal source file.

INPUT DATA - That information which the application needs to supply in order to interrogate or operate on the model. This data may assume only these forms prescribed by the PDDI Access Software specifications.

INTERPRETED REQUEST - Input data which has been appropriately modified to conform to the PDDI Access Software's internal data representation so that it may be further processed.

JCL - Job Control Language - IBM language used to identify a job and describe its requirements to an operating system.

KEY - An item of data that uniquely identifies some specific instance of an entity.

MAS - The MCAIR acronym with the PDDI Access Software (Model Access System).

METAMODEL - A body of data that defines the characteristics of a data model or structure.

MODEL - A collection of PDD that is transferable, displayable, accessible, and equivalent to a Part. The internal representation of the application data, as initiated and organized by the user. The model is also referred to as the Working Form.

MODEL NETWORK DEFINITION - The set of rules and definitions which outline in detail the data structure whereby higher order entities may be composed of lower order entities, or constituents, and the lower order entities may be constituents of one or more higher order entities.

NATIVE SYSTEM - The PDD and applications in a format that is unique to the database of a CAD system.

NDS - Network Data Structure.

PARSE - The process of analyzing input strings (records) to identify fields and to verify that the data has a valid format.

PDD - Product Definition Data.

POST-PROCESSOR - A phase of the translator where data is received from the Exchange Format and is converted to the Working Form.

PRE-PROCESSOR - A phase of the translator where data is taken from the Working Form and is converted to the Exchange Format.

QUALITY - The composite of all the attributes or characteristics including performance of an item or product.

QUALITY ASSURANCE - The planned and systematic establishment of all actions (management/engineering) necessary to provide adequate confidence and nonconformance prevention provisions and reviews are established during the design phase and performed throughout the software development and life cycle phases.

QUALITY CONTROL - The planned and systematic application of all actions (management/technical) necessary to control raw materials or products through the use of test, inspect, evaluate, and control of processes.

REQUESTED DATA - See Application Request Data.

RUN SYSTEM - The Translator sub-package which provides the communication interface between the user and the pre/post-processors.

SCHEMA - Those definitions which describe the content of the data and the relationship between the various elements or components of the data.

SOFTWARE QUALITY ASSURANCE (SQA) - The planned and systematic establishment of all actions necessary to provide adequate confidence that nonconformance prevention provisions and reviews are established during the design phase and performed throughout the software development and life cycle phases.

SOFTWARE QUALITY ASSURANCE PLAN (SQAP) - An organized description of the methods, policies, and procedures necessary to conduct software quality assurance and control activities during the design, development, delivery, and maintenance phases.

SOFTWARE QUALITY CONTROL - The planned and systematic application of all actions (management/technical) necessary to ensure that the software under development or maintenance satisfies the technical requirements through the use of tests, demonstrations, inspections, evaluations, and control of processes.

SYSTEM CONSTRAINTS - Those hardware and software environmental constraints which will be imposed upon the PDDI Access Software that will limit its implementation and application. An example of such constraints might be the particular compiler used to compile the PDDI Access Software package.

TRANSFER DATA - The data required to make an exchange of data between systems (e.g., delimiters, record counts, record length, entity counts, numeric precision).

TRANSLATOR - A software program that is used for passing data between the Exchange Format and Working Form of the PDD.

TSO - Time Sharing Option - IBM function which provides conversational time sharing from remote terminals.

UDB - User Data Block (AKA, Attribute Data Block).

USER COMPUTER SYSTEM - The specific hardware, operating systems, and applications software systems that the user will employ to implement the PDDI Access Software.

WORKING FORM - A memory resident form of a model that supports rapid access to entities via the Access Software.

WORKING FORMAT - The physical representation of the Working Form within the computer.

APPENDIX B

TRANSLATOR SOFTWARE ROUTINES

This appendix provides a listing of each routine in the Translator Software package. The routines are listed in alphabetic order. Hierarchy charts are included and an index gives a brief description of each routine function.

Routine Index. B-2
Post-Processor Hierarchy Chart B-4
Pre-Processor Hierarchy Chart. B-10
Routine Dictionary B-13

PDDI TRANSLATOR SOFTWARE ROUTINE INDEX

ACSSWF - retrieves the working form model
ADDCNST - adds an entity to a constituent list at a specified position, adding nil entities if necessary for padding
ATRDAT - Creates Data Section from Attribute data
CONTOK - converts a field from character type to whatever type is specified
CONTROL - Controls the flow of entire system
CONVRT - convert to or from native form
CRDECL - create declaration section of PDES file
CRHEAD - create Header Section of PDES file
CRRULE - create Rules Section of PDES file
DOTOP - set up parameters for routine ATRDAT
ENCODE - Create an Exchange Format Entity
FILEPTR - update or query the relationship structure of file pointer to kind and ident
FILEWF - file the working form model to native database
FILLADB - to fill values in the attribute data block of the current entity
FILLCNST - to fill values in the constituent list of the current entity
FILLSTRC - to fill values in structures
FILLSUB - to fill values in the current subentity
FILLTOP - places values that are common to all kinds in the attribute data block for the current entity
FILLVAL - to fill values in the working form
FORINFO - Formats user provided data
FTCHADB - to read the attribute data block and give the actual value in the data field.
FTCHCHR - Obtains a character string from the ADB
FTCHHAF - Obtains a 2-byte integer from the ADB
FTCHINT - Obtains an integer from the ADB
FTCHLOG - Obtains a logical from the ADB
FTCHREL - Obtains an 8-byte real number from the ADB
FTCHSCL - Obtains a 1-byte integer from the ADB
GETCOMM - to extract a comment and write it to the message file
GETDDPOS - gets the position in the current data dictionary of the next exchange format field.
GETHI - Obtains current highest instance of specified entity kind
GETIND - Gets the instance of an entity kind
GETKND - gets the kind corresponding to the string name
GETMTOK - gets the next meaningful token, or a null string if there is a defaulted value.
GETNEW - Reads the Data Dictionary of a specified entity
GETQUOT - gets a character string from the exchange format file
GETSCRIN - gets the position in the current data dictionary of the next exchange format field.
GETSEMI - sets the position to after the next semi-colon - the beginning of the next entity.
GETTOKEN - gets a token from the exchange format file
GETUNI - gets the starting line in the data dictionary
GETWKF - Gets the information about the Working Form entity

INITLR -initializes the level record
INITMAP - initializes the mapping array
INITNEW - Initializes Data Dictionary to blanks
KNDLIS - reads a list of kinds and their associated entity names
MAERRM - Produces Input Error Message
MAKDAT - Makes Exchange Format Entity
NILCON - Writes the appropriate punctuation to the data section record for a nil attribute
OPERAT - Converts Values to character form
PERROR - communicates with the user
POST - Post-process PDD
PRDATA - puts data information from exchange file into the working form
PRDECL - processes declaration information from the exchange file
PRE - Pre-process PDD
PRHEAD - processes header information from the exchange file
PRTENT - Prints the definitions of all entities in the input list
PRTOKEN - processes an input token RETMAP - retrieves mapping information
PUNTAB - Generates Delimiters for Fields
READEF - Reads Exchange Format Record
RETMAP Retreives mapping information
STEP - Steps through processors
STORADB - puts a value in the attribute data block
STORCHR - Puts a character string into the ADB
STORHAF - Puts a scalar into the ADB
STORINT - Puts an integer into the ADB
STORLOG - Puts a logical into the ADB
STORMAP Stores the Exchange Format pointer and associated entity key in an array
STORREL - Puts an 8-byte real number into the ADB
STORSCL - Puts a scalar into the ADB
STRNAM - gets the string name corresponding to the input kind VALCON - validates the field
TABRED - Converts a value to character and concatenate punctuation onto it for Exchange Format file
TRMPRT - Determines whether a print "all entity" or "by kind of entity" will run on the Working Form model
VALCON validates the field
WFPRNT - Initiates the print of a Working Form model
WRITEMSG - writes output to the message file
WRIREC - Writes a string to Record and File

Routine Hierarchy for the POST Processor
Exchange Format / Working Format Translator
(IBM Version)

Legend:
- * MAS Rountine
- \+ Stub Rountine
- ** IDB Rountine (Not Deliverable)
- ++ McAir Utility (Not Deliverable)

```
USERMENU   ( CLIST )
  CONTROL
    FORINFO
    STEP
      PERROR
      MAINIT
      MAKILL
      FILEWF  ++
      CONVRT  ++
      ACSSWF  ++
      WFPRNT
        TRMPRT
          GETNEW
            INITNEW
          PRTENT
            MALRD  *
            MALSTF  *
            MALD  *
            MALNO  *
            MALGTK  *
            MAEU  *
            FTCHADB
              FTCHCHR
              FTCHINT
                FTCHSCL
                FTCHHAF
              FTCHLOG
              FTCHREL
              FTCHSCL
          MAECTK  *
          MAEKND  *
          MALK  *
          MALNO  *
      PRE
      POST
        PERROR
        MAERRM  *
        MAECR  *
        FILLTOP
```

```
  GETHI
    MAERRM  *
    MALK  *
    MALD  *
    MALNO  *
    MALSTF  *
    MALRD  *
    MAEGTK  *
    PERROR
  STORADB
    STORINT
      STORSCL
      STORHAF
    STORREL
    STORCHR
    STORLOG
    STORSCL
GETNEW
  INITNEW
GETMTOK
  GETTOKEN
    PERROR
    READEF
    GETCOMM
      PERROR
      WRITEMSG
        PERROR
      READEF
    GETQUOT
      PERROR
      READEF
PRHEAD
  PERROR
  WRITEMSG
    PERROR
PRDECL
  PERROR
  WRITEMSG
    PERROR
PRDATA
  PERROR
  MAERRM  *
  MAECR  *
  MALD  *
  STORMAP
  ADDCNST
    PERROR
    MAERRM  *
    MALNO  *
    MALRPL  *
    MALATC  *
```

```
GETDDPOS
FTCHADB
  FTCHCHR
  FTCHINT
    FTCHSCL
    FTCHHAF
  FTCHLOG
  FTCHREL
  FTCHSCL
PRTOKEN
  PERROR
  GETNEW
    INITNEW
  GETKND
  FILLTOP
    GETHI
      MAERRM  *
      MALK  *
      MALD  *
      MALNO  *
      MALSTF  *
      MALRD  *
      MAEGTK  *
      PERROR
    STORADB
      STORINT
        STORSCL
        STORHAF
      STORREL
      STORCHR
      STORLOG
      STORSCL
  INITLR
  FILLVAL
    PERROR
    GETDDPOS
    FILLADB
      PERROR
      CONTOK
        VALCON
      STORADB
        STORINT
          STORSCL
          STORHAF
        STORREL
        STORCHR
        STORLOG
        STORSCL
    FILLCNST
      PERROR
      MAERRM  *
```

```
MAL  *
MALD  *
MALATC  *
MAECR  *
INITLR
GETDDPOS
GETNEW
  INITNEW
ADDCNST
  PERROR
  MAERRM  *
  MALNO  *
  MALRPL  *
  MALATC  *
FILLTOP
  GETHI
    MAERRM  *
    MALK  *
    MALD  *
    MALNO  *
    MALSTF  *
    MALRD  *
    MAEGTK  *
    PERROR
  STORADB
    STORINT
      STORSCL
      STORHAF
    STORREL
    STORCHR
    STORLOG
    STORSCL
CONTOK
  VALCON
RETMAP
FILLSUB
  PERROR
  MAERRM  *
  MAL  *
  MALATC  *
  MAECR  *
  MALD  *
  INITLR
  FILLTOP
    GETHI
      MAERRM  *
      MALK  *
      MALD  *
      MALNO  *
      MALSTF  *
      MALRD  *
```

```
            MAEGTK  *
            PERROR
          STORADB
            STORINT
              STORSCL
              STORHAF
            STORREL
            STORCHR
            STORLOG
            STORSCL
        GETNEW
          INITNEW
        ADDCNST
          PERROR
          MAERRM  *
          MALNO  *
          MALRPL  *
          MALATC  *
        GETDDPOS
    FILLSUB
      PERROR
      MAERRM  *
      MAL  *
      MALATC  *
      MAECR  *
      MALD  *
      INITLR
      FILLTOP
        GETHI
          MAERRM  *
          MALK  *
          MALD  *
          MALNO  *
          MALSTF  *
          MALRD  *
          MAEGTK  *
          PERROR
        STORADB
          STORINT
            STORSCL
            STORHAF
          STORREL
          STORCHR
          STORLOG
          STORSCL
      GETNEW
        INITNEW
```

```
          ADDCNST
            PERROR
            MAERRM  *
            MALNO   *
            MALRPL  *
            MALATC  *
          GETDDPOS
        FILLSTRC
KNDLIS
GETSEMI
  PERROR
  READEF
INITMAP
```

Routine Hierarchy for the PRE Processor
Exchange Format / Working Format Translator
(IBM Version)

Legend:
* MAS Rountine
+ Stub Rountine
** IDB Rountine (Not Deliverable)
++ McAir Utility (Not Deliverable)

```
USERMENU   ( CLIST )
  CONTROL
    FORINFO
    STEP
      PERROR
      FILEWF ++
      CONVRT  ++
      ACSSWF ++
      WFPRNT
        TRMPRT
          MAECTK  *
          MAEKND  *
          MALK  *
          MALNO  *
          GETNEW
            INITNEW
          PRTENT
            FTCHADB
              FTCHCHR
              FTCHINT
                FTCHSCL
                FTCHHAF
              FTCHLOG
              FTCHREL
              FTCHSCL
            MALRD  *
            MALSTF  *
            MALD  *
            MALNO  *
            MALGTK  *
            MAEGTK  *
            MAEU  *
      PRE
        PERROR
        MAERRM  *
        MALNO  *
        MALSTF  *
        MALRD  *
        MALK  *
```

```
KNDLIS
WRTREC
ENCODE
  PERROR
  MAKDAT
    PERROR
    FTCHADB
      FTCHCHR
      FTCHINT
        FTCHSCL
        FTCHHAF
      FTCHLOG
      FTCHREL
      FTCHSCL
    STRNAM
    FILEPTR
    MAEGTK  *
    GETNEW
      INITNEW
    WRTREC



    ATRDAT
      FILEPTR
      GETUNI
      DOTOP
        WRTREC
        GETSCRIN
      GETSCRIN
      GETNEW
        INITNEW
      WRTREC
      FTCHADB
        FTCHCHR
        FTCHINT
          FTCHSCL
          FTCHHAF
        FTCHLOG
        FTCHREL
        FTCHSCL
      GETIND
        FTCHADB
          FTCHCHR
          FTCHINT
            FTCHSCL
            FTCHHAF
```

```
            FTCHLOG
            FTCHREL
            FTCHSCL
        MAEGKN  *
        MALGTK  *
        MAEGTK  *
        MAEC  *
        MALSTF  *
        MALRD  *
        MALNO  *
        NILCON
          GETSCRIN
          WRTREC
        OPERAT
          PERROR
          WRTREC
          TABRED
            PUNTAB
        PERROR
  GETWKF
    PERROR
    MAERRM  *
    MAEGTK  *
    MAEC  *
  CRHEAD
    PERROR
    MAERRM  *
    MAECTK  *
    MAECTK  *
    MAEKND  *
    MALK  *



    MALNO  *
    MALD  *
    WRTREC
    STRNAM
  CRDECL
  CRRULE
MAINIT  *
MAKILL  *
POST
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN          K315 2G  CREATED: 86/10/29       *)
(*    VERSION: 1                             REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : ACSSWF                                          *)
(*                                                                   *)
(*    FUNCTION : RETRIEVES THE WORKING FORM MODEL                    *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*         STEP                                                      *)
(*      CALLS:                                                       *)
(*         ENRTV                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*      USEREC   I   A RECORD OF USER INFORMATION                    *)
(*      IRC      O   RETURN CODE                                     *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE RETRIEVES THE WORKING FORM MODEL FROM EITHER  *)
(*    THE NATIVE DATABASE OR THE PDDI SUPPLIED DATABASE              *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*  AUTHOR:   PHIL DORR                ORG_ID  CREATED:  06/06/86  *)
(*  VERSION:  1                                REVISED:             *)
(*                                                                 *)
(*  ROUTINE NAME : ADDCNST                                         *)
(*                                                                 *)
(*  FUNCTION :                                                     *)
(*    ADDS AN ENTITY TO A CONSTITUENT LIST AT A SPECIFIED POSITION,*)
(*    ADDING NIL ENTITIES IF NECESSARY FOR PADDING.                *)
(*                                                                 *)
(*  ENVIRONMENT:                                                   *)
(*    IBM PASCAL LANGUAGE                                          *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.     *)
(*                                                                 *)
(*  EXECUTION PROCEDURE:                                           *)
(*  CALLS:                                                         *)
(*    PERROR          WRITES OUT ERROR MESSAGES                    *)
(*    MALNO           DETERMINES THE NUMBER OF ELEMENTS IN LIST    *)
(*    MALRPL          REPLACES AN ENTITY IN A LIST                 *)
(*    MALATC          ATACHES AN ENTITY TO THE END OF A LIST       *)
(*  CALLED BY:                                                     *)
(*    PRDATA                                                       *)
(*    FILLCNST        FILLS THE CONSTITUENT LIST                   *)
(*    FILLSUB         FILLS THE SUBENTITY                          *)
(*                                                                 *)
(*  DESCRIPTION OF ARGUMENTS:                                      *)
(*    WFDD       I/O  THE ARRAY OF DATA DICTIONARYS                *)
(*    WFDD_POS    I   THE POSITION IN THE CURRENT WFDD             *)
(*    CLIST      I/O  THE ARRAY OF ENTITY CONTITUENT LISTS         *)
(*    CUR_LEVEL   I   THE CURRENT LEVEL WITHIN CLIST ARRAY         *)
(*    ENTITY_KEY  I   THE KEY OF THE ENTITY TO INSERT              *)
(*    RC          O   RETURN CODE                                  *)
(*                    0  : OK                                      *)
(*                    1  : NIL ENTITY NOT CREATED EARILIER - FAILURE*)
(*                    2  : MAS ERROR - FAILURE                     *)
(*                                                                 *)
(*  PROCESSING DESCRIPTION:                                        *)
(*    ADDS A KEY TO THE CONSTITUENT LIST IN THE APPROPRIATE POSITION*)
(*                                                                 *)
(*  CHANGE CONTROL:                                                *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 84/12/20  CC   *)
(*    VERSION: 1.0                             REVISED: 85/09/17  CC   *)
(*                                                                   *)
(*    ROUTINE NAME: ATRDAT                                           *)
(*              CREATE DATA SECTION FROM ATTRIBUTE DATA              *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        CREATES DATA SECTION RECORD INFORMATION FROM ATTRIBUTE      *)
(*        DATA BLOCK AND CONSTITUTENT LIST.                          *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        MAKDAT                                                     *)
(*        SELF (RECURSIVE ROUTINE)                                   *)
(*      CALLS:                                                       *)
(*        GETUNI                                                     *)
(*        FILEPTR                                                    *)
(*        GETNEW                                                     *)
(*        WRTREC                                                     *)
(*        FTCHADB                                                    *)
(*        GETIND                                                     *)
(*        NILCON                                                     *)
(*        OPERAT                                                     *)
(*        PERROR                                                     *)
(*        GETSCRIN                                                   *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND      I   ENTITY KIND                                  *)
(*        PASADB    I   ATTRIBUTE DATA BLOCK FOR ENTITY              *)
(*        EDBCOM    I   WORKING FORM DATADICTIONARY FOR KIND         *)
(*        CONSTI    I   WORKING FORM KEY TO ENTITY'S CONSTITUENT LIST *)
(*        EFFILE    I   EXCHANGE FORMAT FILE                         *)
(*        IRC       O   RETURN CODE                                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        USING THE DATA DICTIONARY, PROCESS THE DATA ASSOCIATED     *)
(*        WITH THE ENTITY.                                           *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        09/17/85 - L.A.DAVIS USES NEW GLOBAL EF DATADICTIONARY     *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: K. CHI                    W315 2G CREATED: 84/12/04    *)
(*    VERSION: 1.0                             REVISED:              *)
(*                                                                   *)
(*    ROUTINE NAME : CONTOK                                          *)
(*                                                                   *)
(*    FUNCTION : CONVERT FIELDS                                      *)
(*                CONVERT A FIELD FROM CHARACTER TYPE TO WHATEVER    *)
(*                TYPE IS SPECIFIED                                  *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        FILL-VALUE-IN-ATTRIBUTE-DATA-BLOCK                         *)
(*        FILL-VALUE-IN-CONSTITUENT-LIST                             *)
(*      CALLS:                                                       *)
(*        VALIDATE-CONVERSION-FIELD                                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        TOKEN    I   FIELD TO BE CONVERTED                         *)
(*        CONREC  I/O  RECORD OF INFORMATION TO BE PASSED IN AND OUT *)
(*        RC       O   RETURN CODE                                   *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*             THIS ROUTINE NEEDS NO OPEN/CLOSE FILES                *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  K. CHI                  W315 2G CREATED: 84/10/19     *)
(*    VERSION:  1.0                           REVISED: 85/09/17     *)
(*                                                     86/06/06     *)
(*    ROUTINE : CONTROL                                              *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*         CONTROLS THE FLOW OF THE ENTIRE SYSTEM.                   *)
(*         WHEN A MESSAGE NEEDS TO BE PRINTED OR HAS BEEN PRINTED    *)
(*         CONTROL WILL CALL THE APPROPRIATE SUBROUTINE FOR FURTHER   *)
(*         ACTION                                                    *)
(*         ENTRY POINT FOR ANY TRANSLATOR FUNCTION                   *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        THIS ROUTINE WILL BE CALLED BY A CLIST                     *)
(*        CALLS:                                                     *)
(*          FORINFO                                                  *)
(*          STEP                                                     *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*        U_REC    O   HOLDS THE USER RECORD.  IE. DRAWING NAME     *)
(*                                                 DISK/TAPE...      *)
(*                                                                   *)
(*        DSECT    O   FLAG THAT INDICATES WHETHER THE CURRENT      *)
(*                     DATA IS PART OF THE DATA SECTION.             *)
(*                     AT FIRST ENTRY, DSECT IS SET TO FALSE.        *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE NEEDS OPEN/CLOSE NO FILES                     *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              ORG_ID  CREATED: 84/12/11  CC  *)
(*    VERSION:  1.0                            REVISED: 86/06/10  CC  *)
(*                                                                   *)
(*    ROUTINE NAME : CONVRT                                          *)
(*                                                                   *)
(*    FUNCTION: CONVERT TO OR FROM NATIVE FORM                       *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        THIS ROUTINE WILL BE CALLED BY STEP                        *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME     I/O  DESCRIPTION                                  *)
(*        FORMAT    I   WHICH PROCESSOR IN USE                       *)
(*        IRC       O   RETURN CODE                                  *)
(*                      0 : OKAY                                     *)
(*                      2 : CONVERSION UNSUCCESSFUL                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR: A. M. WHELAN                ORG_ID  CREATED: 86/12/04    *)
(*  VERSION:  1                                 REVISED:             *)
(*                                                                   *)
(*  ROUTINE NAME : CRDECL                                            *)
(*                                                                   *)
(*  FUNCTION :  CREATE DECLARATION SECTION OF PDES FILE              *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    PRE                                                            *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0  : OK                                          *)
(*                  >0 : FAILURE                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            ORG_ID  CREATED: 86/12/04      *)
(*    VERSION: 1 ( JAN. DEMO )                REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : CRHEAD                                          *)
(*                                                                   *)
(*    FUNCTION : CREATE HEADER SECTION                               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        PRE                                                        *)
(*      CALLS:                                                       *)
(*        PERROR                                                     *)
(*        WRTREC                                                     *)
(*        STRNAM                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*         EFFILE  I   EXCHANGE FORMAT FILE                          *)
(*         KNDTBL  I   TABLE OF KINDS AND THEIR ASSOCIATED NAME      *)
(*         NUMKND  I   NUMBER OF ENTRIES IN KNDTBL                   *)
(*         IRC     O   RETURN CODE                                   *)
(*                        0 : OK                                     *)
(*                        1 : FAILURE                                *)
(*                        2 : MAS FAILURE                            *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        READ INFORMATION FROM HEADER FILE FOR HEADER               *)
(*        CREATE STATISTICS                                          *)
(*        COLLECT INFORMATION INTO HEADER                            *)
(*        WRITE HEADER TO THE FILE (WRTREC)                          *)
(*        READ COMMENTS FROM EXPLANATION FILE                        *)
(*        WRITE COMMENTS TO EXCHANGE FILE                            *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR: A. M. WHELAN                ORG_ID  CREATED: 85/12/04    *)
(*  VERSION:  1                                 REVISED:             *)
(*                                                                   *)
(*  ROUTINE NAME : CRRULE                                            *)
(*                                                                   *)
(*  FUNCTION : TO CREATE RULE SECTION OF PDES FILE                   *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    PRE                                                            *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0  : OK                                          *)
(*                  >0 : FAILURE                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 84/12/20  CC  *)
(*    VERSION: 1.0                             REVISED: 85/09/17  CC  *)
(*                                                                   *)
(*    ROUTINE NAME: DOTOP                                            *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        SET UP PARAMETERS FOR ATRDAT                               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          ATRDAT                                                   *)
(*        CALLS:                                                     *)
(*          WRTREC                                                   *)
(*          GETSCRIN                                                 *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME    I/O  DESCRIPTION                                   *)
(*        UNIDEX   I   BEGINNING LINE IN WFDD                        *)
(*        WEDONE   I   FLAG INDICATING END OF ENTITY                 *)
(*        EDBCOM   I   WORKING FORM DATADICTIONARY FOR KIND          *)
(*        WEHERE   I   CS ORDER PLACE HOLDER                         *)
(*        SCRIND   I   PLACE HOLDER IN WFDD                          *)
(*        BNAME    I   STRUCTURE NAMES                               *)
(*        INSTRC   I   FLAG INDICATING IN A STRUCTURE                *)
(*        MTSTRC   I   FLAG INDICATING AN EMPTY STRUCTURE            *)
(*        FFLAG    I   FIELD FLAG FOR PUNCTUATION                    *)
(*        SFLAG    I   STRUCTURE FLAG FOR PUNCTUATION                *)
(*        LEVEL    I   LEVEL OF STRUCTURES                           *)
(*        IRC      O   RETURN CODE                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            W315 2G CREATED: 86/05/10      *)
(*    VERSION: 1 ( JAN. DEMO )                REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : ENCODE                                          *)
(*                                                                   *)
(*    FUNCTION :  CREATE AN EXCHANGE FORMAT ENTITY                   *)
(*                CREATE EXCHANGE FORMAT ENTITIES FROM THE           *)
(*                WORKING FORM.                                      *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        PRE                                                        *)
(*      CALLS:                                                       *)
(*        PERROR                                                     *)
(*        MAKDAT                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND              I    THE WORKING FORM KIND             *)
(*        INDEX             O    THE INSTANCE IN KIND              *)
(*        WFKEY             I    THE WORKING FORM KEY              *)
(*        KNDTBL            I    A TABLE OF KINDS                  *)
(*                               AND THEIR ASSOCIATED NAMES        *)
(*        NUMKND            I    THE NUMBER OF ENTRIES IN KNDTBL   *)
(*        CURRENT_NUMBER   I/O   THE CURRENT EXCHANGE FORMAT INDEX  *)
(*        ENTADB            I    THEN ENTITY ATTRIBUTE DATA BLOCK  *)
(*        CONLIS            I    THE ENTITY CONSTITUENT LIST       *)
(*        EFFILE           I/O   THE EXCHANGE FORMAT FILE          *)
(*        IRC               O    THE RETURN CODE                   *)
(*                                 0 : OK                          *)
(*                                 2 : DATA ENTITY NOT MADE        *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE GETS THE INSTANCE OF THE KIND, AND CREATES    *)
(*        THE DATA ENTITY.                                           *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR: A. M. WHELAN            K315 2G CREATED: 85/12/04    *)
(*    VERSION: 1 ( JAN. DEMO )                REVISED:             *)
(*                                                                 *)
(*    ROUTINE NAME:  FILEPTR                                       *)
(*                                                                 *)
(*    FUNCTION :                                                   *)
(*        UPDATE OR QUERY THE RELATIONSHIP STRUCTURE OF FILE       *)
(*     POINTER TO KIND AND IDENT                                   *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*        IBM PASCAL LANGUAGE                                      *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W. *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*        CALLED BY:                                               *)
(*          MAKDAT                                                 *)
(*          ATRDAT                                                 *)
(*        CALLS:                                                   *)
(*          NONE                                                   *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*        OPERATION I  THE OPERATION TO BE PERFORMED               *)
(*                        VALID CHOICES:                           *)
(*                           'UPDATE'                              *)
(*                           'QUERY'                               *)
(*        KIND     I/O THE KIND NUMBER TO BE PUT IN OR KEYED ON    *)
(*        INDEX    I/O THE IDENT TO BE PUT IN OR KEYED ON          *)
(*        EFPTR    I/O THE EXCHANGE FORMAT POINTER                 *)
(*        RC       O   RETURN CODE                                 *)
(*                      0 : OK                                     *)
(*                      1 : KIND OR IDENT NOT FOUND (QUERY)        *)
(*                      2 : DUPLICATE KIND AND IDENT (UPDATE)      *)
(*                      3 : INVALID OPERATION                      *)
(*                                                                 *)
(*                                                                 *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*                                                                 *)
(*    CHANGE CONTROL:                                              *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN          K315 2G  CREATED: 86/10/29       *)
(*    VERSION: 1                             REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : FILEWF                                          *)
(*                                                                   *)
(*    FUNCTION : FILE THE WORKING FORM MODEL                         *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*         STEP                                                      *)
(*      CALLS:                                                       *)
(*         ENFILE                                                    *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*      IRC       O   RETURN CODE                                    *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE FILES THE WORKING FORM MODEL TO EITHER        *)
(*    THE NATIVE DATABASE OR THE PDDI SUPPLIED DATABASE              *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN                ORG_ID  CREATED: 86/05/14   *)
(*  VERSION:  1                                  REVISED:            *)
(*                                                                   *)
(*  ROUTINE NAME : FILLADB                                           *)
(*                                                                   *)
(*  FUNCTION : TO FILL VALUES IN THE ATTRIBUTE DATA BLOCK OF THE     *)
(*             CURRENT ENTITY                                        *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      CONTOK                                                       *)
(*      STORADB                                                      *)
(*    CALLED BY:                                                     *)
(*      FILL-VALUE-IN-ENTITY                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN             I   THE INPUT TOKEN                        *)
(*    ENTADB           I/O THE ARRAY OF ENTITY ADB'S               *)
(*    CLIST            I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S  *)
(*    WFDD             I/O THE ARRAY OF DATA DICTIONARIES          *)
(*    WFDD_POS         I/O THE POSITION IN THE CURRENT WFDD        *)
(*    CURR_LEVEL       I/O CURRENT LEVEL WITHIN ARRAYS             *)
(*    LEVEL_REC        I/O RECORD OF LEVEL INFORMATION             *)
(*    ENTITY_COMPLETE  I/O FLAG INDICATING THE END OF THE ENTITY   *)
(*    RC                O  RETURN CODE                             *)
(*                         0   : OK                                *)
(*                         >0  : FAILURE                           *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*  COMMENTS:                                                        *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE   *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION-  *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED*)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE     *)
(*  ALLOCATED.                                                       *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/14     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : FILLCNST                                          *)
(*                                                                   *)
(*  FUNCTION : TO FILL VALUES IN THE CONSTITUENT LIST OF THE         *)
(*             CURRENT ENTITY                                        *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      GETNEW                                                       *)
(*      ADDCNST                                                      *)
(*      INITIALIZE-LEVEL-RECORD                                      *)
(*      FILL-TOP-OF-ENTITY                                           *)
(*      CONTOK                                                       *)
(*      RETRIEVE-MAPPING-INFORMATION                                 *)
(*      GET-DATA-DICTIONARY-POSITION                                 *)
(*      FILL-VALUE-IN-SUBENTITY                                      *)
(*    CALLED BY:                                                     *)
(*      FILL-VALUE-IN-ENTITY                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN             I  THE INPUT TOKEN                           *)
(*    ENTADB          I/O THE ARRAY OF ENTITY ADB'S                  *)
(*    CLIST           I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S     *)
(*    WFDD            I/O THE ARRAY OF DATA DICTIONARIES             *)
(*    WFDD_POS        I/O THE POSITION IN THE CURRENT WFDD           *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS                *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION                *)
(*    ENTITY_COMPLETE I/O FLAG INDICATING THE END OF THE ENTITY      *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        >0  : FAILURE                              *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*  COMMENTS:                                                        *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE   *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION-  *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED *)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE     *)
(*  ALLOCATED.                                                       *)
(*                                                                   *)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/15     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : FILLSTRC                                          *)
(*                                                                   *)
(*  FUNCTION : TO FILL VALUES IN STRUCTURES                          *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      NONE                                                         *)
(*    CALLED BY:                                                     *)
(*      FILL-VALUE-IN-ENTITY                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN             I  THE INPUT TOKEN                           *)
(*    ENTADB          I/O THE ARRAY OF ENTITY ADB'S                  *)
(*    CLIST           I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S     *)
(*    WFDD            I/O THE ARRAY OF DATA DICTIONARIES             *)
(*    WFDD_POS        I/O THE POSITION IN THE CURRENT WFDD           *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS                *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION                *)
(*    ENTITY_COMPLETE I/O FLAG INDICATING THE END OF THE ENTITY      *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        >0  : FAILURE                              *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*  COMMENTS:                                                        *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE   *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION-  *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED*)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE     *)
(*  ALLOCATED.                                                       *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/14   *)
(*  VERSION:  1                                REVISED:            *)
(*                                                                 *)
(*  ROUTINE NAME : FILLSUB                                         *)
(*                                                                 *)
(*  FUNCTION : TO FILL VALUES IN THE CURRENT SUBENTITY             *)
(*                                                                 *)
(*  ENVIRONMENT:                                                   *)
(*    IBM PASCAL LANGUAGE                                          *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.     *)
(*                                                                 *)
(*  EXECUTION PROCEDURE:                                           *)
(*    CALLS:                                                       *)
(*      PERROR                                                     *)
(*      INITIALIZE-LEVEL-RECORD                                    *)
(*      FILL-TOP-OF-ENTITY                                         *)
(*      FILL-DATA-DICTIONARY                                       *)
(*      GET-DATA-DICTIONARY-POSITION                               *)
(*      ADD-CONSTITUENT                                            *)
(*    CALLED BY:                                                   *)
(*      FILL-VALUE-IN-ENTITY                                       *)
(*      FILL-VALUE-IN-CONSTITUENT-LIST                             *)
(*                                                                 *)
(*  DESCRIPTION OF ARGUMENTS:                                      *)
(*    TOKEN             I  THE INPUT TOKEN                         *)
(*    ENTADB          I/O THE ARRAY OF ENTITY ADB'S                *)
(*    CLIST           I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S   *)
(*    WFDD            I/O THE ARRAY OF DATA DICTIONARIES           *)
(*    WFDD_POS        I/O THE POSITION IN THE CURRENT WFDD         *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS              *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION              *)
(*    ENTITY_COMPLETE I/O FLAG INDICATING THE END OF THE ENTITY    *)
(*    RC               O  RETURN CODE                              *)
(*                        0   : OK                                 *)
(*                        >0  : FAILURE                            *)
(*                                                                 *)
(*  PROCESSING DESCRIPTION:                                        *)
(*  COMMENTS:                                                      *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION- *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED *)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE   *)
(*  ALLOCATED.                                                     *)
(*                                                                 *)
(*  CHANGE CONTROL:                                                *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/02/02     *)
(*    VERSION: 1                               REVISED:              *)
(*                                                                   *)
(*    ROUTINE NAME: FILLTOP                                          *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        PLACES VALUES IN ADB FOR THOSE AREAS COMMON TO ALL KINDS   *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLS:                                                       *)
(*        GETHI                                                      *)
(*        STORADB                                                    *)
(*      CALLED BY:                                                   *)
(*        PROCESS-TOKEN                                              *)
(*        FILL-VALUE-IN-CONSTITUENT-LIST                             *)
(*        FILL-VALUE-IN-SUBENTITY                                    *)
(*        POST                                                       *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND     I   TYPE OF WORKING FORM ENTITY                   *)
(*        ENTADB  I/O  BUILDING AREA FOR WORKING FORM ENTITY         *)
(*        EDBCOM  I/O  WORKING FORM DATA DICTIONARY                  *)
(*        RC       O   THE RETURN CODE                               *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        FILLS IN THE AREAS IN THE ADB COMMON TO ALL WORKING FORM   *)
(*        TYPES.  USES DATA DICTIONARIES TO DETERMINE HOW DATA SHOULD*)
(*        BE PLACED IN THE ADB.                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/09     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : FILLVAL                                           *)
(*                                                                   *)
(*  FUNCTION : TO FILL VALUES IN THE WORKING FORM                    *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      FILL-VALUE-IN-ADB                                            *)
(*      FILL-VALUE-IN-CONSTITUENT-LIST                               *)
(*      FILL-VALUE-IN-SUBENTITY                                      *)
(*      FILL-VALUE-IN-STRUCTURE                                      *)
(*      GET-DATA-DICTIONARY-POSITION                                 *)
(*    CALLED BY:                                                     *)
(*      PROCESS-TOKEN                                                *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN             I  THE INPUT TOKEN                           *)
(*    ENTADB          I/O THE ARRAY OF ENTITY ADB'S                  *)
(*    CLIST           I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S     *)
(*    WFDD            I/O THE ARRAY OF DATA DICTIONARIES             *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS                *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION                *)
(*    ENTITY_COMPLETE I/O FLAG INDICATING THE END OF THE ENTITY      *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        1   : FAILURE FROM LOWER ROUTINES          *)
(*                        2   : LEVELS DIDN'T MATCH UP               *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*  COMMENTS:                                                        *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE   *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION-  *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED*)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE     *)
(*  ALLOCATED.                                                       *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  K. CHI                  W315 2G CREATED: 84/10/19     *)
(*    VERSION:  1.0                           REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: FORINFO                                          *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        FORMATS THE USER PROVIDED INFORMATION FROM INTERFACE       *)
(*        INTO A RECORD FORM                                         *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          CONTROL                                                  *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*      NONE                                                         *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*        U_REC    I   CONTAINS ALL THE USER INFORMATION IN A RECORD *)
(*                     FORMAT                                        *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE WILL OPEN THE PASFIL FILE WHERE ALL THE USER  *)
(*        INFORMATION IS KEPT                                        *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        THIS ROUTINE WILL ONLY BE CALLED UPON ONCE THROUGHOUT      *)
(*        THE ENTIRE PROCESS                                         *)
(*        THAT IS ONLY IN THE BEGINNING                              *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*    AUTHOR: LAUREL A. CASSIN        ORG_ID  CREATED: 85/01/04  CC      *)
(*    VERSION: 1.0                            REVISED:           CC      *)
(*                                                                       *)
(*    ROUTINE NAME:  FTCHADB                                             *)
(*                                                                       *)
(*    FUNCTION: TO READ THE ATTRIBUTE DATA BLOCK AND GIVE THE            *)
(*    ACTUAL VALUE IN THE DATA FIELD.                                    *)
(*                                                                       *)
(*    ENVIRONMENT:                                                       *)
(*        IBM PASCAL LANGUAGE                                            *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                       *)
(*    EXECUTION PROCEDURE:                                               *)
(*        CALLED BY:                                                     *)
(*          PRTENT                                                       *)
(*          MAKDAT                                                       *)
(*          ATRDAT                                                       *)
(*          GETIND                                                       *)
(*          PRDATA                                                       *)
(*        CALLS:                                                         *)
(*          FTCHCHR                                                      *)
(*          FTCHINT                                                      *)
(*          FTCHLOG                                                      *)
(*          FTCHREL                                                      *)
(*          FTCHSCL                                                      *)
(*                                                                       *)
(*    DESCRIPTION OF ARGUMENTS:                                          *)
(*    INPUT :                                                            *)
(*       PASABD - THE ARRAY OF ADB INFO. IT IS SET EQUAL TO              *)
(*                EDBCOM IN THE CALLING ROUTINE.                         *)
(*                                                                       *)
(*       FTCHI  - A RECORD WITH INFORMATION THAT FCTHADB NEEDS.          *)
(*                  SUCH AS LENGTH,TYPE,DISP,MIN,MAX.                    *)
(*    OUTPUT :                                                           *)
(*       FTCHER - A RECORD WITH INFORMATION FROM FTCHADB, SUCH AS        *)
(*                 WHETHER THE DATA FIELD IS AN INTEGER,REAL,CHAR,       *)
(*                 LOGICAL,SCALAR,SET,POINTER,SUBENTITY,STRUCTURE.       *)
(*                                                                       *)
(*       CRC    - THE RETURN CODE THAT INDICATES THE CONDITION          *)
(*                OF THE ROUTINE.                                        *)
(*                                                                       *)
(*    PROCESSING DESCRIPTION:                                            *)
(*        THIS PROGRAM USES AMPXMOVE WHICH IS A SUB-SUBROUTINE           *)
(*        IT READS THE BYTES OF THE ATTRIBUTE DATA BLOCK.                *)
(*                                                                       *)
(*    COMMENTS:                                                          *)
(*        NO RETURN CODE IS SET                                          *)
(*                                                                       *)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/01/04     *)
(*    VERSION: 1                               REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHCHR                                         *)
(*                                                                   *)
(*    FUNCTION: OBTAIN A CHARACTER STRING FROM THE ADB               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          FTCHADB                                                  *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VAROUT   O   THE STRING MADE BY THE ROUTINE                *)
(*        PASADB   I   THE ADB TO GET THE STRING FROM                *)
(*        DDISP    I   DISPLACEMENT INTO ADB WHERE STRING STARTS     *)
(*        LENG     I   THE LENGTH OF THE STRING                      *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       MOVE A NUMBER OF BYTES FROM THE ADB TO A TEMPORARY ARRAY.   *)
(*       CREATE A STRING FROM THIS ARRAY.                            *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/01/11  CC   *)
(*    VERSION: 1                               REVISED:           CC   *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHHAF                                         *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        OBTAIN A TWO-BYTE INTEGER FROM THE ADB                     *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          FTCHINT                                                  *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VAROUT   O   HOLDS THE INTEGER                             *)
(*        PASADB   I   AREA TO GET THE INTEGER FROM  (ADB)           *)
(*        DDISP    I   DISPLACEMENT INTO THE ADB                     *)
(*        LENG     I   LENGTH OF THE INTEGER                         *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        PUT TWO BYTES OF THE ADB INTO THE LAST TWO BYTE OF A       *)
(*        "NORMAL" (FOUR-BYTE) INTEGER.                              *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/01/04     *)
(*    VERSION: 1                              REVISED:              *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHINT                                         *)
(*                                                                   *)
(*    FUNCTION: OBTAIN AN INTEGER FROM THE ADB                       *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*         IBM PASCAL LANGUAGE                                       *)
(*         IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.  *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        FTCHADB                                                    *)
(*      CALLS:                                                       *)
(*        FTCHSCL                                                    *)
(*        FTCHHAF                                                    *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       VAROUT    O   THE INTEGER FROM   THE ADB                   *)
(*       PASADB    I    AREA TO GET THE INTEGER FROM                *)
(*       DDISP     I    DISPLACEMENT INTO ADB                       *)
(*       LENG      I    LENGTH OF INTEGER                           *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*         IF A 1-BYTE INTEGER CALL FTCHSCL.  IF 2-BYTE CALL FTCHHAF *)
(*         ELSE MOVE BYTES FROM ADB TO TEMPORARY ARRAY.              *)
(*         EQUIVALENCE TO INTEGER.                                   *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/01/04     *)
(*    VERSION: 1                               REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHLOG                                         *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        OBTAIN A LOGICAL FROM THE ADB                              *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          FTCHADB                                                  *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VAROUT   O   VARIABLE TO BE RETURNED                       *)
(*        PASADB   I   AREA TO GET LOGICAL FROM                      *)
(*        DDISP    I   DISPLACEMENT INTO ADB                         *)
(*        LENG     I   LENGTH OF VARIABLE                            *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        PUT BYTES FROM ADB INTO TEMPORARY ARRAY.  EQUIVALENCE TO   *)
(*        A LOGICAL (BOOLEAN).                                       *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/01/04     *)
(*    VERSION: 1                               REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHREL                                         *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        OBTAIN AN 8-BYTE REAL NUMBER FROM THE ADB                  *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          FTCHADB                                                  *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VAROUT   O   THE VARIABLE TO BE OUTPUT                     *)
(*        PASADB   I   THE ADB TO GET THE VARIABLE FROM              *)
(*        DDISP    I   DISPLACEMENT INTO THE ADB                     *)
(*        LENG     I   THE LENGTH OF THE VARIABLE                    *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        PUT BYTES FROM ADB INTO TEMPORARY ARRAY.  EQUIVALENCE TO   *)
(*        A REAL.                                                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/12/14  CC   *)
(*    VERSION: 1                              REVISED:           CC   *)
(*                                                                   *)
(*    ROUTINE NAME : FTCHSCL                                         *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        OBTAIN A ONE-BYTE INTEGER FROM THE ADB.                    *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        FTCHADB                                                    *)
(*        FTCHINT                                                    *)
(*      CALLS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VAROUT   O   VARIABLE TO HOLD THE SCALAR                   *)
(*        PASADB   I   AREA TO GET THE INTEGER FROM (ADB)            *)
(*        DDISP    I   DISPLACEMENT INTO THE ADB                     *)
(*        LENG     I   LENGTH OF THE INTEGER                         *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        MOVE ONE BYTE OF ADB INTO ARRAY OVERLAYED BY INTEGER.      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/12     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETCOMM                                           *)
(*                                                                   *)
(*  FUNCTION : TO EXTRACT A COMMENT AND WRITE IT TO THE MESSAGE FILE *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      READEF                                                       *)
(*      WRITE-TO-MESSAGE-FILE                                        *)
(*    CALLED BY:                                                     *)
(*      GETTOKEN                                                     *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFFILE   I/O  THE EXCHANGE FORMAT FILE                         *)
(*    EFREC    I/O  THE CURRENT RECORD                               *)
(*    POS      I/O  THE CURRENT POSITION IN THE RECORD               *)
(*    ENDFIL    O   END OF EXCHANGE FILE FLAG                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0   : OK                                         *)
(*                  -1  : WRITE TO MESSAGE FILE FAILED               *)
(*                  1   : UNABLE TO READ EXCHANGE FORMAT RECORD      *)
(*                  2   : END OF FILE BEFORE LOGICAL END OF EXCHANGE *)
(*                        FORMAT                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/15     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETDDPOS                                          *)
(*                                                                   *)
(*  FUNCTION : TO GET THE POSITION IN THE CURRENT WFDD WHERE         *)
(*             THE CURRENT EFDD-POS IS EQUAL TO THE CSORDR           *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      NONE                                                         *)
(*    CALLED BY:                                                     *)
(*      FILL-VALUE-IN-ENTITY                                         *)
(*      FILL-VALUE-IN-CONSTITUENT-LIST                               *)
(*      FILL-VALUE-IN-SUBENTITY                                      *)
(*      PROCESS-DATA                                                 *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    LEVEL_REC        I/O RECORD OF LEVEL INFORMATION               *)
(*    CURR_LEVEL       I/O CURRENT LEVEL WITHIN ARRAYS               *)
(*    WFDD             I/O THE ARRAY OF DATA DICTIONARIES            *)
(*    WFDD_POS          O  THE POSITION IN THE WFDD                  *)
(*    RC                O  RETURN CODE                               *)
(*                         0   : OK                                  *)
(*                         1   : POSITION NOT FOUND                  *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*  COMMENTS:                                                        *)
(*    THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE SPACE   *)
(*  FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA DICTION-  *)
(*  ARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL BE PRINTED*)
(*  OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, LEAVING THIS *)
(*  ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE WILL HAVE TO BE     *)
(*  ALLOCATED.                                                       *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 3G CREATED: 84/12/12     *)
(*    VERSION: 1.0                             REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME : GETHI                                           *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          FILLTOP                                                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND     I   THE KIND TO FIND HIGH INSTANCE OF             *)
(*        FREEID   O   THE HIGHEST INSTANCE                          *)
(*        IRC      O   THE RETURN CODE                               *)
(*                     0 : OK                                        *)
(*                     2 : A MAS ROUTINE FAILED                      *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       MAKE A LIST OF ALL KEYS OF THIS KIND. RUN THROUGH LIST,     *)
(*       SAVING HIGHEST INSTANCE.                                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR: A. M. WHELAN                ORG_ID  CREATED: 85/12/04    *)
(*  VERSION:  1                                 REVISED:             *)
(*                                                                   *)
(*  ROUTINE NAME : GETIND                                            *)
(*                                                                   *)
(*  FUNCTION : GET IDENT OF ENTITY                                   *)
(*                                                                   *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    FTCHADB                                                        *)
(*  CALLED BY:                                                       *)
(*    ATRDAT                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    PASADB    I   THE ATTRIBUTE DATA BLOCK OF THE ENTITY           *)
(*    INDEX     O   THE IDENT OF THE ENTITY                          *)
(*    RC        O   RETURN CODE                                      *)
(*                  0  : OK                                          *)
(*                  >0 : FAILURE                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*    GETS THE IDENT OF THE ENTITY                                   *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            K315 2G CREATED: 86/05/15      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: GETKND                                           *)
(*                                                                   *)
(*    FUNCTION :                                                     *)
(*        GET THE KIND CORRESPONDING TO THE STRING NAME              *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*        CALLED BY:                                                 *)
(*          PROCESS-TOKEN                                            *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        STR_NAME I   THE STRING NAME CORRESPONDING TO THE KIND     *)
(*        KNDTBL  I/O  TABLE OF KINDS AND CORRESPONDING NAMES        *)
(*        NUMKND  I/O  NUMBER OF ENTRIES IN THE KNDTBL               *)
(*        KIND     O   THE KIND VALUE                                *)
(*        RC       O   RETURN CODE                                   *)
(*                      0 : OK                                       *)
(*                      2 : KIND VALUE NOT VALID                     *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       SEARCH KNDTBL FOR KIND AND ASSOCIATED ENTITY NAME.          *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:   PHIL DORR                ORG_ID  CREATED:  07/02/86    *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETMTOK                                           *)
(*                                                                   *)
(*  FUNCTION :  TO RETURN THE NEXT MEANINGFUL TOKEN OR A NULL        *)
(*    STRING IF THERE IS A DEFAULTED VALUE.                          *)
(*                                                                   *)
(*    ORIGINALLY THIS FUNCTION WAS PERFORMED BY GETTOKEN.            *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    GETTOKEN                                                       *)
(*  CALLED BY:                                                       *)
(*    POST                                                           *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFFILE   I/O  THE EXCHANGE FORMAT FILE                         *)
(*    EFREC    I/O  THE CURRENT EXCHANGE FORMAT RECORD               *)
(*    POS      I/O  THE CURRENT POSITION IN THE RECORD               *)
(*    TOKEN     O   THE OUTPUT TOKEN OR NULL FOR DEFAULTED FIELDS    *)
(*    TOKENO   I/O  THE LAST TOKEN (CALLING PROGRAM SHOULD NOT ALTER)*)
(*    SNDAGN   I/O  SEND LAST TOKEN AFTER NULL FIELD IS PROCESSED    *)
(*    EOF       O   END OF EXCHANGE FILE FLAG                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0   : OK                                         *)
(*                  -1  : WARNING : NO SEMI-COLON AFTER ENDSEC       *)
(*                        KEYWORD                                    *)
(*                  1   : ERROR : END OF FILE BEFORE LOGICAL END OF  *)
(*                        EXCHANGE FORMAT                            *)
(*                  2   : ERROR : BAD TOKEN                          *)
(*     *** NOTE ***                                                  *)
(*       THESE RETURNS CODES ARE GENERATED IN GETTOKEN AND PASSED    *)
(*       TO GET-MEANINGFUL-TOKEN (GETMTOK) AND THEN PASSED UNALTERD  *)
(*       TO POST.  GETMTOK DOES NOT DETECT OR REPORT ANY OTHER       *)
(*       ERRORS.                                                     *)
(*     *** NOTE ***                                                  *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*     THIS PROCEDURE WILL CALL GETTOKEN EITHER ONCE OR TWICE TO     *)
(*     GET THE NEXT MEANINGFUL TOKEN AND DETECTING NULL FIELDS.      *)
(*     A NULL STRING IS RETURNED IF A NULL FIELD IS DETECTED TO      *)
(*     INDICATE THAT A VALUE HAS BEEN DEFAULTED.  ALL TOKENS,        *)
(*     INCLUDING DELIMITERS ARE PASSED UP FROM GETTOKEN.             *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 84/12/19     *)
(*    VERSION:1.0                             REVISED:               *)
(*                                                                   *)
(*    ROUTINE : GETNEW                                               *)
(*        FROM JOHN PURSES PGM                                       *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       READS THE DATA DICTIONARY OF A SPECIFIED PDD ENTITY FROM    *)
(*       THE DATA SET "CAD5.GMAP.DEFN0801.DATA" AND INSERTS THE      *)
(*       DATA INTO AN AREA REQUESTED                                 *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       CALLED BY:                                                  *)
(*         TRMPRT                                                    *)
(*         MAKDAT                                                    *)
(*         ATRDAT                                                    *)
(*         POST                                                      *)
(*         PRTOKEN                                                   *)
(*         FILLCNST                                                  *)
(*         FILLSUB                                                   *)
(*      CALLS:                                                       *)
(*        INITNEW                                                    *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND     I   THE KIND NUMBER OF THE ENTITY DEFINITION      *)
(*                     TO BE READ                                    *)
(*        DATDIC   O   THE ARRAY FILLED WITH INFO                    *)
(*        RC       O   THE RETURN CODE                               *)
(*                     0  : OK                                       *)
(*                     >0 : FAILURE                                  *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*      THIS PROGRAM OPENS 'CAD5.GMAP.DEFN0801.DATA' AND READS THE   *)
(*      DATA FROM IT FOR THE KIND.  EACH MEMBER IN THE INPDD STREAM  *)
(*      LOOKS LIKE:                                                  *)
(*                                                                   *)
(*      FIRST RECORD:                                                *)
(*      ABBBBBBBBBBBBBBB,CCCCC,DD,E                                  *)
(*                                                                   *)
(*      A:  CONTINUATION FLAG FIELD (NOT USED ON FIRST RECORD)       *)
(*      B:  ENTITY NAME (16 CHARACTERS)                              *)
(*      C:  KIND NUMBER                                              *)
(*      D:  NUMBER OF ATTRIBUTES                                     *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/12     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETQUOT                                           *)
(*                                                                   *)
(*  FUNCTION : TO EXTRACT A CHARACTER STRING FROM THE EXCHANGE FILE  *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      READEF                                                       *)
(*      PERROR                                                       *)
(*    CALLED BY:                                                     *)
(*      GETTOKEN                                                     *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFFILE    I   THE EXCHANGE FORMAT FILE                         *)
(*    EFREC    I/O  THE CURRENT RECORD                               *)
(*    POS      I/O  THE CURRENT POSITION IN THE RECORD               *)
(*    ENDFIL    O   END OF EXCHANGE FILE FLAG                        *)
(*    TOKEN     O   THE OUTPUT CHARACTER STRING                      *)
(*    RC        O   RETURN CODE                                      *)
(*                  0   : OK                                         *)
(*                  1   : UNABLE TO READ EXCHANGE FORMAT RECORD      *)
(*                  2   : END OF FILE BEFORE LOGICAL END OF EXCHANGE *)
(*                        FORMAT                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/15     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETSCRIN                                          *)
(*                                                                   *)
(*  FUNCTION : TO GET THE POSITION IN THE WFDD WHERE WEHERE          *)
(*             IS EQUAL TO THE CSORDR                                *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      NONE                                                         *)
(*    CALLED BY:                                                     *)
(*      ATRDAT                                                       *)
(*      NILCON                                                       *)
(*      DOTOP                                                        *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    WFDD             I/O THE WORKING FORM DATA DICTIONARY         *)
(*    WEHERE           I/O THE EFDD POS                             *)
(*    WFDD_POS          O  THE POSITION IN THE WFDD                 *)
(*    RC                O  RETURN CODE                              *)
(*                         0   : OK                                  *)
(*                         1   : POSITION NOT FOUND                  *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:   PHIL DORR                ORG_ID  CREATED:  5/29/86     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETSEMI                                           *)
(*                                                                   *)
(*  FUNCTION : SET THE POSITION TO AFTER THE NEXT SEMI-COLON -       *)
(*             THE BEGINNING OF THE NEXT ENTITY.                     *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    READEF                                                         *)
(*    PERROR                                                         *)
(*  CALLED BY:                                                       *)
(*    POST                                                           *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFFILE   I/O  THE EXCHANGE FORMAT FILE                         *)
(*    EFREC    I/O  THE CURRENT EXCHANGE FORMAT RECORD               *)
(*    POS      I/O  THE CURRENT POSITION IN THE RECORD               *)
(*    ENDFIL   I/O  A FLAG INDICATING THE END OF FILE                *)
(*    RC        O   RETURN CODE                                      *)
(*                  0  : OK                                          *)
(*                  >0 : FAILURE                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*     THIS PROCEDURE WILL SET THE VARIBLE POS EQUAL TO THE          *)
(*     POSITION AFTER THE NEXT SEMICOLON IN THE FILE,                *)
(*     READING NEW RECORDS IF IT HAS TO.                             *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/12     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : GETTOKEN                                          *)
(*                                                                   *)
(*  FUNCTION : TO EXTRACT A TOKEN FROM THE EXCHANGE FILE             *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      READEF                                                       *)
(*      GET-COMMENT                                                  *)
(*      GET-QUOTE                                                    *)
(*    CALLED BY:                                                     *)
(*      GETMTOK  (GET MEANINGFUL TOKEN)                              *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFFILE   I/O  THE EXCHANGE FORMAT FILE                         *)
(*    EFREC    I/O  THE CURRENT EXCHANGE FORMAT RECORD               *)
(*    POS      I/O  THE CURRENT POSITION IN THE RECORD               *)
(*    TOKEN     O   THE OUTPUT TOKEN                                 *)
(*    EOF       O   END OF EXCHANGE FILE FLAG                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0   : OK                                         *)
(*                  -1  : WARNING : NO SEMI-COLON AFTER ENDSEC       *)
(*                        KEYWORD                                    *)
(*                  1   : ERROR : END OF FILE BEFORE LOGICAL END OF  *)
(*                        EXCHANGE FORMAT                            *)
(*                  2   : ERROR : BAD TOKEN                          *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 84/12/20  CC   *)
(*    VERSION: 1.0                             REVISED: 85/09/17  CC   *)
(*                                                                   *)
(*    ROUTINE NAME: GETUNI                                           *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*        GETS STARTING LINE IN DATA DICTIONARY                      *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          ATRDAT                                                   *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        EDBCOM  I/O  WORKING FORM DATADICTIONARY FOR KIND          *)
(*        UNIDEX   O   STARTING LINE ID DD                           *)
(*        WEDONE   O   FLAG INDICATING IF DONE WITH ENTITY OR NOT    *)
(*        WEHERE   O   WHERE WE ARE IN THE EXCHANGE FORMAT           *)
(*        LEVEL    O   WHAT LEVEL OF STRUCTURES WE ARE IN            *)
(*        BNAME    O   STRUCTURE NAMES                               *)
(*        INSTRC   O   FLAG INDICATING IF IN A STRUCTURE             *)
(*        RC       O   RETURN CODE                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        USING THE DATA DICTIONARY, FIND THE STARTING LINE          *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN          K315 2G CREATED: 85/12/30        *)
(*    VERSION: 1 ( JAN. DEMO )             REVISED:                 *)
(*                                                                   *)
(*  ROUTINE NAME : GETWKF                                            *)
(*                                                                   *)
(*      FUNCTION : GET WORKING FORM ENTITY INFORMATION               *)
(*                 GET THE ATTRIBUTES AND CONSTITUENTS OF THE        *)
(*                 WORKING FORM ENTITY TO BE MAPPED.                 *)
(*                 RETURN THE KIND VALUE.                            *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        THIS ROUTINE IS CALLED BY PRE TO GET THE WORKING FORM      *)
(*        INFORMATION.  IT CALLS MAEGTK TO GET THE ADB AND MAEC TO   *)
(*        GET THE CONSTITUENT LIST KEY.                              *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        WFKEY    I   THE MAS KEY TO THE ENTITY                     *)
(*        KIND     O   THE KIND OF THE ENTITY                        *)
(*        ENTADB   O   THE ATTRIBUTE DATA BLOCK OF THE ENTITY        *)
(*        CONLIS   O   THE CONSTITUENT LIST OF THE ENTITY            *)
(*        IRC      O   THE RETURN CODE                               *)
(*                      0 : GOOD                                     *)
(*                      2 : MISSING ADB OR CONSTITUENT LIST          *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE OBTAINS THE ADB AND THE KEY TO THE            *)
(*        CONSTITUENT LIST GIVEN THE MAS KEY.                        *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/14     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : INITLR                                            *)
(*                                                                   *)
(*  FUNCTION : INITIALIZE THE LEVEL RECORD                           *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    PROCESS-TOKEN                                                  *)
(*    FILL-VALUE-IN-CONSTITUENT-LIST                                 *)
(*    FILL-VALUE-IN-SUBENTITY                                        *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION                *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS                *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        >0  : FAILURE                              *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN                ORG_ID  CREATED: 86/05/27   *)
(*  VERSION:  1                                  REVISED:            *)
(*                                                                   *)
(*  ROUTINE NAME : INITMAP                                           *)
(*                                                                   *)
(*  FUNCTION : INITIALIZE THE MAPPING ARRAY                          *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    POST                                                           *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    NONE                                                           *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS            W315 2G CREATED: 85/01/07  CC   *)
(*    VERSION:1.0                            REVISED:           CC   *)
(*                                                                   *)
(*    ROUTINE NAME : INITNEW                                         *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       INITIALIZE DATDIC TO BLANKS                                 *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        GETNEW                                                     *)
(*      CALLS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        DATDIC  I/O  AREA TO BE INITIALIZED                        *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS PROGRAM FILLS DATDIC ARRAY WITH BLANKS                *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            W315 2G CREATED: 86/03/17      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : KNDLIS                                          *)
(*                                                                   *)
(*    FUNCTION : READ A LIST OF KINDS AND ASSOCIATED ENTITY          *)
(*               NAMES.                                              *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          PRE                                                      *)
(*          POST                                                     *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KNDTBL   O   TABLE OF KINDS AND THEIR ASSOCIATED NAMES     *)
(*        NUMKND   O   NUMBER OF ENTRIES IN THE KNDTBL               *)
(*        IRC      O   RETURN CODE                                   *)
(*                       0 : OK                                      *)
(*                      >0 : FAILURE                                 *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE READS IN THE KIND TABLE FROM THE FILE KNDFIL  *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: J. M. FRANKLIN          K315 2G  CREATED: 84/11/12  CC *)
(*    VERSION: 1 ( DEC. DEMO )                 REVISED: 85/01/09  CC *)
(*                                                                   *)
(*    ROUTINE NAME : MAERRM                                          *)
(*                                                                   *)
(*    FUNCTION : PRODUCE AN ERROR MESSAGE FROM AN INPUT ERROR NUMBER *)
(*                                                                   *)
(*    ENVIRONMENT: IBM PASCAL                                        *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        THIS ROUTINE IS CALLED BY ANY ROUTINE THAT CALLS MAS       *)
(*        ROUTINES.                                                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        IERR     I   MAS ERROR NUMBER                              *)
(*        ERRM     O   ERROR MESSAGE                                 *)
(*        IRC      O   RETURN CODE                                   *)
(*                       0 = OK                                      *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THE MAS ERROR NUMBER IS USED TO GET AN ERROR MESSAGE.  EACH*)
(*        NUMBER HAS A UNIQUE MESSAGE.                               *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        NONE                                                       *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: LAUREL A. CASSIN        W315 2G CREATED: 84/12/17  CC  *)
(*    VERSION: 1.0                            REVISED: 85/09/17  CC  *)
(*                                                                   *)
(*    ROUTINE NAME:  MAKDAT                                          *)
(*                                                                   *)
(*    FUNCTION: TO MAKE AN EXCHANGE FORMAT ENTITY WITH THE DATA      *)
(*              SECTION OF THE WORKING FORM DATA                     *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        ENCODE                                                     *)
(*      CALLS:                                                       *)
(*        PERROR                                                     *)
(*        FTCHADB                                                    *)
(*        FILEPTR                                                    *)
(*        STRNAM                                                     *)
(*        GETNEW                                                     *)
(*        ATRDAT                                                     *)
(*        WRTREC                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND             I    THE KIND TO LOOK FOR IN THE ADB     *)
(*        INDEX            I    THE INSTANCE OF THE KIND            *)
(*        WFKEY            I    THE KEY OF THE INSTANCE             *)
(*        KNDTBL           I    A TABLE OF KINDS AND THEIR NAMES    *)
(*        NUMKND           I    THE NUMBER OF ENTRIES IN KNDTBL     *)
(*        CURRENT_NUMBER   I    THE CURRENT EXCHANGE FORMAT IDENTIFIER *)
(*        CONLIS           I    THE INSTANCE OF THE KIND            *)
(*        EFFILE          I/O   THE EXCHANGE FORMAT FILE            *)
(*        MAPFIL          I/O   THE MAPPING FILE                    *)
(*        IRC              O    THE RETURN CODE                     *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        09/17/85 - L.A.DAVIS USE NEW GLOBAL EF DATADICTIONARY      *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS              W315 2G CREATED: 85/05/24     *)
(*    VERSION: 1                              REVISED:              *)
(*                                                                   *)
(*    ROUTINE NAME: NILCON                                           *)
(*                                                                   *)
(*    FUNCTION: NIL CONSTITUENT -                                    *)
(*        WRITE THE APPROPRIATE PUNCTUATION TO THE DATA SECTION      *)
(*        RECORD GIVEN A NIL OR DEFAULTED ENTITY POINTER.            *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        ATRDAT                                                     *)
(*      CALLS:                                                       *)
(*        GETSCRIN                                                   *)
(*        WRTREC                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        EDBCOM  I/O  WORKING FORM DATA DICTIONARY                  *)
(*        WEHERE  I/O  POINTER TO CURRENT LINE OF DATADICTIONARY     *)
(*        LEVEL    I   NUMBER OF EMBEDDEDNESS INSIDE STRUCTURES      *)
(*        INSTRC   I   POINTER INSIDE STRUCTURE FLAG                 *)
(*        WEDONE  I/O  END OF ENTITY FLAG                            *)
(*        EFFILE  I/O  EXCHANGE FORMAT FILE                          *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETERMINE WHETHER THIS IS THE LAST CONSTITUENT OF THE      *)
(*        ENTITY.  IF NOT, WRITE A COMMA TO THE DATA RECORD.         *)
(*        UPDATE THE DATADICTIONARY POINTER.                         *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: LAUREL A. CASSIN        W315 2G CREATED: 84/12/18      *)
(*    VERSION: 1.0                            REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME:  OPERAT                                          *)
(*                                                                   *)
(*    FUNCTION :  TO TAKE CARE OF CONVERTING THE VALUE BY CALLING    *)
(*                THE APPROPRIATE ROUTINES AND LOOKING AT THE        *)
(*                RETURN CODES.                                      *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        ATRDAT                                                     *)
(*      CALLS:                                                       *)
(*        PERROR                                                     *)
(*        WRTREC                                                     *)
(*        TABRED                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*   INPUT:                                                          *)
(*    FTCHER -   A PIECE OF THE ENTITY'S ATTRIBUTE DATA BLOCK        *)
(*               OR CONSTITUENT LIST THAT NEEDS PUNCTUATION IN       *)
(*               A RECORD BY VARIABLE TYPE.                          *)
(*    FTCHI  -   A PIECE OF THE ENTITY'S ATTRIBUTE DATA BLOCK        *)
(*               OR CONSTITUENT LIST THAT NEEDS PUNCTUATION IN       *)
(*               A RECORD BY VARIABLE TYPE.                          *)
(*    SFLAG  -   A FLAG FOR A STRUCTURE THAT SIGNALS 1:START,        *)
(*               2: END STRUCTURE AND 3:NONE.                        *)
(*    FFLAG  -   A FLAG FOR A FIELD THAT SIGNALS 1:LAST,             *)
(*               2: NEITHER AND 3:ONLY.                              *)
(*    AFLAGS  -  A FLAG FOR AN ARRAY THAT SIGNALS 1:START,           *)
(*               2: LAST AND 3:NEITHER AND 4:BOTH--FOR BOTH          *)
(*               DIMENSIONS                                          *)
(*    EFFILE -   THE DATA FILE TO FILL WITH RECORDS OF VARSTR.       *)
(*                                                                   *)
(*   OUTPUT:                                                         *)
(*    RC     -   THE RETURN CODE THAT INDICATES THE CONDITION        *)
(*               OF THE ROUTINE.                                     *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        EFFILE IS OPENED FOR THE CHARACTER DATA TO BE PUT INTO IT.  *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            W315 2G CREATED: 86/05/21      *)
(*    VERSION: 1.0                            REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: PERROR                                           *)
(*                                                                   *)
(*    FUNCTION:  COMMUNICATE WITH THE USER                           *)
(*        WRITES MESSAGES OUT TO THE USER OR TO A BATCH FILE         *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        THIS ROUTINE CAN BE CALLED FROM ANY ROUTINE AND WILL       *)
(*        RETURN BACK TO THE CALLER ROUTINE                          *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        STR_MSG  I   MESSAGE IDENTIFIER                            *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*        STA      I   INDICATES WHETHER THE MESSAGE TABLE NEEDS     *)
(*                     TO BE INITIALIZED                             *)
(*                                                                   *)
(*        M_TAB    I   ARRAY OF MESSAGES                             *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE WILL OPEN AND CLOSE THE FILE - MGTAB          *)
(*        MGTAB CONTAINS THE MESSAGE TABLE DATA WHICH NEEDS TO       *)
(*        BE INITIALIZED FIRST TIME IN THE PROGRAM                   *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN                ORG_ID  CREATED: 86/05/09   *)
(*  VERSION:  1                                  REVISED:            *)
(*                                                                   *)
(*  ROUTINE NAME : POST                                              *)
(*                                                                   *)
(*  FUNCTION : TO POSTPROCESS A PDES EXCHANGE FILE                   *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      KNDLIS                                                       *)
(*      GETSEMI                                                      *)
(*      INITMAP                                                      *)
(*      GET-MEANINGFUL-TOKEN (GETMTOK)                               *)
(*      PROCESS-HEADER-TOKEN                                         *)
(*      PROCESS-DECLARATION-TOKEN                                    *)
(*      PROCESS-DATA-TOKEN                                           *)
(*      GETNEW                                                       *)
(*      FILLTOP                                                      *)
(*                                                                   *)
(*    CALLED BY:                                                     *)
(*      STEP                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    RC        O   RETURN CODE                                      *)
(*                  0   : OK                                         *)
(*                  >0  : FAILURE                                    *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*------------------------------------------------------------ -----*)
(*                                                                  *)
(*  AUTHOR:  A. M. WHELAN                ORG_ID  CREATED: 86/05/12  *)
(*  VERSION:  1                                  REVISED:           *)
(*                                                                  *)
(*  ROUTINE NAME : PRDATA                                           *)
(*                                                                  *)
(*  FUNCTION:                                                       *)
(*    PUT DATA INFORMATION FROM EXCHANGE FILE INTO THE WORKING FORM *)
(*                                                                  *)
(*  ENVIRONMENT:                                                    *)
(*    IBM PASCAL LANGUAGE                                           *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.      *)
(*                                                                  *)
(*  EXECUTION PROCEDURE:                                            *)
(*    CALLS:                                                        *)
(*      PERROR                                                      *)
(*      PROCESS-TOKEN                                               *)
(*      FTCHADB                                                     *)
(*      ADDCNST                                                     *)
(*      GETDDPOS                                                    *)
(*      STORE-MAPPING-INFORMATION                                   *)
(*    CALLED BY:                                                    *)
(*      POST                                                        *)
(*                                                                  *)
(*  DESCRIPTION OF ARGUMENTS:                                       *)
(*    TOKEN             I    THE EXCHANGE FORMAT TOKEN              *)
(*    MAPFIL           I/O   THE MAPPING FILE                       *)
(*    ENTADB           I/O   THE ARRAY OF ENTITY ADB'S              *)
(*    CLIST            I/O   THE ARRAY OF ENTITY CONSTITUENT LIST'S *)
(*    WFDD             I/O   THE ARRAY OF DATA DICTIONARIES         *)
(*    CURR_LEVEL       I/O   CURRENT LEVEL WITHIN ARRAYS            *)
(*    FIRST_TOKEN      I/O   FLAG INDICATING BEGINNING OF ENTITY    *)
(*    SECOND_TOKEN     I/O   FLAG INDICATING SECOND TOKEN IN ENTITY *)
(*    LEVEL_REC        I/O   RECORD OF LEVEL INFORMATION            *)
(*    ENTITY_COMPLETE  I/O   FLAG INDICATING END OF ENTITY          *)
(*    EFPTR            I/O   EXCHANGE FORMAT IDENTIFIER             *)
(*    KNDTBL           I/O   TABLE OF KINDS AND THE ASSOCIATED NAMES*)
(*    NUMKND           I/O   NUMBER OF ENTRIES IN THE KNDTBL        *)
(*    RC                0    RETURN CODE                           *)
(*                           0   :  OK                              *)
(*                           >0  :  FAILURE                         *)
(*                                                                  *)
(*  PROCESSING DESCRIPTION:                                         *)
(*                                                                  *)
(*  CHANGE CONTROL:                                                 *)
(*                                                                  *)
(*------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/12     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : PRDECL                                            *)
(*                                                                   *)
(*  FUNCTION:                                                        *)
(*    TO PROCESS DECLARATION INFORMATION FROM THE EXCHANGE FILE      *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERRORSG                                                     *)
(*      WRITEMSG                                                     *)
(*    CALLED BY:                                                     *)
(*      POST                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN      I   THE EXCHANGE FORMAT TOKEN                       *)
(*    RC         O   RETURN CODE                                     *)
(*                   0    : OK                                      *)
(*                   >0   : FAILURE                                 *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            ORG_ID  CREATED: 86/05/10      *)
(*    VERSION: 1 ( JAN. DEMO )                REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : PRE                                             *)
(*                                                                   *)
(*    FUNCTION : PREPROCESS PDD                                      *)
(*               INFORMATION GIVEN ABOUT THE PDD, WORKING FORM       *)
(*               ENTITIES, AND THE METADATA RULES ARE USED TO CREATE *)
(*               THE PDD.  IT IS THEN PLACED IN THE APPROPRIATE      *)
(*               FILE.                                               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        STEP                                                       *)
(*      CALLS:                                                       *)
(*        PERROR                                                     *)
(*        KNDLIS                                                     *)
(*        WRTREC                                                     *)
(*        ENCODE                                                     *)
(*        GETWKF                                                     *)
(*        CRHEAD                                                     *)
(*        CRDECL                                                     *)
(*        CRRULE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*      IRC        O   RETURN CODE                                   *)
(*                       0  : OK                                     *)
(*                       >0 : FAILURE                                *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THE HEADER, DECLARATION, AND RULES SECTIONS ARE CREATED.   *)
(*        A LIST OF ENTITIES AND THEIR CORRESPONDING ENTITY NAMES    *)
(*        IS READ IN AND THAT LIST IS USED TO PLACE THE              *)
(*        ENTITIES IN EXCHANGE FORMAT RECORDS.                       *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/12     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : PRHEAD                                            *)
(*                                                                   *)
(*  FUNCTION : TO EXTRACT HEADER INFORMATION FROM THE EXCHANGE FILE  *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLS:                                                         *)
(*      PERROR                                                       *)
(*      WRITEMSG                                                     *)
(*    CALLED BY:                                                     *)
(*      POST                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN     I   THE EXCHANGE FORMAT TOKEN                        *)
(*    RC        O   RETURN CODE                                      *)
(*                    0    : OK                                      *)
(*                    >0   : FAILURE                                 *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR: J.M. PURSES              W315 2G CREATED: 6/24/85     *)
(*    VERSION:  1.0                            REVISED:            *)
(*                                                                 *)
(*    ROUTINE NAME: PRTENT                                         *)
(*                                                                 *)
(*    FUNCTION:                                                    *)
(*      PRINTS THE DEFINITIONS OF ALL THE ENTITIES IN THE INPUT    *)
(*      LIST.                                                      *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*        IBM PASCAL LANGUAGE                                      *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W. *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*        THIS ROUTINE WILL BE CALLED BY TRMPRT                    *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*        OUTFIL  O FILE TO WHICH THE ENTITIES ARE TO BE WRITTEN   *)
(*        KIND    I THE KIND VALUE OF THE ENTITY TYPE TO BE WRITTEN*)
(*        LSTKY   I THE LIST OF ALL ENTITIES OF A CERTAIN KIND     *)
(*        NUMATT2 I NUMBER OF ATTRIBUTES FOR THE ENTITY            *)
(*        EDBCOM  I WORKING FORM DATA DICTIONARY FOR THIS ENTITY   *)
(*        RETCOD  O RETURN CODE                                    *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*        THIS ROUTINE NEEDS NO OPEN/CLOSE FILES                   *)
(*                                                                 *)
(*    CHANGE CONTROL:                                              *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN                ORG_ID  CREATED: 86/05/12   *)
(*  VERSION:  1                                  REVISED:            *)
(*                                                                   *)
(*  ROUTINE NAME : PRTOKEN                                           *)
(*                                                                   *)
(*  FUNCTION : PROCESSES THE INPUT TOKEN.  IF IT IS A VALUE, THEN    *)
(*             IT IS CONVERTED TO THE CORRECT TYPE AND PUT IN THE    *)
(*             WORKING FORM                                          *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    PERROR                                                         *)
(*    FILL-VALUE-IN-ENTITY                                           *)
(*    FILL-DATA-DICTIONARY                                           *)
(*    GET-ENTITY-KIND                                                *)
(*    FILL-TOP-OF-ENTTY                                              *)
(*    INITIALIZE-LEVEL-RECORD                                        *)
(*  CALLED BY:                                                       *)
(*    PROCESS-DATA-TOKEN                                             *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN             I  THE TOKEN TO BE PROCESSED                 *)
(*    ENTADB          I/O THE ARRAY OF ENTITY ADB'S                  *)
(*    CLIST           I/O THE ARRAY OF ENTITY CONSTITUENT LIST'S     *)
(*    WFDD            I/O THE ARRAY OF DATA DICTIONARIES             *)
(*    CURR_LEVEL      I/O CURRENT LEVEL WITHIN ARRAYS                *)
(*    FIRST_TOKEN     I/O FLAG INDICATING BEGINNING OF ENTITY        *)
(*    SECOND_TOKEN    I/O FLAG INDICATING SECOND TOKEN IN ENTITY     *)
(*    LEVEL_REC       I/O RECORD OF LEVEL INFORMATION                *)
(*    ENTITY_COMPLETE I/O FLAG INDICATING THE END OF THE ENTITY      *)
(*    EFPTR           I/O EXCHANGE FORMAT POINTER                    *)
(*    KNDTBL          I/O THE LIST OF KINDS AND THE ASSOCIATED NAMES *)
(*    NUMKND          I/O NUMBER OF ENTRIES IN THE KNDTBL            *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        >0  : FAILURE                              *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*    COMMENTS:                                                      *)
(*      THIS ROUTINE WILL MAINTAIN UP TO 5 'LAYERS' OF STORAGE       *)
(*    SPACE FOR ATTRIBUTE DATA BLOCKS, CONSTITUENT LISTS, AND DATA   *)
(*    DICTIONARIES.  IF THAT LIMIT IS EXCEEDED, THEN A MESSAGE WILL  *)
(*    BE PRINTED OUT, AND THE PROGRAM WILL GO ON TO THE NEXT ENTITY, *)
(*    LEAVING THIS ONE NOT TRANSLATED.  TO FIX THIS, MORE SPACE      *)
```

```
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR: F. E. DISNEY          W315 2G  CREATED: 85/01/05     *)
(*    VERSION: 1                             REVISED:             *)
(*                                                                 *)
(*    ROUTINE NAME : PUNTAB                                        *)
(*                                                                 *)
(*    FUNCTION : GENERATE THE DELIMITERS FOR FIELDS.               *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*        IBM PASCAL LANGUAGE; PROCEDURE                           *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W. *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*      CALLED BY:                                                 *)
(*        TABRED                                                   *)
(*      CALLS:                                                     *)
(*        NONE                                                     *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*        SFLAG    I   STRUCTURE FLAG (1=START,2=END,3=NEITHER)    *)
(*        FFLAG    I   FIELDS FLAG (1=LAST,2=NOT LAST,3=ONLY)      *)
(*        AFLAGS   I   ARRAY FLAGS (1=FIRST,2=LAST,3=NEITHER,4=BOTH, *)
(*                                  5=NOT AN ARRAY)                *)
(*        PUNC1    O   DELIMITERS THAT GO BEFORE THE FIELD         *)
(*        PUNC2    O   DELIMITERS THAT GO AFTER THE FIELD          *)
(*        PRC      O   RETURN CODE                                 *)
(*                      0 = OK                                     *)
(*                      1 = DELIMITERS COULD NOT BE PRODUCED       *)
(*                          DUE TO AN INVALID COMBINATION OF FLAGS *)
(*                                                                 *)
(*    COMMONS:                                                     *)
(*      NONE                                                       *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*        THIS ROUTINE IS GIVEN FLAGS THAT DEFINE THE CONTEXT OF A *)
(*        FIELD.  THE PRECEDING AND FOLLOWING DELIMITERS ARE PRODUCED *)
(*        FROM THE DIFFERENT COMBINATIONS OF THE FLAGS.             *)
(*                                                                 *)
(*    COMMENTS:                                                    *)
(*        NONE                                                     *)
(*                                                                 *)
(*    CHANGE CONTROL:                                              *)
(*        NONE                                                     *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN          W315 2G  CREATED: 85/05/12       *)
(*    VERSION: 1                             REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : READEF                                          *)
(*                                                                   *)
(*    FUNCTION : READ AN EXCHANGE FORMAT RECORD                      *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*        CALLED BY:                                                 *)
(*          GETTOKEN                                                 *)
(*          GET-COMMENT                                              *)
(*          GET-QUOTE                                                *)
(*          GETSEMI                                                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        EFFILE   I   THE EXCHANGE FORMAT FILE                      *)
(*        EFREC    O   AN EXCHANGE FORMAT RECORD                     *)
(*        ENDFIL   O   THE END-OF-FILE FLAG                          *)
(*        RC       O   THE RETURN CODE                               *)
(*                       0   GOOD                                    *)
(*                       >0  FAILURE                                 *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        USES EFFILE FOR THE EXCHANGE FORMAT FILE FROM WHICH TO     *)
(*        OBTAIN THE RECORDS                                         *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/27     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : RETMAP                                            *)
(*                                                                   *)
(*  FUNCTION : RETRIEVES THE ENTITY KEY USING THE EXCHANGE FORMAT    *)
(*             POINTER AS AN INDEX INTO A STORAGE ARRAY.             *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    FILL-VALUE-IN-CONSTITUENT                                      *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFPTR            I  THE EXCHANGE FORMAT POINTER                *)
(*    KEY              O  THE ENTITY KEY                             *)
(*    RC               O  RETURN CODE                                *)
(*                        0   : OK                                   *)
(*                        1   : KEY DOES NOT EXIST                   *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. DAVIS            K315 2G  CREATED: 84/10/15  CC  *)
(*    VERSION: 1                              REVISED: 86/06/06  CC  *)
(*                                                                   *)
(*    ROUTINE NAME : STEP                                            *)
(*                                                                   *)
(*    FUNCTION : STEP THROUGH PROCESSORS                             *)
(*                  THE SYSTEM IS INITIALIZED AND ONE OF THE         *)
(*                  PROCESSORS IS CALLED.  ANY CONVERSIONS BETWEEN   *)
(*                  NATIVE AND WORKING FORM NECESSARY ARE            *)
(*                  ACCOMPLISHED.                                    *)
(*                  NOTE : LOWER LEVEL ROUTINES ARE TO PUT OUT THEIR *)
(*                         OWN MESSAGES.                             *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE. OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        CONTROL                                                    *)
(*      CALLS:                                                       *)
(*        PRE                                                        *)
(*        POST                                                       *)
(*        FILEWF                                                     *)
(*        ACSSWF                                                     *)
(*        CONVRT                                                     *)
(*        WFPRNT                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        USEREC   I   CONTAINS INFO ON DRAWING, TRANSFER METHOD, ETC*)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE NEEDS OPEN/CLOSE NO FILES.                    *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*    86/06/06 - A.M.WHELAN-CHANGED TO CALL PERROR                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: LAUREL A. CASSIN        ORG_ID  CREATED: 85/02/06  CC  *)
(*    VERSION: 1.0                            REVISED:           CC  *)
(*                                                                   *)
(*    ROUTINE NAME: STORADB                                          *)
(*                                                                   *)
(*    FUNCTION: TO PUT A VALUE IN THE ATTRIBUTE DATA BLOCK           *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLS:                                                       *)
(*        STORINT                                                    *)
(*        STORREL                                                    *)
(*        STORCHR                                                    *)
(*        STORLOG                                                    *)
(*        STORSCL                                                    *)
(*      CALLED BY:                                                   *)
(*        FILLADB                                                    *)
(*        FILLTOP                                                    *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*    INPUT :                                                        *)
(*       ENTABD - THE ARRAY OF ADB INFO.                             *)
(*                                                                   *)
(*       STORREC- A RECORD WITH INFORMATION THAT STORADB NEEDS.      *)
(*                 SUCH AS LENGTH,TYPE,DISP.                         *)
(*    OUTPUT :                                                       *)
(*       STORREC- A RECORD WITH INFORMATION FROM STORADB, SUCH AS    *)
(*                 WHETHER THE DATA FIELD IS AN INTEGER,REAL,CHAR,   *)
(*                 LOGICAL,SCALAR,SET,POINTER,SUBENTITY,STRUCTURE.   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        IT READS THE BYTES OF THE ATTRIBUTE DATA BLOCK.            *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. CASSIN            W315 2G CREATED: 85/02/06      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STORCHR                                          *)
(*                                                                   *)
(*    FUNCTION: TO PUT A CHARACTER STRING INTO THE ADB               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY STORADB                                          *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       STORREC   I   THE RECORD WITH THE INFORMATION ON IT         *)
(*       ENTADB    O   AREA TO PUT THE INTEGER INTO                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        MOVE A STRING INTO THE ADB                                 *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                          ñ.ã(                     *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. CASSIN            W315 2G CREATED: 85/02/06      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STORHAF                                          *)
(*                                                                   *)
(*    FUNCTION: TO PUT A SCALAR INTO THE ADB                         *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY STORINT                                          *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       STORREC   I   THE RECORD WITH THE INFORMATION ON IT         *)
(*       ENTADB    O   AREA TO PUT THE INTEGER INTO                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        MOVE A SCALAR INTO THE ADB                                 *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                          ñ.ã(                     *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. CASSIN            W315 2G CREATED: 85/02/06      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STORINT                                          *)
(*                                                                   *)
(*    FUNCTION: TO PUT AN INTEGER INTO THE ADB                       *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        STORADB                                                    *)
(*      CALLS:                                                       *)
(*        STORSCL                                                    *)
(*        STORHAF                                                    *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       STORREC   I   THE RECORD WITH THE INFORMATION ON IT         *)
(*       ENTADB    O   AREA TO PUT THE INTEGER INTO                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        IF A 1-BYTE INTEGER CALL STORSCL, ELSE MOVE BYTES FROM     *)
(*        ADB TO TEMPORARY ARRAY.  EQUIVALENCE TO INTEGER.           *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. CASSIN            W315 2G CREATED: 85/02/06      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STORLOG                                          *)
(*                                                                   *)
(*    FUNCTION: TO PUT A LOGICAL INTO THE ADB                        *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY STORADB                                          *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       STORREC   I   THE RECORD WITH THE INFORMATION ON IT         *)
(*       PASADB    O   AREA TO PUT THE INTEGER INTO                  *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        MOVE A LOGICAL INTO THE ADB                                *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                          ñ.ã(                     *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:  A. M. WHELAN              ORG_ID  CREATED: 86/05/27     *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : STORMAP                                           *)
(*                                                                   *)
(*  FUNCTION : STORES THE EXCHANGE FORMAT POINTER AND ASSOCIATED     *)
(*             ENTITY KEY IN AN ARRAY.  WRITES OUT THE POINTER,      *)
(*             KIND, AND IDENT TO AN OUTPUT FILE.                    *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    NONE                                                           *)
(*  CALLED BY:                                                       *)
(*    PRDATA                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    EFPTR             I  THE EXCHANGE FORMAT POINTER               *)
(*    KIND              I  THE ENTITY KIND                           *)
(*    IDENT             I  THE ENTITY IDENT                          *)
(*    KEY               I  THE ENTITY KEY                            *)
(*    MAPFIL           I/O THE MAPPING OUTPUT FILE                   *)
(*    RC                O  RETURN CODE                               *)
(*                         0   : OK                                  *)
(*                         >0  : FAILURE                             *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: L. A. CASSIN            W315 2G CREATED: 85/02/06      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STORREL                                          *)
(*                                                                   *)
(*    FUNCTION: PUT AN 8-BYTE REAL NUMBER INTO THE ADB               *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*         IBM PASCAL LANGUAGE                                       *)
(*         IBM 30XX,43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*         CALLED BY STORADB                                         *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       STORREC   I   RECORD WITH THE INFORMATION ON IT             *)
(*       ENTADB    O   AREA TO PUT THE REAL INTO                     *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*         PUT BYTES FROM TEMPORARY ARRAY INTO ADB.  EQUIVALENCE TO  *)
(*         A REAL.                                                   *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR: A.M. WHELAN                  W315 2G CREATED: 85/02/06   *)
(*  VERSION: 1                                    REVISED            *)
(*                                                                   *)
(*  ROUTINE NAME: STORSCL                                            *)
(*                                                                   *)
(*  FUNCTION: TO PUT A SCALAR INTO THE ADB                           *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*    CALLED BY STORADB.                                             *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    STORREC     I   THE RECORD WITH THE INFORMATION ON IT          *)
(*    ENTADB      O   AREA TO PUT THE INTEGER INTO                   *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*    MOVE A SCALAR INTO THE ADB                                     *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: A. M. WHELAN            K315 2G CREATED: 85/12/04      *)
(*    VERSION: 1                              REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME: STRNAM                                           *)
(*                                                                   *)
(*    FUNCTION :                                                     *)
(*        GET THE STRING NAME CORRESPONDING TO THIS KIND             *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*        CALLED BY:                                                 *)
(*          MAKDAT                                                   *)
(*          CRHEAD                                                   *)
(*        CALLS:                                                     *)
(*          NONE                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        KIND     I   THE KIND VALUE                                *)
(*        STR_NAME O   THE STRING NAME CORRESPONDING TO THE KIND     *)
(*        IRC      O   RETURN CODE                                   *)
(*                      0 : OK                                       *)
(*                      2 : KIND VALUE NOT VALID                     *)
(*                                                                   *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       SEARCH KNDTBL FOR KIND AND ASSOCIATED ENTITY NAME.          *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: LAUREL A CASSIN          W315 2G CREATED: 84/12/15     *)
(*    VERSION: 1.0                             REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME : TABRED                                          *)
(*                                                                   *)
(*    FUNCTION: CONVERT A VALUE TO CHARACTER AND CONCATENATE THE     *)
(*         PUNCTUATION ONTO IT FOR THE EXCHANGE FORMAT DATA FILE     *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        OPERAT                                                     *)
(*      CALLS:                                                       *)
(*        PUNTAB                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       CUTER      I  VALUE TO BE CONVERTED, IN A RECORD BY TYPE   *)
(*       SFLAG      I  A FLAG FOR A STRUCTURE THAT SIGNALS 1:START, *)
(*                     2:END STRUCTURE AND 3:NONE                    *)
(*       FFLAG      I  A FLAG FOR A FIELD THAT SIGNALS 1:LAST,      *)
(*                     2:NOT LAST AND 3:ONLY                         *)
(*       AFLAGS     I  A FLAG FOR AN ARRAY THAT SIGNALS 1:START,    *)
(*                     2:LAST, 3:NEITHER AND 4:BOTH-FOR BOTH DIMENSIONS*)
(*       VARSTR     O  CONVERTED STRING OF VALUE AND PUNCTUATION    *)
(*       LENG       O  LENGTH OF VARSTR                             *)
(*       RC         O  RETURN CODE     0:GOOD   4:BAD               *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE USES A TABLE WITH EVERY POSSIBLE PUNCTUATION  *)
(*        NEEDED FOR THE EXCHANGE FORMAT FILE                        *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: J.M. PURSES              W315 2G CREATED: 6/24/85      *)
(*    VERSION:  1.0                           REVISED:              *)
(*                                                                   *)
(*    ROUTINE NAME: TRMPRT                                           *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       THIS ROUTINE DETERMINES WEITHER A PRINT "ALL ENTITIES"      *)
(*       OR A PRINT "BY KIND OF ENTITY" WILL BE RUN ON THE WORKING   *)
(*       FORM MODEL.                                                 *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        WFPRNT                                                     *)
(*      CALLS:                                                       *)
(*        GETNEW                                                     *)
(*        PRTENT                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        TTYIN   I TERMINAL INPUT                                   *)
(*        TTYOUT  O TERMINAL OUTPUT                                  *)
(*        OUTFIL  O FILE OUTPUT                                      *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE NEEDS NO OPEN/CLOSE FILES                     *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:   PHIL DORR                ORG_ID  CREATED:  6/2/86      *)
(*  VERSION:  1                                REVISED:              *)
(*                                                                   *)
(*  ROUTINE NAME : VALCON                                            *)
(*                                                                   *)
(*  FUNCTION : VALIDATE THE FIELD                                    *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*  CALLED BY:                                                       *)
(*    CONTOK                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN         I  THE INPUT TOKEN                               *)
(*    CONREC      I/O THE FIELD INFORMATION RECORD                   *)
(*    VALID_FIELD   O  A FLAG INDICATING THE VALIDITY OF THE FIELD   *)
(*    RC            O  RETURN CODE                                   *)
(*                     0  : OK                                       *)
(*                     >0 : FAILURE                                  *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*     WILL CHECK THE TOKEN PASSED FOR VALIDITY AS THE DATATYPE      *)
(*     INDICATED IN TOKEN_TYPE.                                      *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: J.M. PURSES              W315 2G CREATED: 6/24/85      *)
(*    VERSION:  1.0                            REVISED:             *)
(*                                                                   *)
(*    ROUTINE NAME: WFPRNT                                           *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       THIS IS THE MAIN ROUTINE THAT INITIATES THE PRINT OF A      *)
(*       WORKING FORM MODEL.  IT ALLOWS FOR EITHER A PRINT BY        *)
(*       ENTITY KIND OR ALL ENTITIES.                                *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*      CALLED BY:                                                   *)
(*        STEP                                                       *)
(*      CALLS:                                                       *)
(*        TRMPRT                                                     *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE NEEDS NO OPEN/CLOSE FILES                     *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*  AUTHOR:   PHIL DORR                  ORG_ID  CREATED: 5/30/86    *)
(*  VERSION:  1                                  REVISED:            *)
(*                                                                   *)
(*  ROUTINE NAME : WRITEMSG                                          *)
(*                                                                   *)
(*  FUNCTION : WRITE TO THE MESSAGE FILE                             *)
(*                                                                   *)
(*  ENVIRONMENT:                                                     *)
(*    IBM PASCAL LANGUAGE                                            *)
(*    IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.       *)
(*                                                                   *)
(*  EXECUTION PROCEDURE:                                             *)
(*  CALLS:                                                           *)
(*    PERROR                                                         *)
(*  CALLED BY:                                                       *)
(*    GET_COMMENT                                                    *)
(*    PRHEAD                                                         *)
(*    PRDECL                                                         *)
(*                                                                   *)
(*  DESCRIPTION OF ARGUMENTS:                                        *)
(*    TOKEN     I   THE INPUT TOKEN                                  *)
(*    RC        O   RETURN CODE                                      *)
(*                  0  : OK                                          *)
(*                  >0 : FAILURE                                     *)
(*                                                                   *)
(*  PROCESSING DESCRIPTION:                                          *)
(*    THIS PROCEDURE WILL PRINT THE TOKEN PASSED TO IT ON THE        *)
(*    FILE OUTPUT.                                                   *)
(*                                                                   *)
(*  CHANGE CONTROL:                                                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR: LAUREL A. CASSIN        W315 2G CREATED: 84/12/15      *)
(*    VERSION: 1.0                            REVISED:               *)
(*                                                                   *)
(*    ROUTINE NAME : WRTREC                                          *)
(*                                                                   *)
(*    FUNCTION: WRITE THE STRING TO THE RECORD AND THEN TO THE FILE  *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.   *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*     CALLS:                                                        *)
(*       NONE                                                        *)
(*     CALLED BY:                                                    *)
(*       PRE                                                         *)
(*       MAKDAT                                                      *)
(*       DOTOP                                                       *)
(*       ATRDAT                                                      *)
(*       NILCON                                                      *)
(*       OPERAT                                                      *)
(*       CRHEAD                                                      *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        VARSTR   I   THE STRING                                    *)
(*        VARLEN   I   THE LENGTH OF VARSTR                          *)
(*        EFFILE   I   THE EXCHANGE FORMAT FILE                      *)
(*        WRC      O   THE RETURN CODE                               *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE FILLS THE RECORD(DATREC) AND THEN             *)
(*        THE FILE(EFFILE) MAKING SURE TO END THE LINE WITH          *)
(*        A COMMA, SEMICOLON, OR COLON OR SLASH.                     *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*       LORI A DAVIS    CHANGE CONCATENTATION AND END OF RECORD     *)
(*-------------------------------------------------------------------*)
```

APPENDIX C

TRANSLATOR DATA DICTIONARY

This appendix provides the data dictionary for the PDDI Translator. The following index provides a brief description of the data entries function. The entities are listed in alphabetic order.

EDBDEF - Contains the file structure for accessing the Working Form data dictionary

PRINT - Turns output print off

```
(*************************************************************************)
(*                                                                       *)
(* EDBDEF                                                                *)
(*                                                                       *)
(* CONTAINS THE FILE STRUCTURE FOR ACCESSING THE WORKING FORM            *)
(*     DATA DICTIONARY                                                   *)
(*                                                                       *)
(*************************************************************************)

 TYPE
       SCALAR_REC = RECORD
                 DNUMB : INTEGER;
                 DSCALAR : ARRAY (.1..10.) OF
                      PACKED ARRAY (.1..16.) OF CHAR
                 END;

       ENTITY_REC = RECORD
                 CLDISP : INTEGER;
                 DNUMB : INTEGER;
                 DENTITY : ARRAY (.1..36.) OF INTEGER
                 END;

       SUBENT_REC = RECORD
                 CLDISP : INTEGER;
                 DNUMB : INTEGER;
                 DSUBENT : ARRAY (.1..36.) OF INTEGER
                 END;

       D_ENTITY = RECORD
                DNAME : PACKED ARRAY(.1..16.) OF CHAR;
                CSORDR : INTEGER;
                DMIN : INTEGER;
                DMAX : INTEGER;
                DTYPE: INTEGER;
                DSIZE : INTEGER;
                DDISP : INTEGER;
                CASE DTYPE : OF
                   5 : (DSCA : SCALAR_REC);
                   7 : (DENT : ENTITY_REC);
                   8 : (DSEN : SUBENT_REC)
                END;

       DICTYP = ARRAY (.1..45.) OF D_ENTITY;
```

```
(*************************************************************************)
(*                                                                       *)
(* PRINT                                                                 *)
(*                                                                       *)
(* USED TO PREVENT LISTING OF MAS ROUTINES DURING COMPILATION            *)
(*                                                                       *)
(*************************************************************************)
%PRINT OFF
```

APPENDIX D

ACCESS SOFTWARE HIERARCHY

This appendix provides a cross-reference listing for PDDI Access Software (MAS) routines. The Control Sections (CSECTs) that are referred to by a particular CSECT (routine) are provided.

Routine:	Refers to:
ADCRBM	
	NEWCRB
	EXPCRB
	EXCRBE

Routine:	Refers to:
ADRLSM	
	NEWLSM
	LSTMXLNM
	LSTLNM
	DISPLSM
	MOVRLSM

Routine:	Refers to:
ADSCH	
	FDSCH
	NEWNSI
	CRURUL
	ADSCHR
	ADTLSM

Routine:	Refers to:
ADSCHR	
	FDSCH
	ADRLSM
	EXPSUDB

Routine:	Refers to:
ADTLSM	
	NEWLSM
	LSTMXLNM
	LSTLNM
	MOVRLSM
	DISPLSM

Routine:	Refers to:
ADTNM	
	ADTLSM

Routine:	Refers to:
CHKDEL	
	ADTLSM
	ADTNM
	DELCNST
	INDLSM
	LSTLNM
	MSTART
	MSTOP
	RDLSM
	RSTLSM
	SETRULS

Routine:	Refers to:
CHKTDEL	
	ADTNM
	DETCNST
	DELRLSM
	MSTART
	MSTOP
	LSTLNM
	RDLSM
	RSTLSM
	SETRULS

Routine:	Refers to:
CMPCRB	
	MASNEW
	DISPCRB

Routine:	Refers to:
CNNODM	
	MRKNM
	NEWNM
	ADTLSM
	VERCN
	RLSNM
	CREMM
	TVERIFY

Routine:	Refers to:
CNVOSP	

Routine:	Refers to:
CNVRR	

Routine:	Refers to:
CPYAUDB	
	AMPXMOVE
	NEWSADB

Routine:	Refers to:
CPYCST	
	LSTLNM
	MRGTLSM
	NEWNM
	FDSCH

Routine:	Refers to:
CPYLSM	
	LSTLNM
	NEWLSM
	MOVRLSM

Routine:	Refers to:
CPYNM	
	NEWNM
	RSTLSM
	RDLSM
	ADTLSM

Routine:	Refers to:
CRCLST	
	RSTLSM
	RDLSM
	CREMM

Routine:	Refers to:
CRCNM	
	CRCLST

Routine:	Refers to:
CRDLST	
	ADTLSM
	DISPLSM
	DISPNM
	EXPCLSM
	MRGTLSM
	NEWLSM
	NEWNM
	RSTSFLG
	SORTDLST

Routine:	Refers to:
CREMM	
	ADTLSM

Routine:	Refers to:
CRURUL	

Routine:	Refers to:
DELALNL	
	RSTLSM
	RDLSM
	DISPEMM
	DISPLSM

Routine:	Refers to:
DELCNST	
	CHKDEL
	ADTLSM
	ADTNM
	INDLSM
	INNM
	MSTART
	MSTOP
	RDLSM
	RSTLSM
	SETRULS

Routine:	Refers to:
DELCRBE	
	FNDCRBE
	CMPCRB
	EXCRBE

Routine:	Refers to:
DELEMM	
	RSTLSM
	RDLSM
	DELRLSM
	DISPEMM

Routine:	Refers to:
DELENTY	
	ADTNM
	ADTLSM
	DELRLSM
	DELRUL
	DISPNM
	FNDURUL
	INDLSM
	NEWNM
	RDLSM
	RSTLSM
	XIEMM

Routine:	Refers to:
DELPLST	
	LSTMXLNM
	LSTLNM
	MOVRLSM
	NEWLSM
	DISPLSM

Routine:	Refers to:
DELPNLA	
	NEWLSM
	RDLSM
	LSTLNM
	LSTMXLNM
	DISPLSM
	ADTLSM
	DISPEMM
	MOVRLSM
	MRGTLSM

Routine:	Refers to:
DELRLSM	
	LSTMXLNM
	LSTLNM
	MOVRLSM
	NEWLSM
	DISPLSM

Routine:	Refers to:
DELRUL	
	ADTLSM
	CHKDEL
	CPYLSM
	DELCNST
	DISPLSM
	ELDNM
	MRGTLSM
	NEWLSM
	NEWNM
	RDLSM
	RSTLSM
	XIEMM

Routine:	Refers to:
DELSCH	
	FDSCH
	INDLSM
	DELRLSM
	DELPLST
	DISPEMM

Routine:	Refers to:
DELTLSM	
	LSTLNM
	LSTMXLNM
	NEWLSM
	MOVRLSM
	DISPLSM

Routine:	Refers to:
DETCLST	
	ADTNM
	DETRUL
	FNDURUL
	RDLSM
	RSTLSM

Routine:	Refers to:
DETCNST	
	ADTNM
	CHKTDEL
	DELRLSM
	MSTART
	MSTOP
	RDLSM
	RSTLSM
	SETRULS

Routine:	Refers to:
DETRUL	
	ADTNM
	CHKTDEL
	DETCNST
	DELRLSM
	DISPNM
	INNM
	MRGTNM
	NEWNM
	RSTLSM
	RDLSM

Routine:	Refers to:
DIFLSM	
	LSTLNM
	CPYLSM
	NEWLSM
	INDLSM

Routine:	Refers to:
DISPCRB	
	MASDSP

Routine:	Refers to:
DISPEMM	
	DISPLSM

Routine:	Refers to:
DISPKND	
	RSTLSM
	RDLSM
	DISPEMM

Routine:	Refers to:
DISPLSM	

Routine:	Refers to:
DISPNDM	
	DISPCRB
	RSTLSM
	RDLSM
	DISPKND
	DISPEMM
	DELRLSM
	DELALNL

Routine:	Refers to:
DISPNM	
	DELRLSM
	DISPEMM
	RDLSM
	RSTLSM

Routine:	Refers to:
ELDNM	
	LSTLNM
	RSTLSM
	RDLSM
	ADTLSM
	DISPLSM

Routine:	Refers to:
ELMNODM	
	REVAADB

Routine:	Refers to:
EXCRBE	
	EXPCRB

Routine:	Refers to:
EXPCLSM	
	ADTLSM
	RDLSM
	RSTLSM

Routine:	Refers to:
EXPCRB	
	MASNEW
	DISPCRB

Routine:	Refers to:
EXPSUDB	
	AMPXMOVE
	NEWSADB

Routine:	Refers to:
EXPULSM	
	ADTLSM
	RDLSM
	RSTLSM

Routine:	Refers to:
EXPUNM	
	NEWLSM
	EXPULSM
	DISPLSM

Routine:	Refers to:
FDSCH	
	RDRLSM

Routine:	Refers to:
FNDCRBE	

Routine:	Refers to:
FNDSKIND	
	RSTLSM
	RDLSM

Routine:	Refers to:
FNDURUL	
	FDSCH

Routine:	Refers to:
GTCRBE	

Routine:	Refers to:
INDLSM	
	LSTLNM

Routine:	Refers to:
INNM	
	INDLSM

Routine:	Refers to:
INTLSM	
	LSTLNM
	NEWLSM
	INDLSM

Routine:	Refers to:
INTNM	
	NEWNM
	INTLSM

Routine:	Refers to:
LSTLNM	

Routine:	Refers to:
LSTMXLNM	

Routine:	Refers to:
MAEA	
	RDLSM
	RSTLSM
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAEAI	
	CNVRR
	CNVOSP
	DISPNM
	EXPCLSM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAEAV	
	CNVRR
	CNVOSP
	MSTART
	MSTOP

Routine:	Refers to:
MAEC	
	CNVRR
	CNVOSP
	DISPNM
	MRGTLSM
	MSTART
	MSTOP
	NEWNM
	NODECNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAECI	
	CNVRR
	CNVOSP
	DELRLSM
	DISPNM
	EXPCLSM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAECIK	
	ADTLSM
	CNVRR
	CNVOSP
	DISPNM
	EXPCLSM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAECMP	
	ADTLSM
	CNVRR
	CNVOSP
	DISPNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM
	SETRULS
	TVERIFY

Routine:	Refers to:
MAECQY	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	SETRULS
	TVERIFY

Routine:	Refers to:
MAECR	
	CNVRR
	CNVOSP
	CRCLST
	MSTART
	MSTOP
	NEWNODE
	TVERIFY
	VERCR

Routine:	Refers to:
MAECTK	
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAECXQ	
	ADCRBM
	ADTLSM
	CNVRR
	CNVOSP
	DELCRBE
	DISPCRB
	DISPNM
	GTCRBE
	FNDCRBE
	LSTLNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY
	UPDCRBE

Routine:	Refers to:
MAED	
	CNVRR
	CNVOSP
	CPYNM
	DELRUL
	DISPLSM
	DISPNM
	ELDNM
	LSTLNM
	MSTART
	MSTOP
	NEWLSM
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	SORTDLST
	TVERIFY
	VERDEL

Routine:	Refers to:
MAEDI	
	CNVRR
	CNVOSP
	CRDLST
	DELRUL
	DISPLSM
	DISPNM
	LSTLNM
	MSTART
	MSTOP
	NEWLSM
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY
	VERDEL

Routine:	Refers to:
MAEDT	
	CPYNM
	DETRUL
	DISPNM
	ELDNM
	LSTLNM
	MRGTNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	SORTDLST
	TVERIFY
	VERDEL
	CNVRR
	CNVOSP

Routine:	Refers to:
MAEDTI	
	CNVRR
	CNVOSP
	CRDLST
	LSTLNM
	DISPLSM
	DISPNM
	DETRUL
	MRGTNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY
	VERDEL

Routine:	Refers to:
MAEDTS	
	CPYNM
	DETRUL
	DISPNM
	ELDNM
	LSTLNM
	MRGTNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	SORTDLST
	TVERIFY
	VERDEL
	CNVRR
	CNVOSP

Routine:	Refers to:
MAEGKN	
	CNVRR
	CNVOSP
	MSTART
	MSTOP

Routine:	Refers to:
MAEGTK	
	CNVRR
	CNVOSP
	ELMNODM
	MSTART
	MSTOP
	TVERIFY
	VERGT

Routine:	Refers to:
MAEKND	
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAERST	
	RSTLSM
	RDLSM
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAESVL	
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAESWA	
	RSTLSM
	RDLSM
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MAESWT	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	RDLSM
	RSTLSM

Routine:	Refers to:
MAEU	
	CNVRR
	CNVOSP
	DISPNM
	MRGTLSM
	MSTART
	MSTOP
	NEWNM
	NODEUNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAEUD	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	REVNODM
	TVERIFY
	VERUD

Routine:	Refers to:
MAEUI	
	CNVRR
	CNVOSP
	DELRLSM
	DISPNM
	EXPULSM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAEUIK	
	ADTLSM
	CNVOSP
	CNVRR
	DISPNM
	EXPULSM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RSTLSM

Routine:	Refers to:
MAEUSR	
	CNVRR
	CNVOSP
	LSTLNM
	MSTART
	MSTOP

Routine:	Refers to:
MAEUXQ	
	ADTLSM
	CNVRR
	CNVOSP
	DISPNM
	INDLSM
	LSTLNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY

Routine:	Refers to:
MAEXEQ	
	CNVRR
	CNVOSP
	LSTLNM
	MSTART
	MSTOP
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY

Routine:	Refers to:
MAINIT	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	NEWNDM
	OSTART
	TVERIFY

Routine:	Refers to:
MAKCNT	
	FDSCH
	CNVRR
	CNVOSP
	MSTART
	MSTOP

Routine:	Refers to:
MAKILL	
	MSTART
	NDSRML
	CNVRR
	CNVOSP
	MSTOP

Routine:	Refers to:
MAKXEQ	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	FDSCH
	LSTLNM
	RDRLSM
	TVERIFY

Routine:	Refers to:
MAL	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	NEWNM

Routine:	Refers to:
MALAND	
	CNVRR
	CNVOSP
	DISPNM
	INTLSM
	MSTART
	MSTOP
	NEWNM

Routine:	Refers to:
MALATC	
	ADTNM
	CNVRR
	CNVOSP
	CRCNM
	CREMM
	MRGTNM
	MSTART
	MSTOP
	TVERIFY
	VERAPN

Routine:	Refers to:
MALCPY	
	CNVRR
	CNVOSP
	CPYNM
	MSTART
	MSTOP

Routine:	Refers to:
MALD	
	CNVRR
	CNVOSP
	DELRLSM
	DISPEMM
	MSTART
	MSTOP
	RDLSM
	RSTLSM

Routine:	Refers to:
MALDA	
	RDTLSM
	DELPNLA
	DISPLSM
	MSTART
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MALDI	
	CNVRR
	CNVOSP
	INDLSM
	MSTART
	MSTOP
	RDTLSM
	DELPNLA
	DISPLSM

Routine:	Refers to:
MALFND	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	RDLSM

Routine:	Refers to:
MALGTK	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	RDRLSM

Routine:	Refers to:
MALINS	
	ADRLSM
	ADTLSM
	CNVRR
	CNVOSP
	FNDCRBE
	GTCRBE
	LSTLNM
	MSTART
	MSTOP
	RDLSM
	RSTLSM
	UPDCRBE

Routine:	Refers to:
MALK	
	CNVRR
	CNVOSP
	CPYCST
	DISPNM
	FDSCH
	MSTART
	MSTOP
	NEWNM

Routine:	Refers to:
MALKL	
	ADTNM
	CNVRR
	CNVOSP
	FDSCH
	FNDSKIND
	MSTART
	MSTOP
	NEWNM
	RSTLSM
	RDLSM

Routine:	Refers to:
MALN	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	NEWNM
	NEWLSM

Routine:	Refers to:
MALNO	
	CNVRR
	CNVOSP
	LSTLNM
	MSTART
	MSTOP

Routine:	Refers to:
MALNOT	
	CNVRR
	CNVOSP
	DIFLSM
	DISPNM
	MSTART
	MSTOP
	NEWNM

Routine:	Refers to:
MALOCK	
	CNVRR
	CNVOSP
	MSTART
	MSTOP

Routine:	Refers to:
MALOR	
	CNVRR
	CNVOSP
	DISPNM
	ELDNM
	MRGTLSM
	MSTART
	MSTOP
	NEWNM

Routine:	Refers to:
MALPUT	

Routine:	Refers to:
MALRD	
	ADCRBM
	CNVRR
	CNVOSP
	DELCRBE
	DISPCRB
	FNDCRBE
	GTCRBE
	MSTART
	MSTOP
	RDRLSM
	UPDCRBE

Routine:	Refers to:
MALRDE	
	CNVRR
	CNVOSP
	ELDNM
	MSTART
	MSTOP

Routine:	Refers to:
MALREP	
	ADTLSM
	CNVRR
	CNVOSP
	CPYLSM
	DELCRBE
	DELRLSM
	DISPCRB
	FNDURUL
	LSTLNM
	MSTART
	MSTOP
	RDLSM
	RSTLSM

Routine:	Refers to:
MALRMV	
	CNVRR
	CNVOSP
	DELCRBE
	DELPLST
	DELRLSM
	DELRUL
	DISPCRB
	DISPNM
	DISPLSM
	FNDCRBE
	GTCRBE
	LSTLNM
	MSTART
	MSTOP
	NEWLSM
	NEWNM
	SETRULS
	UPDCRBE

Routine:	Refers to:
MALROR	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	ORDRLST
	TVERIFY

Routine:	Refers to:
MALRPL	
	ADTLSM
	CNVRR
	CNVOSP
	DELPLST
	DELRLSM
	DELRUL
	DISPLSM
	DISPNM
	INDLSM
	LSTLNM
	MSTART
	MSTOP
	NEWNM
	NEWLSM
	REVRLSM

Routine:	Refers to:
MALRVS	
	CNVRR
	CNVOSP
	RVRLSM
	DISPLSM
	MSTART
	MSTOP

Routine:	Refers to:
MALSRT	
	CNVRR
	CNVOSP
	SORTLSM
	MSTART
	MSTOP
	TVERIFY

Routine:	Refers to:
MALSTF	
	ADCRBM
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	UPDCRBE

Routine:	Refers to:
MALSTR	
	ADCRBM
	CNVRR
	CNVOSP
	LSTLNM
	MSTART
	MSTOP
	UPDCRBE

Routine:	Refers to:
MALXEQ	
	ADCRBM
	ADTLSM
	CNVRR
	CNVOSP
	DELCRBE
	DISPCRB
	DISPNM
	FNDCRBE
	GTCRBE
	LSTLNM
	MSTART
	MSTOP
	NEWNM
	RDLSM
	RDRLSM
	RSTLSM
	TVERIFY
	UPDCRBE

Routine:	Refers to:
MAQURY	
	CNVRR
	CNVOSP
	MSTART
	MSTOP

Routine:	Refers to:
MASALOC	*

Routine:	Refers to:
MASDSP	

Routine:	Refers to:
MASMSZ	
	MSTART
	NDSFCT
	MSTOP
	CNVRR
	CNVOSP

Routine:	Refers to:
MASNEW	

Routine:	Refers to:
MASOVR	

Routine:	Refers to:
MAUPDT	
	CNVRR
	CNVOSP
	MSTART
	MSTOP
	RDLSM
	RSTLSM

Routine:	Refers to:
MIDBD	
	CNVOSP
	CNVRR
	XIEMM
	ELDNM
	MSTART
	MSTOP
	RDLSM
	RSTLSM

Routine:	Refers to:
MIDBRV	
	ADTLSM
	CNVRR
	CNVOSP
	DELCRBE
	DELPLST
	DELRLSM
	DELENTY
	DELRUL
	DISPCRB
	DISPLSM
	FNDCRBE
	GTCRBE
	LSTLNM
	MSTART
	MSTOP
	NEWLSM
	UPDCRBE

Routine:	Refers to:
MOVRLSM	
	AMPXMOVE
	LSTLNM
	LSTMXLNM

Routine:	Refers to:
MRGTLSM	
	CPYLSM
	LSTMXLNM
	LSTLNM
	DISPLSM
	NEWLSM
	MOVRLSM

Routine:	Refers to:
MRGTNM	
	MRGTLSM

Routine:	Refers to:
MRKNM	
	ADTLSM
	NEWEMM
	NEWLSM

Routine:	Refers to:
MSTART	

Routine:	Refers to:
MSTOP	

Routine:	Refers to:
NDSCMM	

Routine:	Refers to:
NDSFCT	

Routine:	Refers to:
NDSGBM	

Routine:	Refers to:
NDSRML	

Routine:	Refers to:
NEWCRB	MASNEW

Routine:	Refers to:
NEWEMM	

Routine:	Refers to:
NEWIIM	NEWEMM NEWLSM CPYAUDB

Routine:	Refers to:
NEWLSM	

Routine:	Refers to:
NEWNDM	NEWEMM ADTLSM NEWNSR

Routine:	Refers to:
NEWNM	NEWEMM NEWLSM RDTLSM ADTLSM

Routine:	Refers to:
NEWNODE	NEWIIM ADSCH

Routine:	Refers to:
NEWNSC	NEWSCHC

Routine:	Refers to:
NEWNSI	NEWSCHI

Routine:	Refers to:
NEWNSR	NEWSCHR

Routine:	Refers to:
NEWSADB	AMPXNEW

Routine:	Refers to:
NEWSCHC	NEWIIT

Routine:	Refers to:
NEWSCHI	NEWIIM

Routine:	Refers to:
NEWSCHR	NEWIIM

Routine:	Refers to:
NODECNM	NEWNM CPYLSM

Routine:	Refers to:
NODEUNM	NEWNM CPYLSM

Routine:	Refers to:
OCOUNT	

Routine:	Refers to:
ORDRLST	
	CPYLSM
	DISPNM
	INNM
	LSTLNM
	NODEUNM
	RDLSM
	REVRLSM
	RSTLSM

Routine:	Refers to:
OSTART	

Routine:	Refers to:
PASASM	*

Routine:	Refers to:
RDLSM	
	LSTLNM

Routine:	Refers to:
RDNM	
	RDLSM
	ELMNODM

Routine:	Refers to:
RDRLSM	
	LSTMXLNM

Routine:	Refers to:
RDTLSM	
	LSTLNM

Routine:	Refers to:
REVAADB	
	AMPXMOVE

Routine:	Refers to:
REVNODM	
	REVSADB

Routine:	Refers to:
REVRLSM	
	LSTLNM

Routine:	Refers to:
REVSADB	
	AMPXMOVE
	NEWSADB

Routine:	Refers to:
RLSNM	
	DELTLSM
	DISPEMM
	RDLSM
	RDTLSM
	RSTLSM

Routine:	Refers to:
RSTLSM	

Routine:	Refers to:
RSTSFLG	
	RDLSM
	RSTLSM

Routine:	Refers to:
RVRLSM	
	NEWLSM
	RDRLSM
	ADTLSM
	LSTLNM

Routine:	Refers to:
SETRULS	
	INDLSM

Routine:	Refers to:
SETSWCI	
	RSTLSM
	RDLSM

Routine:	Refers to:
SORTDLST	
	RSTLSM
	RDLSM
	NEWLSM
	SRTBYCNT

Routine:	Refers to:
SORTLSM	
	LSTLNM
	ELMNODM
	ELMNODM

Routine:	Refers to:
SRTBYCNT	
	RSTLSM
	RDLSM
	ADTLSM

Routine:	Refers to:
UPDCRBE	
	FNDCRBE

Routine:	Refers to:
VERAPN	

Routine:	Refers to:
VERCN	

Routine:	Refers to:
VERCR	

Routine:	Refers to:
VERDEL	

Routine:	Refers to:
VERGT	

Routine:	Refers to:
VERUD	

Routine:	Refers to:
XIEMM	
	DELCRBE
	DELSCH
	DELEMM
	DISPCRB
	RDTLSM
	DELCRBE
	DELTLSM
	DELRLSM
	DISPCRB
	FNDCRBE
	GTCRBE
	INDLSM
	UPDCRBE
	RDTLSM
	DELTLSM
	DELRLSM

Routine:	Refers to:
XREMM	
	DELRLSM

APPENDIX E

ACCESS SOFTWARE ROUTINES

This appendix provides a listing of each procedure in the PDDI Access Software Package.

The routines are listed in alphabetic order. An index with a brief description of the routine function is provided.

A hierarchy dictionary is provided in Appendix D to show the relationship of the routines.

Routine Index. E-2
Routine Dictionary E-11

PDDI ACCESS SOFTWARE ROUTINE INDEX

ROUTINE NAME	FUNCTION
ADCRBM	- Adds a new CRB entry
ADRLSM	- Adds an entity after a relative position in a system list
ADSCH	- Connects an internal item to the correct portion of the NDS superstructure
ADSCHR	- Connects an internal item to the schema root
ADTLSM	- Adds an entity to a system list
ADTNM	- Adds an entity to the end of an application list
CHKDEL	- Check deletability of an entity relative to its users
CHKTDEL	- Check deletability of an entity relative to its users
CMPCRB	- Compresses the CRB
CNNODM	- Connects two entities
CNVOSP	- Convert out of space system error code to user recognizeable error code
CNVRR	- Gets the external return code corresponding to the internal format
CYPAUDB	- Stores the value of an application entity block in an uninitialized system UDB
CPYCST	- Adds the entities in a constituent list into a list
CPYLSM	- Copies the non-vacant elements of LIST FROM to LIST TO
CPYNM	- Creates a new list which contains a copy of the entities referenced by KEYL
CRCLST	- Creates relations between a user entity and a list of constituents
CRCNM	- Creates relations between a user entity and a list of constituents
CRDLST	- Creates a sorted inclusive list of an entity or a list of entities and their direct and indirect constituents
CREMM	- Creates a user-constituent relation between entities

CRURUL - Creates the user's rules for deletability

DELALNL - Disposes of all application lists in the Working Form model

DELCNST - Determine deletability of entity's constituents

DELCRBE - Deletes a CRB entry

DELEMM - Deletes all references to this entity from all application lists and disposes of the entity

DELENTY - Deletes the entity

DELPLST - Removes an entity from a specified position in a system list

DELPNLA - Deletes all non-"locked" APPL lists after a specified position in the LIST_OF_LISTS

DELRLSM - Removes an entity from a system list

DELRUL - Deletes an entity according to the delete rules

DELSCH - Disconnects an internal item from the correct portion of the NDS superstructure

DELTLSM - Removes the last non-vacant entity reference in a list

DETCLST - Checks the constituents of a marked entity or an entity on the MARKLIST to determine if the constituents may also be tested for deletion

DETCNST - Check deletability of entity's constituents

DETRUL - Tests delete of an entity according to the delete rules

DIFLSM - Creates a system's list consisting of all entities in LIST1 that are not in LIST2

DISPCRB - Disposes of CRB

DISPEMM - Releases all space allocated to an entity

DISPKND - Disposes of all entities of specific instance collector

DISPLSM - Deletes space allocated to a system list

DISPNDM - Forces an NDS out of memory

DISPNM - Removes all entities from the list and free the allocated space

ELDNM - Creates a list with all duplicate entities eliminated

ELMNODM - Returns an ENTBLOCK corresponding to a key

EXCRBE - Exchanges two entries in the CRB

EXPCLSM - Expands list with all of its constituents and places this expanded list in LISTOUT

EXPCRB - Expands the CRB

EXPSUDB - Expands a system UDB

EXPULSM - Places the expanded list with all of its users in LISTOUT

EXPUNM - Expands the list to include all users of the entities

FDSCH - Finds a Schema_Instance_Collector or Schema_Class entity on the specified Schema_Root's constituent list

FNDCRBE - Finds a specific entry in the CRB

FNDSKIND - Builds an array of kind value collected by a class or instance collector in the schema

FNDURUL - Gets the rule from the instance collector for a given ENTKEY

GTCRBE - Gets an entry in the CRB

INDLSM - Locates an entity in a system list

INNM - Indicates whether a list references an entity

INTLSM - Creates a list which is the intersection of two lists

INTNM - Creates a list which is the intersection of lists referenced by KEYL1 and KEYL2

LSTLNM - Returns the number of non-vacant entities in a system list

LSTMXLNM - Returns the number of entries allocated to a system list

MAEA - Activates an entity

MAEAI - Activates an entity or a list of entities and their inclusive constituents

MAEAV - Finds the present value of the activation setting for an entity

MAEC - Creates an application list of constituent entities

MAECI - Creates an application list of inclusive constituent entities

MAECIK - Creates a list of inclusive constituents by kind

MAECMP - Determine which of it's constituents an entity compresses with

MAECQY - Determine if an entity's user should compress with it

MAECR - Creates an entity

MAECTK - Returns the number of "KIND" values in the Working Form model

MAECXQ - Execute a procedure on a list, creating an output list

MAED - Deletes an entity or list of entities

MAEDI - Deletes inclusively an entity or list of entities

MAEDT - Tests delete an entity or list of entities

MAEDTI - Tests for inclusive deletion of an entity or list of entities and their direct and indirect constituents

MAEDTS - Test delete an entity or list of entities, and return three lists

MAEGKN - Retrieves the KIND value of an entity

MAEGTK - Retrieves the entity block which corresponds to KEYE

MAEKND - Returns a "KIND" value from the list of KINDS in the Working Form model

MAERST - Reset the specified flag in all entities in the working form model

MAESVL - Finds the current binary switch setting of an entity

MAESWA - Sets the process bit "off" in all entities in the model

MAESWT - Sets an entity switch or the switches for each entity in a list as requested by the user

MAEU - Creates a list of user entity references

MAEUD - Updates the entity block corresponding to a key

MAEUI - Creates an application list of inclusive user entities

MAEUIK - Creates a list of inclusive users by kind

MAEUSR - Determines if an entity has any users

MAEUXQ - Executes a procedure on the users of an entity

MAEXEQ - Executes a procedure on an entity, or a list of entities

MAINIT - Initializes the MAS network

MAKCNT - Determine the number of entities of a specified kink in the working form model

MAKILL - Deletes the Working Form model

MAKXEQ - Executes a procedure on all entities of a specified kind

MAL - Creates an empty list

MALAND - Creates an application list of entities common to two input lists

MALATC - Appends an entity or list (LIST2) to an entity or list (KEY1)

MALCPY - Makes a copy of a list

MALD - Deletes an application list

MALDA - Deletes all application lists that are not "locked"

MALDI - Deletes an application list and all lists after it that are not locked

MALFND - Finds the position of an entity (KEY2) in an application list (KEY1)

MALGTK - Gets the Nth key from the list

MALINS - Inserts an entity or list into a list

MALK - Creates a list of all entities of a specified kind

MALKL - Creates a list of entity kinds which are found within another list

MALN - Creates an empty list of a specified size

MALNO - Counts the entities on the list

MALNOT - Creates an application list of entities in KEY1 but not in KEY2

MALOCK - Sets an application list for delete or non-delete status

MALOR - Creates an application list from a BOOLEAN "OR" on two input lists

MALPUT - Inserts an entity into the IDB big list

MALRD - Reads the next entry in a directed list

MALRDE - Removes duplicate entries in a list

MALREP - Replaces a list

MALRMV - Removes an entity from a list

MALROR - Reorder an application list in user-constituent order

MALRPL - Replaces an entity in a list

MALRVS - Reverse the order of an application list

MALSRT - Sort an application list

MALSTF - Initializes for reading a directed list in forward order

MALSTR - Initializes for reading a directed list in reverse order

MALXEQ - Executes a procedure on an entity or a list of entities

MAQURY - Determine the value of the specified flag for an entity

MASALOC - MAS memory management runtime
MASDSP - Disposes of a MAS dynamically allocated memory area

MASMSZ - Returns the actual model space used and the amount of the free space in the allocated memory blocks of the model

MASNEW - Allocates a new dynamic memory area for MAS elements

MASOVR - MAS Memory management runtime

MAUPDT - Update the specified flag for an entity

MIDBD - Delete an entity without checking delete rules

MIDBRV - Remove an entity from a list without checking delete rules

MOVRLSM - Moves entities between system lists

MRGTLSM - Concatenates the entities in LIST2 to LIST1

MRGTNM - Concatenates the entities in LIST2 to LIST1

MRKNM - Marks the stack of lists so that the next release list will only destroy lists created after this mark operation

MSTART - Generates start statistics

MSTOP - Generates stop statistics

NDSCMM - Defines dummy program

NDSFCT - Computes the amount of used model space and the amount of free space in the allocated memory blocks

NDSGBM - Dummy procedure for compile time initialization of NDS global area

NDSRML - Releases all memory blocks allocated to the Working Form

NEWCRB - Creates a CRB

NEWEMM - Creates a new NDS object

NEWIIM - Creates a new entity and copies into it the application ENTDATA

NEWLSM - Initializes LISTREF and allocates enough space to hold size entities

NEWNDM - Creates a new empty model in memory

NEWNM - Creates an empty application list

NEWNODE - Creates a new entity in the NDS and copies into it the application ENTDATA

NEWNSC - Creates an empty schema class collector attached to the schema root

NEWNSI - Creates an empty schema instance collector attached to the schema root

NEWNSR - Creates a new null schema root and attaches it to the NDS

NEWSADB - Creates a new application data block

NEWSCHC - Creates an empty schema class entity attached to the schema root

NEWSCHI - Creates an empty schema instance collector entity attached to the schema root

NEWSCHR - Creates an empty rot collector entity attached to the NDS

NODECNM - Creates a list which contains a copy of the entity's constituent list

NODEUNM - Creates a list which contains a copy of the entity's user list

OCOUNT - MAS memory management runtime

ORDRLST - Reorder an application list

OSTART - MAS memory management runtime

PASASM - Link to a user defined procedure

RDLSM - Reads a system list as a first-in first-out order

RDNM - Reads the next entity in the KIND range from an application list

RDRLSM - Reads the last entity key from LISTREF

RDTLSM - Reads the last entity key from LISTREF

REVAADB - Assigns the value of a system UDB to an application ENTBLOCK

REVNODM - Revises an entity's user data block

REVRLSM - Changes an entity in a system list

REVSADB - Replace the value of a system ENTBLOCK with the value of ENTDEF

RLSNM - Releases all the lists on the current list of lists

RSTLSM - Resets position to indicate the beginning of a list

RSTSFLG - Resets the requested position in the internal MAS process flag (MAPROB) in the IIT to the requested BOOLEAN value

RVRLSM - Copy an application list in the reverse order

SETRULS - Sets delete flags according to user's dependence and strength rules

SETSWCI - Sets a switch in the user data block for each entity and all constituents inclusive

SORTDLST - Gives an application list of entities to be deleted, DEL_LST returns a system list sorted in user_constituent order in SRT_LST

SORTLSM - Sorts a system list

SRTBYCNT - Create an application list of inclusive constituents in constituent-user order

UPDCRBE - Updates an entry in the CRB

VERAPN - Verifies legality of appending an entity or list of entities (KEY2) to an entity or list of entities (KEY1)

VERCN - Verifies legality of connecting each entity on a list of users to each entity on a list of constituents

VERCR - Verifies legality of creating an entity with the user-supplied entity data block and list of constituents

VERDEL - Verifies legality of deleting an entity

VERGT - Verifies legality of retrieving an entity with the user-supplied entity key

VERUD - Verifies legality of updating an entity with the user-supplied entity key using the user-supplied entity data block and list of constituents

XIEMM - Deletes an entity

XREMM - Deletes the first relation between user and constituent

```
%PAGE
(* %INCLUDE ADCRBM *)
(**)
  PROCEDURE ADCRBM(VAR CRB:CRBPNTR; CONST EKEY:ENTKEY;
      CONST POS:LISTPSTN; CONST DIR:LISTDIR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/07  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                 *)
(*    FUNCTION:                                                    *)
(*       ADD A NEW CRB ENTRY                                       *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*       IBM PASCAL LANGUAGE                                       *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.  *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*       NAME    I/O  DESCRIPTION                                  *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS               *)
(*       EKEY     I   ENTITY CONTAINING THE CONSTITUENT LIST BEING *)
(*                    READ                                         *)
(*       POS      I   LIST POSITION SETTING                        *)
(*       DIR      I   DIRECTION TO READ THE LIST (FORWARD OR REVERSE)*)
(*       RR       O   ERROR CONDITION RETURN CODE                  *)
(*                    = 0  OK RETURN CODE                          *)
(*                    = 1  YOU BLEW IT                             *)
(*                    = 2  THE ROUTINE BLEW IT                     *)
(*                                                                 *)
(*    COMMONS:                                                     *)
(*      COM1                                                       *)
(*        VAR1     I   VAR1 NAME MUST BE FILLED, CHARACTER DATA    *)
(*                     MUST BE PROVIDED                            *)
(*        VAR2     I   VAR2 MUST BE SPECIFIED                      *)
(*      COM2                                                       *)
(*        VAR3     I   CHARACTER DATA MUST BE SPECIFIED            *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH    *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.         *)
(*                                                                 *)
(*    COMMENTS:                                                    *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                  *)
(*                                                                 *)
```

```
(*    CHANGE CONTROL:                                                    *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                               *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                          *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                            *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE       *)
(*           NARRATION ON THE NEXT LINE.                                 *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                  *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                           *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(**)
(* END %INCLUDE ADCRBM *)
```

```
%PAGE
(* %INCLUDE ADRLSM. *)
(**)
  PROCEDURE ADRLSM(CONST INCREMENT:LISTSIZE;CONST POSITION:LISTPSTN;
      CONST KEYE:ENTKEY;VAR LISTREF:LISTPNTR;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       ADD AN ENTITY AFTER A RELATIVE POSITION IN A SYSTEM LIST.    *)
(*       A POSITION OF ZERO INDICATES THE TOP OF THE LIST.  IF THE    *)
(*       LIST REQUIRES EXPANSION TO HOLD THE NEW ENTITY, IT IS        *)
(*       EXPANDED BY INCREMENT ENTRIES.                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        INCREMENT      I   NUMBER OF ENTITIES BY WHICH A LIST IS    *)
(*                           EXPANDED AT A TIME                      *)
(*        POSITION       I   RELATIVE POSITION AFTER WHICH THE NEW    *)
(*                           ENTRY IS ADDED                          *)
(*        KEYE               ENTITY TO BE ADDED.                     *)
(*        LISTREF            POINTER TO THE SYSTEM LIST TO WHICH KEYE *)
(*                           WAS ADDED                               *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ADDS KEYE TO THE SYSTEM LIST LISTREF AT POSITION AFTER THE *)
(*        THE RELATIVE POSITION GIVEN                                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 12/30/85          B. A. ULMER                 FRMI    *)
(*     ADD PROCESSING FOR LARGE LISTS                                *)
(*                                                                   *)
```

```
(*    REVISED: 12/24/84          R. A. MCCLUSKEY              FRMI     *)
(*     ADDED SYSTEM LIST CURRENT LENGTH INDICATOR -- LSTLNM             *)
(*                                                                      *)
(*    REVISED: 10/01/84          E. D. SHREVE                 FRMI     *)
(*     CORRECT THE MOVE OF ENTRIES FROM OLD TO NEW LIST WHEN OLD LIST   *)
(*     MUST BE EXPANDED                                                 *)
(*                                                                      *)
```

```
%PAGE
 (* %INCLUDE ADSCH. *)
 (**)
   PROCEDURE ADSCH(CONST KEYE:ENTKEY;VAR RR:RET_REC);EXTERNAL;
 (**)
 (*-------------------------------------------------------------------*)
 (*                                                                   *)
 (*   $FUNCTION:                                                      *)
 (*       CONNECT AN INTERNAL ITEM TO THE CORRECT PORTION OF THE      *)
 (*       NDS SUPERSTRUCTURE.                                         *)
 (*                                                                   *)
 (*   $DESCRIPTION OF ARGUMENTS:                                      *)
 (*        NAME          I/O  DESCRIPTION                             *)
 (*        ====          ===  ===========                             *)
 (*        SCH_ROOT       I   KEY OF THE SCHEMA_ROOT TO WHICH THE     *)
 (*                           INTERNAL ITEM WILL BE ATTACHED          *)
 (*        KEYE           I   KEY OF THE INTERNAL ITEM TO BE ATTACHED *)
 (*        RR             O   EXTERNAL RETURN CODE-                   *)
 (*                           =0 OK                                   *)
 (*                           >0 CRITICAL ERROR                       *)
 (*                           <0 WARNING                              *)
 (*                                                                   *)
 (*   $COMMONS:                                                       *)
 (*                                                                   *)
 (*   $ENVIRONMENT:                                                   *)
 (*        LANGUAGE: IBM PASCAL                                       *)
 (*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
 (*                                                                   *)
 (*   $EXECUTION PROCEDURE:                                           *)
 (*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
 (*                                                                   *)
 (*   $PROCESSING DESCRIPTION:                                        *)
 (*        DESCRIPTION OF HOW THIS ROUTINE WORKS (INTERNAL ACTIONS)   *)
 (*                                                                   *)
 (*   $COMMENTS:                                                      *)
 (*                                                                   *)
 (*   $CHANGE CONTROL:                                                *)
 (*                                                                   *)
 (*    REVISED: 06/19/86          B. A. ULMER                 FRMI    *)
 (*     CHANGE CALLING PARAMETERS TO CRURUL - NEW DELETE RULES        *)
 (*                                                                   *)
 (*    REVISED: 09/09/85          B. A. ULMER                 FRMI    *)
 (*     ADD TWO NEW PARAMETERS TO FNDURUL                             *)
 (*                                                                   *)
 (*    REVISED: 02/18/85          B. A .ULMER                 FRMI    *)
 (*     CHANGED STRUCTURE OF THE INTERNAL ITEM FOT IMPLEMENTATION OF  *)
 (*     THE CRB                                                       *)
 (*                                                                   *)
 (*    REVISED: 10/04/84          E. D. SHREVE                FRMI    *)
 (*     TO CHANGE LIST INCREMENT WHEN ADDING TO THE INSTANCE COLLECTOR *)
 (*     CONSTITUENT LIST                                              *)
 (*                                                                   *)
```

```
(*     REVISED: 05/14/84          E. D. SHREVE                    FRMI     *)
(*      TO RESET THE SCH_INST 'KIND' TO 'SCH_INST' AFTER THE ENTITY      *)
(*      KIND IS PUT INTO THE STANDARD ARRAY OF THE SCHEMA_ROOT           *)
(*                                                                       *)
```

```
%PAGE
(* %INCLUDE ADSCHR. *)
(**)
  PROCEDURE ADSCHR(CONST KEYE:ENTKEY;VAR POSITION:LISTPSTN;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CONNECT AN INTERNAL ITEM TO THE SCHEMA ROOT.                 *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       SCHEMA PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE      - KEY OF THE INTERNAL ITEM TO BE ATTACHED.      *)
(*       OUTPUT                                                       *)
(*          POSITION  - RELATIVE POSITION OF THIS SCHEMA INSTANCE     *)
(*                      OR CLASS ENTITY IN THE SCHEMA ROOT'S          *)
(*                      CONSTITUENT LIST.                             *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE ADSCHR. *)
```

```
%PAGE
(* %INCLUDE ADTLSM. *)
(**)
  PROCEDURE ADTLSM(CONST INCREMENT:LISTSIZE;CONST KEYE:ENTKEY;
    VAR LISTREF:LISTPNTR;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       ADD AN ENTITY TO A SYSTEM LIST. IF LISTREF IS NIL, THEN          *)
(*       THE LIST IS EMPTY. IF NO ROOM IS AVAILABLE, THEN THE LIST        *)
(*       IS EXPANDED BY INCREMENT ENTITIES.                               *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       LIST PACKAGE.                                                    *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          INCREMENT - THE NUMBER OF ENTITIES BY WHICH A LIST IS         *)
(*                      EXPANDED AT A TIME.                               *)
(*          KEYE      - KEY OF THE ENTITY TO BE ADDED.                    *)
(*          LISTREF   - A POINTER TO A SYSTEM LIST.                       *)
(*       OUTPUT                                                           *)
(*          LISTREF   - POINTER TO THE SYSTEM LIST TO WHICH KEYE          *)
(*                      WAS ADDED.                                        *)
(*          RR        - THE FUNCTION RETURN RECORD.                       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE ADTLSM. *)
```

```
%PAGE
(* %INCLUDE ADTNM *)
(**)
  PROCEDURE ADTNM(CONST KEYE:ENTKEY;VAR KEYL:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    AUTHOR:  UNKNOWN                 CADD    CREATED: YY/MM/DD CC *)
(*    VERSION: MAS VER 2                       REVISED: 84/10/11 CC *)
(*                                                                  *)
(*    FUNCTION:                                                     *)
(*       ADD AN ENTITY TO THE END OF AN APPLICATION LIST.           *)
(*                                                                  *)
(*    ENVIRONMENT:                                                  *)
(*        IBM PASCAL LANGUAGE                                       *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                            *)
(*                                                                  *)
(*    DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME    I/O  DESCRIPTION                                  *)
(*        KEYE     I   KEY OF ENTITY TO BE ADDED.                   *)
(*        KEYL     I   KEY OF THE APPLICATION LIST TO WHICH THE     *)
(*                     ENTITY IS ADDED.                             *)
(*        KEYL     O   THE KEY OF THE LIST WITH THE ENTITY ADDED TO *)
(*                     THE END.                                     *)
(*        RR       O   ERROR CONDITION RETURN CODE.                 *)
(*                     = 0  NORMAL RETURN CODE.                     *)
(*                                                                  *)
(*    COMMONS:                                                      *)
(*                                                                  *)
(*    PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*    COMMENTS:                                                     *)
(*                                                                  *)
(*    CHANGE CONTROL:                                               *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                       *)
(*           UPDATED DOCUMENTATION.                                 *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                         *)
(*           CHANGED DECLARATION OF KEYL TO VAR.                    *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE ADTNM *)
```

THIS PAGE
INTENTIONALLY BLANK

THIS PAGE
INTENTIONALLY BLANK

```
%PAGE
(* %INCLUDE CHKDEL *)
(**)
  PROCEDURE CHKDEL(CONST KEYE:ENTKEY; VAR TEMP_DEL_LIST:LISTPNTR;
      VAR MARK_LIST:LISTKEY; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        CHECK DELETABILITY OF A GIVEN ENTITY BASED ON THE RELATION- *)
(*        SHIP BETWEEN ITS USERS AND ITSELF                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEYE           I   ENTITY WHOSE DELETABILITY IS TO BE     *)
(*                           CHECKED                                 *)
(*        TEMP_DEL_LIST I/O  LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                           ELIGIBLE FOR DELETE                     *)
(*        MARK_LIST     I/O  LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                           MARKED                                  *)
(*        RR             O   RETURN CODE                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE CHKTDEL *)
(**)
  PROCEDURE CHKTDEL(CONST KEYE:ENTKEY; VAR MARK_LIST:LISTKEY;
      VAR TEMP_DEL_LIST:LISTKEY; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        CHECK DELETABILITY OF A GIVEN ENTITY BASED ON THE RELATION- *)
(*        SHIP BETWEEN ITS USERS AND ITSELF                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEYE           I   ENTITY WHOSE DELETABILITY IS TO BE     *)
(*                           CHECKED                                 *)
(*        MARK_LIST     I/O  LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                           TO BE MARKED BY MAED, MAEDI             *)
(*        TEMP_DEL_LIST I/O  LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                           ELIGIBLE FOR DELETE BY MAED, MAEDI      *)
(*        RR             O   RETURN CODE                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE CMPCRB *)
(**)
  PROCEDURE CMPCRB(VAR CRB:CRBPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       COMPREE THE CRB                                             *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O  DESCRIPTION                                   *)
(*       CRB      I/O  CONSTITUENT READ BLOCK ADDRESS                *)
(*       RR        O   ERROR CONDITION RETURN CODE                   *)
(*                     = 0  OK RETURN CODE                           *)
(*                     = 1  YOU BLEW IT                              *)
(*                     = 2  THE ROUTINE BLEW IT                      *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1      I    VAR1 NAME MUST BE FILLED, CHARACTER DATA    *)
(*                       MUST BE PROVIDED                            *)
(*        VAR2      I    VAR2 MUST BE SPECIFIED                      *)
(*      COM2                                                         *)
(*        VAR3      I    CHARACTER DATA MUST BE SPECIFIED            *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                        *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE   *)
(*           NARRATION ON THE NEXT LINE.                             *)
```

```
(*        YY/MM/DD  CCXX  I. M. APERSON                                  *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                           *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(**)
(* END %INCLUDE CMPCRB *)
```

```
%PAGE
(* %INCLUDE CNNODM. *)
(**)
  PROCEDURE CNNODM(CONST KEYEU:ENTKEY;CONST KEYEC:ENTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       CONNECT TWO ENTITIES.                                            *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       ENTITY PACKAGE.                                                  *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          KEYEU     - THE KEY OF THE ENTITY TO BE THE USER.           *)
(*          KEYEC     - THE KEY OF THE ENTITY TO BE THE CONSTITUENT.    *)
(*       OUTPUT                                                           *)
(*          RR        - THE FUNCTION RETURN CODE.                       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE CNNODM. *)
```

```
%PAGE
(* %INCLUDE CNVOSP. *)
(**)
  PROCEDURE CNVOSP(VAR RR:RET_REC;CONST ID:INTEGER;
            CONST THIS_ROUTINE:PGMNAME; VAR RC:EXT_RET_CODE);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CONVERT THE OUT OF CORE SPACE CONDTITION TO A APPLICATION   *)
(*       USER RECOGNIZEABLE FORM                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        RR              I   RETURN RECORD TO BE CONVERTED          *)
(*        ID              I   INTEGER ID OF THE MAS INTERFACE ROUTINE *)
(*                            THAT ISSUED THE RETURN CODE            *)
(*        THIS_ROUTINE    I   CHARACTER REPRESENTATION OF THE INTERFACE*)
(*                            ROUTINE THAT ISSUED THE RETURN CODE    *)
(*        RC              O   EXTERNAL RETURN CODE                    *)
(*                            = 0  OK                                *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        CONVERTS THE OUT OF CORE SPACE CONDITION TO APPLICATION USER*)
(*     RECOQNIXEABLE FORM                                            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE CNVRR. *)
(**)
  PROCEDURE CNVRR(CONST RR:RET_REC;CONST PGM_ID:INTEGER;
              CONST PGM_NAME: PGMNAME; VAR RC:EXT_RET_CODE);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*        GET THE EXTERNAL RETURN CODE CORRESPONDING TO THE INTERNAL *)
(*        FORMAT.                                                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                          *)
(*         ====          ===  ===========                          *)
(*         RR             I   RETURN RECORD TO BE CONVERTED        *)
(*         PGM_ID         I   INTEGER ID OF THE MAS INTERFACE ROUTINE *)
(*                            THAT ISSUED THE RETURN CODE          *)
(*         RC             O   EXTERNAL RETURN CODE                 *)
(*                            = 0  OK                              *)
(*                            > 0  CRITICAL ERROR                  *)
(*                            < 0  WARNING                         *)
(*                                                                 *)
(*    $COMMONS:                                                    *)
(*                                                                 *)
(*    $ENVIRONMENT:                                                *)
(*         LANGUAGE: IBM PASCAL                                    *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/438?                  *)
(*                                                                 *)
(*    $EXECUTION PROCEDURE:                                        *)
(*         INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE        *)
(*                                                                 *)
(*    $PROCESSING DESCRIPTION:                                     *)
(*         CONVERTS THE INTERNAL RETURN CODE TO EXTERNAL RETURN CODE *)
(*                                                                 *)
(*    $COMMENTS:                                                   *)
(*                                                                 *)
(*    $CHANGE CONTROL:                                             *)
(*                                                                 *)
(*     REVISED: 85/07/11          B. A. ULMER                FRMI  *)
(*      CHANGED TO ADD ERROR MESSAGE AND PROGRAM NAME TO MSTATUS COMMON*)
(*      WHEN AN INTERFACE GETS A NONE ZERO RETURN CODE             *)
(*                                                                 *)
```

```
%PAGE
(* %INCLUDE CPYAUDB *)
(**)
  PROCEDURE CPYAUDB(VAR ENTBPNTR:ENTPNTR;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  UNKNOWN                CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 2                      REVISED: 84/10/11 CC   *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       STORE THE VALUE OF AN APPLICATION ENTITY BLOCK IN AN        *)
(*       UNINITIALIZED SYSTEM UDB.                                   *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                             *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME     I/O  DESCRIPTION                                  *)
(*        ENTDEF    I   ENTBLOCK CONTAINING THE VALUES TO STORE.     *)
(*        ENTBPNTR  O   POINTER TO ENTBLOCK CREATED.                 *)
(*        RR        O   ERROR CONDITION RETURN CODE.                 *)
(*                      = 0  NORMAL RETURN CODE.                     *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       CPYAUDB USES AMPXMOVE A SYSTEM ROUTINE. AMPXMOVE MOVES      *)
(*       DATA FROM MEMORY TO MEMORY (THE NUMBER OF BYTES TO MOVE     *)
(*       MUST BE SPECIFIED).                                         *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                        *)
(*           UPDATED DOCUMENTATION.                                  *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED DECLARATION OF ENTDEF TO VAR.                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE CPYAUDB *)
```

```
%PAGE
(* %INCLUDE CPYCST. *)
(**)
  PROCEDURE CPYCST(CONST SCH_KEY   : ENTKEY;
                   VAR   KEY1      : LISTKEY;
                   VAR   LIST_LENG : LISTSIZE;
                   VAR   RR        : RET_REC);  EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       ADD THE ENTITIES IN A CONSTITUENT LIST INTO A LIST.        *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL                                                     *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          SCH-KEY   - KEY OF A CLASS OR ENTITY COLLECTOR.         *)
(*          KEY1      - KEY OF THE LIST ONTO WHICH THE ENTITIES     *)
(*                      WILL BE ADDED.                              *)
(*       OUTPUT                                                     *)
(*          LIST_LENG - TOTAL LENGTH OF ALL CNST ADDED TO LIST.     *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*    METHOD                                                        *)
(*       IF SCH_KEY IS AN ENTITY COLLECTOR, THEN ALL CONSTITUENTS   *)
(*       ARE ADDED TO THE LIST.  IF SCH_KEY IS A CLASS COLLECTOR,   *)
(*       'CPYCST' IS CALLED RECURSIVELY TO PROCESS THE ENTITY       *)
(*       COLLECTORS THAT ARE CONSTITUENTS OF SCH_KEY.  LIST_LENG IS *)
(*       ACCUMULATED FOR ALL RECURSIONS.                            *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE CPYCST *)
```

```
%PAGE
(* %INCLUDE CPYLSM. *)
(**)
  PROCEDURE CPYLSM(CONST LISTFROM : LISTPNTR;
                   VAR   POSITION : LISTPSTN;
                   VAR   LISTTO   : LISTPNTR;
                   VAR   RR       : RET_REC);  EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       COPY THE NON-VACANT ELEMENTS OF LISTFROM TO LISTTO.  IF          *)
(*       LISTTO WAS INITIALIZED,  IT IS DELETED PRIOR TO COPYING.         *)
(*       POSITION IS SET TO THE BEGINNING OF LISTTO.  CURRENT LENGTH      *)
(*       OF OF LISTTO IS SET TO CURRENT LENGTH OF LISTFROM.               *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*      LIST PACKAGE.                                                     *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          LISTFROM  - POINTER TO SYSTEM LIST TO BE COPIED.              *)
(*       OUTPUT                                                           *)
(*          LISTTO    - POINTER TO SYSTEM LIST TO WHICH COPY IS MADE.     *)
(*          POSITION  - SET TO INDICATE THE BEGINNING OF LISTTO.          *)
(*          RR        - THE FUNCTION RETURN RECORD.                       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE CPYLSM. *)
```

```
%PAGE
(* %INCLUDE CPYNM. *)
(**)
  PROCEDURE CPYNM(CONST KEYL:LISTKEY;VAR KEYLOUT:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       CREATE A NEW LIST WHICH CONTAINS A COPY OF THE ENTITIES     *)
(*       REFERENCED BY KEYL.                                         *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          KEYL       - KEY OF THE LIST TO BE COPIED.              *)
(*       OUTPUT                                                      *)
(*          KEYLOUT    - KEY OF THE NEW LIST WHICH IS A COPY OF THE *)
(*                       INPUT LIST.                                 *)
(*          RR         - THE FUNCTION RETURN RECORD.                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE CPYNM. *)
```

```
%PAGE
(* %INCLUDE CRCLST. *)
(**)
  PROCEDURE CRCLST(CONST KEYE:ENTKEY;CONST LISTREF:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CREATE RELATIONS BETWEEN A USER ENTITY AND A LIST OF         *)
(*       CONSTITUENTS.                                                *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE       - KEY OF THE USER ENTITY OF THE RELATIONS.     *)
(*          LISTREF    - POINTER TO SYSTEM LIST OF CONSTITUENTS.      *)
(*       OUTPUT                                                       *)
(*          RR         - THE FUNCTION RETURN RECORD.                  *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE CRCLST. *)
```

```
%PAGE
(* %INCLUDE CRCNM. *)
(**)
  PROCEDURE CRCNM(CONST KEYE:ENTKEY;CONST KEYL:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CREATE RELATIONS BETWEEN A USER ENTITY AND A LIST OF         *)
(*       CONSTITUENTS.                                                *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE      - KEY OF THE USER ENTITY OF THE RELATIONS.      *)
(*          KEYL      - KEY OF LIST OF CONSTITUENT ENTITIES.          *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE CRCNM. *)
```

```
%PAGE
(* %INCLUDE CRDLST *)
(**)
  PROCEDURE CRDLST(CONST KEY1:ANYKEY;VAR CNSTS_SRTLST:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        CREATE A SORTED INCLUSIVE  LIST OF AN ENTITY OR A LIST OF  *)
(*        ENTITIES AND THEIR DIRECTAND INDIRECT CONSTITUENTS.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        KEY1            I   AN ENTITY OR LIST OF ENTITIES TO BE PUT *)
(*                            ON A LIST WITH THEIR CONSTITUENTS.     *)
(*        CNSTS_SRTLST    O   AN INCLUSIVE LIST OF AN ENTITY OR LIST *)
(*                            AND THIER DIRECT AND INDIRECT CONSTS. IN *)
(*                            USER-CONSTITUENT ORDER.                *)
(*        RC              O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK RETURN CODE                    *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        IF KEY1 IS AN ENTKEY THEN                                  *)
(*           AN INCLUSIVE LIST OF THE ENTITIES CONSTITUENTS IS BUILT  *)
(*           INCLUDING KEY1.                                         *)
(*        IF KEY1 IS A LIST THEN                                     *)
(*           A LIST OF THE INCLUSIVE CONSTITUENTS OF THE ENTITIES ON  *)
(*           KEY1 IS CREATED, INCLUDING THE ENTITIES ON KEY1.        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        THE OUTPUT LIST IS SORTED IN USER-CONSTITUENT ORDER.       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 04/26/85          E. D. SHREVE              W315     *)
(*     CHANGED TO USE THE INTER PROCESS FLAG (MAPROB).               *)
(*                                                                   *)
```

```
(*    REVISED: 02/18/85          B. A. ULMER                 W315     *)
(*     CHANGED TO IMPLEMENT THE CONST. READ BLOCK.                    *)
(*                                                                    *)
(*    REVISED: 11/01/84          E. D. SHREVE                W315     *)
(*     REMOVE CALL TO DISPLSM                                         *)
(*                                                                    *)
(*    ORIGINATED: 08/23/84       C. J. SAMPLE                W315     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE PROG_ID *)
```

```
%PAGE
(* %INCLUDE CREMM. *)
(**)
  PROCEDURE CREMM(CONST KEYEU:ENTKEY;CONST KEYEC:ENTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       CREATE A USER-CONSTITUENT RELATION BETWEEN ENTITIES.        *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       ENTITY PACKAGE.                                             *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          KEYEU     - KEY OF ENTITY TO BE THE USER.              *)
(*          KEYEC     - KEY OF ENTITY TO BE THE CONSTITUENT.       *)
(*       OUTPUT                                                      *)
(*          RR        - THE FUNCTION RETURN RECORD.                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE CREMM. *)
```

```
%PAGE
(* %INCLUDE CRURUL. *)
(**)
  PROCEDURE CRURUL(CONST ENTITY_TYPE:ORD_KIND;VAR GROUP:T_GROUP_ARRAY;
      VAR NUM_GROUP:LISTPSTN; VAR MIN_CNST:LISTPSTN);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATES THE USER'S RULES.  RULES OF CONNECTIVITY USED TO    *)
(*       DETERMINE DELETABILITY OF ENTITIES.                         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        ENTITY_TYPE     I   ENTITY KIND VALUE WHICH WILL HAVE THE  *)
(*                            DELETE RULE                            *)
(*        GROUP           O   ARRAY THAT WILL BE FILLED WITH THE RULES *)
(*                            AND NUMBER OF CONSTITUENTS OF EACH     *)
(*                            DIFFERENT RELATIONSHIP THAT THIS ENTITY *)
(*                            KIND CAN HAVE WITH ITS CONSTITUENTS    *)
(*        NUM_GROUP       O   NUMBER OF DIFFERENT RELATIONSHIPS THIS *)
(*                            ENTITY CAN HAVE WITH ITS CONSTITUENTS  *)
(*        MIN_CNST        O   MINIMUM NUMBER OF CONSTITUENTS THAT THIS *)
(*                            ENTITY CAN HAVE WHEN IT HAS A GROUP OF *)
(*                            CONSTITUENTS THAT ARE "SECONDARY"      *)
(*        RC              O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK RETURN CODE                    *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       ??????ARE SET TO INDICATE IF THE RELATIONSHIP BETWEEN THE   *)
(*       USER AND ITS CONSTITUTES IS DEPENDENT OR INDEPENDENT AND    *)
(*       STRONG OR WEAK.                                             *)
(*       DEFAULT RULE IS DEPENDENT/STRONG.                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
```

```
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 06/19/86          B. A. ULMER                  FRMI      *)
(*     REDO LOGIC OF HOW CRURUL WORKS BASED ON THE NEW DELETE RULES     *)
(*                                                                       *)
(*    REVISED: 09/  /85          B. A. ULMER                  FRMI      *)
(*     ADD ENTITY KINDS SO AS TO TEST THE NEW DELETE RULES (2070,       *)
(*     2080, 2090)                                                      *)
(*                                                                       *)
(*    REVISED: 09/  /85          B. A. ULMER                  FRMI      *)
(*     ADD PARAMETERS TO HANDLE THE TWO NEW DELETE RULES                *)
(*                                                                       *)
(*    REVISED: 09/18/84          D. J. KERCHNER               FRMI      *)
(*     ADDED I/S RULE FOR THE PICK ENTITY                               *)
(*                                                                       *)
```

```
%PAGE
(* %INCLUDE DELALNL *)
(**)
  PROCEDURE DELALNL(VAR STACK_KEY:LISTPNTR;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       DISPOSE OF ALL APPLICATION LISTS IN THE WORKING FORM MODEL. *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL                                                      *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          STACK_KEY - STACK_OF_LISTS.                              *)
(*       OUTPUT                                                      *)
(*          RC        - THE FUNCTION RETURN CODE.                    *)
(*                                                                   *)
(*    METHOD                                                         *)
(*       EACH LIST IN THE STACK_OF_LISTS IS PROCESSED. EACH          *)
(*       APPLICATION LIST ON THESE LISTS IS DISPOSED.                *)
(*                                                                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE DELALNL *)
```

```
%PAGE
(* %INCLUDE DELCNST *)
(**)
  PROCEDURE DELCNST(CONST KEYE:ENTKEY; VAR TEMP_DEL_LIST:LISTPNTR;
      VAR MARK_LIST:LISTKEY; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DETERMINES THE DELETABILITY OF GIVEN ENTITY'S CONSTITUENTS  *)
(*        BASED ON THE RELATIONSHIP THE CONSTITUENT HAS WITH ITS USERS*)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                          *)
(*        ====           ===  ===========                          *)
(*        KEYE            I   ENTITY WHOSE CONSTITUENTS WILL HAVE THEIR*)
(*                            DELETABILITY DETERMINED               *)
(*        TEMP_DEL_LIST I/O   LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                            ELIGIBLE FOR DELETE                   *)
(*        MARK_LIST     I/O   LIST WHICH CONTAINS ENTITIES THAT ARE  *)
(*                            MARKED                                *)
(*        RR             O    RETURN CODE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE DELCRBE *)
(**)
  PROCEDURE DELCRBE(VAR CRB:CRBPNTR; CONST EKEY:ENTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       DELETE A CRB ENTRY                                          *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME    I/O  DESCRIPTION                                    *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS                 *)
(*       EKEY     I   ENTITY KEY OF ENTRY TO BE DELETED              *)
(*       RR       O   ERROR CONDITION RETURN CODE                    *)
(*                    = 0  OK RETURN CODE                            *)
(*                    = 1  YOU BLEW IT                               *)
(*                    = 2  THE ROUTINE BLEW IT                       *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1      I   VAR1 NAME MUST BE FILLED, CHARACTER DATA     *)
(*                      MUST BE PROVIDED                             *)
(*        VAR2      I   VAR2 MUST BE SPECIFIED                       *)
(*      COM2                                                         *)
(*        VAR3      I   CHARACTER DATA MUST BE SPECIFIED             *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
```

```
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                          *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE     *)
(*           NARRATION ON THE NEXT LINE.                               *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                 *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                         *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE DELCRBE *)
```

```
%PAGE
(* %INCLUDE DELEMM. *)
(**)
  PROCEDURE DELEMM(VAR KEYE:ENTKEY;VAR RR:RET_REC); EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       DELETE ALL REFERENCES TO THIS ENTITY FROM ALL APPLICATION    *)
(*       LISTS AND DISPOSE OF THE ENTITY.  TO COMPLETE DELETE ACTION  *)
(*       REQUIRES BREAKING ALL USER AND CONSTITUENT CONNECTIONS.      *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       ENTITY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE       - KEY OF THE ENTITY TO BE DELETED.            *)
(*       OUTPUT                                                       *)
(*          RR         - THE FUNCTION RETURN RECORD.                 *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       AN ENTRY IN AN APPLICATION LIST HAS A FORM OF INT_ITEM.      *)
(*       ALL REFERENCES TO IT WILL BE DELETED. THE USER WILL NEVER    *)
(*       DIRECTLY DELETE ENTITIES OF FORM INT_ROOT. THESE ARE ONLY    *)
(*       DELETED AS A RESULT OF THE CLEANUP ASSOCIATED WITH THE       *)
(*       DELETION OF AN NDS.                                          *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE DELEMM. *)
```

```
%PAGE
(* %INCLUDE DELENTY. *)
(**)
  PROCEDURE DELENTY(VAR KEYE:ENTKEY;VAR DEL_LST:LISTPNTR;
     VAR POSITION:INTEGER;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       DELETE THE ENTITY.                                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   ENTITY TO BE DELETED                    *)
(*        DEL_LST        O   LIST OF ENTITIES KEYS ELIGIBLE FOR DELETE*)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE -                   *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE ENTITY'S CONSTITUENTS LIST IS READ AND IF A CONSTITUENT  *)
(*       IS MARKED FOR DELETION, THEN DELRUL IS CALLED TO DETERMINE   *)
(*       IF THE CONSTITUENT MAY ALSO BE DELETED.  AFTER ALL           *)
(*       CONSTITUENTS ARE READ, THE ENTITY IS DELETED (REMOVED FROM   *)
(*       USERS AND CNSTS LISTS AND DISPOSED OF.)                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 09/05/85          B. A. ULMER                 FRMI    *)
(*     ADDED CODE TO HANDLE THE TWO NEW DELETE RULES                 *)
(*                                                                   *)
(*    REVISED: 02/18/85          B. A. ULMER                 FRMI    *)
(*     CHANGED THE STRUCTURE OF THE INTERNAL ITEM FOR IMPLEMENTATION *)
(*     OF THE CRB                                                    *)
(*                                                                   *)
(*    ORIGINATED: 06/21/84       C. J. SAMPLE                FRMI    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
%PAGE                                                                  *)
(*---------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                 *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END------------------------------------------------------------------*)
(**)
(* END %INCLUDE DELENTY *)
```

```
%PAGE
(* %INCLUDE DELPLST. *)
(**)
  PROCEDURE DELPLST(CONST INCREMENT:LISTSIZE;CONST IPOS:LISTINDX;
      VAR POSITION:LISTPSTN;VAR LISTREF:LISTPNTR;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*       REMOVE AN ENTITY FROM A SPECIFIED POSITION IN A SYSTEM LIST       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        INCREMEN       I   NUMBER OF ENTITIES  BY WHICH SYSTEM LIST     *)
(*                           LIST IS EXPANDED OR REDUCED                   *)
(*        IPOS           I   POSITION IN THE LIST FROM WHICH THE          *)
(*                           ENTITY WILL BE REMOVED                        *)
(*        POSITION      I/O  LAST LOCATION ON THE SYSTEM LIST THAT WAS     *)
(*                           PROCESSED                                     *)
(*        LISTREF        I   POINTER TO SYSTEM LIST FROM WHICH ENTITY      *)
(*                           WILL BE REMOVED                               *)
(*        RC             O   EXTERNAL RETURN CODE                         *)
(*                           = 0  OK RETURN CODE                           *)
(*                           = 1  YOU BLEW IT                              *)
(*                           = 2  THE ROUTINE BLEW IT                      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE                 *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*       SHIFT ALL FOLLOWING ENTITIES UP UNTIL ALL VACANT ENTITIES         *)
(*       ARE AT THE END OF THE LIST.  RECALCULATE THE POSITION IF          *)
(*       IT WAS AFFECTED BY THIS REMOVAL.  IF MORE THAN INCREMENT          *)
(*       ENTITIES ARE VACANT, THEN COMPRESS THE LIST BY REMOVING           *)
(*       THE INCREMENT ENTITIES.                                           *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: 12/30/85          B. A. ULMER                  FRMI        *)
(*     ADD PROCESSING FOR LARGE LISTS                                      *)
(*                                                                         *)
```

```
(*     REVISED: 02/06/85          E. D. SHREVE                FRMI     *)
(*      TEST FOR NIL POINTER                                           *)
(*                                                                     *)
(*     REVISED: 12/24/84          R. A. MCCLUSKEY             FRMI     *)
(*      ADDED SYSTEM LIST CURRENT LENGTH INDICATOR -- LSTLNM           *)
(*                                                                     *)
```

```
%PAGE
(* %INCLUDE DELPNLA. *)
(**)
  PROCEDURE DELPNLA(VAR POSITION:LISTPSTN; VAR LISTA:LISTPNTR;
                    VAR RR:RET_REC); EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION                                                      *)
(*       DELETE ALL APPL LISTS AFTER A SPECIFIED POSITION IN THE     *)
(*       LIST_OF_LISTS EXCEPT THOSE THAT ARE 'LOCKED'.               *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS                                      *)
(*       NAME          I/O       DESCRIPTION                       *)
(*       ====          ===       ===========                       *)
(*       POSITION       I        POSITION IN LISTA TO START DELETE.  *)
(*       LISTA          I        LIST_OF_LISTS SYSTEM LIST           *)
(*       RR             O        RETURN CODE     -                   *)
(*                               =0  GOOD RETURN                     *)
(*                               >0  CRITICAL ERROR                  *)
(*                               <0  WARNING                         *)
(*                                                                   *)
(*    $COMMONS                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*        LANGUAGE:  IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*        INTERNAL PROCEDURE OF THE MODEL ACCESS SOFTWARE            *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*        STARTING WITH THE INPUT POSITION, EACH APPL LIST ON THE    *)
(*        INPUT LIST_OF_LISTS (LISTA) IS PROCESSED.  IF THE LIST     *)
(*        IS 'LOCKED' (DELTFLG = NODEL), THE LISTKEY IS PLACED ON A  *)
(*        TEMPORARY LIST; ELSE, THE LIST IS DELETED.  AFTER ALL      *)
(*        ENTRIES ARE PROCESSED, THE TEMPORARY LIST IS MERGED WITH   *)
(*        ANY ENTRIES STILL REMAINING ON LISTA.                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*        ORIGINATED:  04/23/85    E. D. SHREVE      W315           *)
(*-------------------------------------------------------------------*)
(**)
(*END %INCLUDE DELPNLA. *)
```

```
%PAGE
(* %INCLUDE DELRLSM. *)
(**)
  PROCEDURE DELRLSM(CONST INCREMENT:LISTSIZE;CONST KEYE:ENTKEY;
      VAR POSITION:LISTPSTN;VAR LISTREF:LISTPNTR;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REMOVE AN ENTITY FROM A SYSTEM LIST.                        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        INCREMENT      I   NUMBER OF ENTITIES BY WHICH A SYSTEM LIST*)
(*                           LIST IS EXPANDED OR REDUCED             *)
(*        KEYE           I   KEY OF THE ENTITY TO BE REMOVED FROM THE *)
(*                           LIST                                    *)
(*        POSITION      I/O  LOCATION ON THE SYSTEM LIST OF ENITY     *)
(*                           TO BE PROCESSED  --  UPDATED LOCATION OF *)
(*                           ENTITY ORIGINALLY INDICATED BY POSITION  *)
(*        LISTREF        I   POINTER TO SYSTEM LIST FROM WHICH ENTITY *)
(*                           WILL BE REMOVED                         *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       SHIFT ALL FOLLOWING ENTITIES UP UNTIL ALL VACANT ENTITIES   *)
(*       ARE AT THE END OF THE LIST.  RECALCULATE THE POSITION IF    *)
(*       IT WAS AFFECTED BY THIS REMOVAL.  IF MORE THAN INCREMENT    *)
(*       ENTITIES ARE VACANT, THEN COMPRESS THE LIST BY REMOVING     *)
(*       THE INCREMENT ENTITIES.                                     *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 12/30/85          B. A. ULMER                 FRMI      *)
(*     ADD PROCESSING FOR LARGE LISTS                                    *)
(*                                                                       *)
(*    REVISED: 12/24/84          R. A. MCCLUSKEY             FRMI      *)
(*     ADDED SYSTEM LIST CURRENT LIST INDICATOR -- LSTLNM                *)
(*                                                                       *)
```

```
%PAGE
(* %INCLUDE DELRUL *)
(**)
  PROCEDURE DELRUL(VAR KEYE:ENTKEY;VAR DEL_LIST:LISTPNTR;VAR
      MARK_LIST:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       DELETE AN ENTITY ACCORDING TO THE DELETE RULES.             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEYE           I   ENTITY TO BE DELETED OR MARKED FOR     *)
(*                           DELETION                                *)
(*        DEL_LST        I   LIST OF KEYS THAT ARE ELIGIBLE FOR     *)
(*                           DELETION                                *)
(*        MARK_LIST      O   LIST OF ENTITIES WHICH HAVE BEEN MARKED *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       ??????? TY'S USER LIST IS READ AND THE DELETE RULES FOR     *)
(*       EACH USER ARE CHECKED TO DETERMINE IF ENTITY CAN BE DELETED. *)
(*       IF UNABLE TO DELETE THE ENTITY THEN CHECK IF THE USER IS ON  *)
(*       THE DELETE LIST.  IF ON THE LIST THEN DELETE THE ENTITY ELSE *)
(*       MARK IT FOR DELETE.  IF UNABLE TO MARK FOR DELETE THEN ADD   *)
(*       ENTITY TO THE EXCEPTION LIST.                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       THE DELETE RULES ARE STORED IN THE INSTANCE COLLECTOR OF AN  *)
(*       ENTITY'S USER AS DEPENDENCE AND STRENGTH FLAGS.  DEPENDENCE  *)
(*       IS DEFINED AS DEPENDENT (TRUE) OR INDEPENDENT (FALSE).      *)
(*       STRENGTH IS DEFINED AS DEPENDENT (TRUE) OR INDEPENDENT      *)
(*       (FALSE).                                                    *)
(*       IF THERE EXISTS A DEPENDENT/STRONG USER CONNECTION, THEN    *)
(*         THE ENTITY MAY NOT BE DELETED.                            *)
```

```
(*       IF THERE EXISTS A DEPENDENT/WEAK USER CONNECTION, BUT NO     *)
(*         DEPENDENT/STRONG CONNECTION THEN THE ENTITY IS MARKED FOR  *)
(*         DELETION AND IF ANY OF ITS USER CONNECTIONS WERE           *)
(*         INDEPENDENT/WEAK, THEN IT IS DISCONNECTED FROM THOSE       *)
(*         INDEPENDENT/WEAK USER CONNECTIONS.                         *)
(*       IF THERE ARE NO DEPENDENT/STRONG NOR DEPENDENT/WEAK          *)
(*         USER CONNECTIONS OR NO USERS AT ALL, THEN THE ENTITY IS    *)
(*         DELETED AND ITS CONSTITUENTS ARE PROCESSED THE SAME AS     *)
(*         THE ENTITY WAS.                                            *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
(*     REVISED: 09/02/86          B. A. ULMER                 DBMA    *)
(*      REMOVE DUPLICATE ENTITTES FROM DELETE LIST - CAUSES A PROBLEM *)
(*        WHEN AN ENTITY HAS THE SAME CNST TWICE                      *)
(*                                                                    *)
(*     REVISED: 06/19/86          B. A. ULMER                 FRMI    *)
(*      MAJOR REWRITE DUE TO NEW DELETE RULES                         *)
(*                                                                    *)
(*     REVISED: 12/17/85          B. A. ULMER                 FRMI    *)
(*      FIX PROBLEM WITH CODE FOR NEW DELETE RULES                    *)
(*                                                                    *)
(*     REVISED: 09/05/85          B. A. ULMER                 FRMI    *)
(*      ADD CODE TO HANDLE THE TWO NEW DELETE RULES                   *)
(*                                                                    *)
(*     ORIGINATED: 06/15/84       C. J. SAMPLE                FRMI    *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE DELRUL. *)
```

```
%PAGE
(* %INCLUDE DELSCH. *)
(**)
  PROCEDURE DELSCH(CONST KEYE:ENTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       DISCONNECT AN INTERNAL ITEM FROM THE CORRECT PORTION OF    *)
(*       THE NDS SUPERSTRUCTURE.                                    *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       SCHEMA PACKAGE.                                            *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          KEYE      - KEY OF THE INTERNAL ITEM TO BE DETACHED.    *)
(*       OUTPUT                                                     *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*    CHANGE CONTROL                                                *)
(*      CHANGED: 12/14/84  E. SHREVE  - TO DELETE THE INSTANCE      *)
(*               COLLECTOR IF ALL IT'S CNSTS ARE DELETED.           *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE DELSCH. *)
```

```
%PAGE
(* %INCLUDE DELTLSM. *)
(**)
  PROCEDURE DELTLSM(CONST INCREMENT:LISTSIZE;VAR LISTREF:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REMOVES THE LAST NON-VACANT ENTITY REFERENCE IN A LIST.     *)
(*       IF THIS REMOVAL PRODUCES MORE THAN INCREMENT VACANT         *)
(*       ENTITIES AT THE BOTTOM OF THE LIST, THEN THE VACANT         *)
(*       ENTITIES ARE ELIMINATED.                                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        LISTREF        I   LIST WHOSE LAST ENTITY IS TO BE REMOVED  *)
(*        INCREMENT      I   MAXIMUM NUMBER OF VACENT ENTITIES THE    *)
(*                           LAST MIGHT CONTAIN                      *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 12/30/85          B. A. ULMER                 FRMI    *)
(*     ADD PROCESSING FOR LARGE LISTS                                *)
(*                                                                   *)
(*    REVISED: 12/24/84          R. A. MCCLUSKEY             FRMI    *)
(*     ADDED SYSTEM LIST CURRENT LENGTH INDICATOR -- LSTLNM          *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE DETCLST. *)
(**)
  PROCEDURE DETCLST(CONST KEYE:ENTKEY;VAR DLIST:LISTKEY;
    VAR MLIST:LISTKEY;VAR ELIST:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CHECK THE CONSTITUENTS OF A MARKED ENTITY OR AN ENTITY ON   *)
(*       THE MARKLIST TO DETERMINE IF THE CONSTITUENTS MAY ALSO BE   *)
(*       TESTED FOR DELETION.                                        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYES          I   ENTITY WHOSE CONSTITUENTS ARE TO BE     *)
(*                           TESTED FOR DELETION -                   *)
(*        DLIST         I/O  ENTITIES THAT MAY BE DELETED            *)
(*        MLIST         I/O  ENTITIES THAT MAY BE MARKED FOR DELETION*)
(*        ELIST         I/O  ENTITIES THAT MAY NOT BE MARKED FOR     *)
(*                           DELETION NOR DELETED                    *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       DETCLST IS CALLED AFTER IT HAS BEEN DETERMINED THAT THE     *)
(*       KEYE IS MARKED FOR DELETION OR IS ON THE MARK FOR DELETION  *)
(*       LIST.  THE KEYE'S CONSTITUENTS ARE CHECKED. IF MARKED FOR   *)
(*       DELETION THEN TEST THEM FOR DELETION.                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 09/06/85          B. A. ULMER                 FRMI    *)
(*     ADDED CODE TO HANDLE THE TWO NEW DELETE RULES                 *)
(*                                                                   *)
```

```
(*    REVISED: 02/18/85          B. A. ULMER                  FRMI     *)
(*     CHANGED THE STRUCTURE OF THE INTERNAL ITEM FOR IMPLEMENTATION  *)
(*     OF THE CRB                                                     *)
(*                                                                    *)
(*    ORIGINATED: 07/10/84       C. J. SAMPLE                 FRMI     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE DETCLST. *)
```

```
%PAGE
(* %INCLUDE DETCNST *)
(**)
  PROCEDURE DETCNST(CONST KEYE:ENTKEY; VAR MARK_LIST:LISTKEY;
      VAR TEMP_DEL_LIST:LISTKEY; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DETERMINES THE DELETABILITY OF GIVEN ENTITY'S CONSTITUENTS  *)
(*        BASED ON THE RELATIONSHIP THE CONSTITUENT HAS WITH ITS USERS*)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        KEYE            I   ENTITY WHOSE CONSTITUENTS WILL HAVE THEIR*)
(*                            DELETABILITY DETERMINED                *)
(*        MARK_LIST      I/O  LIST WHICH CONTAINS ENTITIES THAT ARE    *)
(*                            MARKED                                 *)
(*        TEMP_DEL_LIST  I/O  LIST WHICH CONTAINS ENTITIES THAT ARE    *)
(*                            ELIGIBLE FOR DELETE                    *)
(*        RR              O   RETURN CODE                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE DETRUL *)
(**)
  PROCEDURE DETRUL(CONST KEYE:ENTKEY;VAR MARK_LIST:LISTKEY;
      VAR DEL_LIST:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       TEST DELETE OF AN ENTITY ACCORDING TO THE DELETE RULES.     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   ENTITY TO TESTED FOR DELETION OR MARK FOR*)
(*                           DELETION                                *)
(*        MLIST         I/O  LIST OF ENTITIES WHICH MAY BE MARKED    *)
(*        DLIST         I/O  LIST OF ENTITIES WHICH MAY BE DELETED   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       ????? TITY'S USERS LIST IS READ AND THE DELETE RULES FOR    *)
(*       EACH USER ARE CHECKED TO DETERMINE IF THE ENTITY CAN BE     *)
(*       DELETED.                                                    *)
(*       THE DELETE RULES ARE STORED IN THE INSTANCE COLLECTOR OF AN *)
(*       ENTITY'S USER AS DEPENDENCE AND STRENGTH FLAGS.  DEPENDENCE *)
(*       IS DEFINED AS DEPENDENT (TRUE) OR INDEPENDENT (FALSE).      *)
(*       STRENGTH IS DEFINED AS DEPENDENT (TRUE) OR INDEPENDENT      *)
(*       (FALSE).                                                    *)
(*       IF THERE EXISTS A DEPENDENT/STRONG USER CONNECTION, THEN    *)
(*         THE ENTITY MAY NOT BE DELETED AND IT IS ADDED TO THE      *)
(*         EXCEPTION LIST.                                           *)
(*       IF THERE EXISTS A DEPENDENT/WEAK USER CONNECTION, BUT NO    *)
(*         DEPENDENT/STRONG CONNECTION THEN THE ENTITY CAN BE        *)
(*         MARKED FOR DELETION AND ADDED TO THE MARK LIST.           *)
(*       IF THERE ARE NO DEPENDENT/STRONG USER CONNECTIONS,          *)
(*                    NO DEPENDENT/WEAK USER CONNECTIONS,            *)
(*                    NO USERS AT ALL, OR                            *)
(*                    ALL USERS ARE ON THE DELETE LIST,              *)
```

```
(*       THEN                                                          *)
(*         THE ENTITY IS DELETABLE AND ADDED TO THE DELETE LIST.       *)
(*         IF THE ENTITY IS MARKED FOR DELETION OR                     *)
(*                       IS ON THE MARK LIST,                          *)
(*         THEN                                                        *)
(*           ITS CONSTITUENTS ARE PROCESSED THE SAME AS THE ENTITY.    *)
(*                                                                     *)
(*    $COMMENTS:                                                       *)
(*                                                                     *)
(*    $CHANGE CONTROL:                                                 *)
(*                                                                     *)
(*     REVISED: 06/19/86          B. A. ULMER                 FRMI     *)
(*      MAJOR REWRITE DUE TO THE NEW DELETE RULES                      *)
(*                                                                     *)
(*     REVISED: 04/18/86          E. D. SHREVE                FRMI     *)
(*      TO SET DELETE RULES ONLY WHEN USER IS NOT IN DELIST            *)
(*                                                                     *)
(*     REVISED: 09/06/85          B. A. ULMER                 FRMI     *)
(*      ADDED CODE TO HANDLE THE TWO NEW DELETE RULES                  *)
(*                                                                     *)
(*     ORIGINATED: 06/28/84       C. J. SAMPLE                FRMI     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
%PAGE                                                                  *)
(*---------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                 *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE DETRUL. *)
```

```
%PAGE
(* %INCLUDE DIFLSM. *)
(**)
  PROCEDURE DIFLSM(CONST LIST1:LISTPNTR;CONST LIST2:LISTPNTR;
      VAR POSITION:LISTPSTN;VAR LISTOUT:LISTPNTR;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*-------------------------------------------  ------------------*)
(*                                                                *)
(*    FUNCTION                                                    *)
(*       CREATE A SYSTEMS LIST CONSISTING OF ALL ENTITIES IN LIST1*)
(*       THAT ARE NOT IN LIST2.                                   *)
(*                                                                *)
(*    LANGUAGE                                                    *)
(*       PASCAL.                                                  *)
(*                                                                *)
(*    PACKAGE                                                     *)
(*       LIST PACKAGE.                                            *)
(*                                                                *)
(*    ARGUMENTS                                                   *)
(*       INPUT                                                    *)
(*          LIST1, LIST2 - THE LISTS WHOSE DIFFERENCE IS TO BE FOUND.*)
(*       OUTPUT                                                   *)
(*          LISTOUT   - LIST CONTAINING THE DIFFERENCE OF THE     *)
(*                      TWO LISTS.                                *)
(*          POSITION  - INTEGER INDICATING BEGINNING OF LISTOUT.  *)
(*          RR        - THE FUNCTION RETURN RECORD.               *)
(*                                                                *)
(*----------------------------------------------------------------*)
(**)
(* END %INCLUDE DIFLSM. *)
```

```
%PAGE
(* %INCLUDE DISPCRB *)
(**)
  PROCEDURE DISPCRB(VAR CRB:CRBPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       DISPOSE OF CRB                                              *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O  DESCRIPTION                                   *)
(*       CRB      I/O  CONSTITUENT READ BLOCK ADDRESS                *)
(*       RR        O   ERROR CONDITION RETURN CODE                   *)
(*                     = 0  OK RETURN CODE                           *)
(*                     = 1  YOU BLEW IT                              *)
(*                     = 2  THE ROUTINE BLEW IT                      *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1       I    VAR1 NAME MUST BE FILLED, CHARACTER DATA   *)
(*                        MUST BE PROVIDED                           *)
(*        VAR2       I    VAR2 MUST BE SPECIFIED                     *)
(*      COM2                                                         *)
(*        VAR3       I    CHARACTER DATA MUST BE SPECIFIED           *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                        *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE   *)
(*           NARRATION ON THE NEXT LINE.                             *)
```

```
(*        YY/MM/DD  CCXX  I. M. APERSON                                  *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                           *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPCRB *)
```

```
%PAGE
(* %INCLUDE DISPEMM. *)
(**)
  PROCEDURE DISPEMM(VAR KEYE:ENTKEY;VAR RR:RET_REC); EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       RELEASE ALL SPACE ALLOCATED TO AN ENTITY. NO DANGLING        *)
(*       REFERENCES TO THIS ENTITY SHOULD EXIST IN AN NDS OR          *)
(*       NODELIST.                                                    *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       ENTITY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE      - KEY OF THE ENTITY TO BE DISPOSED.           *)
(*       OUTPUT                                                       *)
(*          KEYE      - SET TO NIL.                                 *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                    *)
(*    CHANGE CONTROL                                                  *)
(*       CHANGED:  12/10/84  J. JOHNSON   - TO CALL 'MASDSP'          *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPEMM. *)
```

```
%PAGE
(* %INCLUDE DISPKND *)
(**)
  PROCEDURE DISPKND(CONST KEYC:ENTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       DISPOSE OF ALL ENTITIES OF A SPECIFIC INSTANCE COLLECTOR.        *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL                                                           *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       ENTITY PACKAGE.                                                  *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          KEYC       - KEY OF A COLLECTOR NODE.                        *)
(*       OUTPUT                                                           *)
(*          RR         - THE FUNCTION RETURN CODE.                       *)
(*                                                                        *)
(*    METHOD                                                              *)
(*       IF KEYC IS AN INSTANCE COLLECTOR, THEN DISPOSE OF ALL            *)
(*       ELEMENTS OF IN THE CONSTITUENT LIST.                             *)
(*                                                                        *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPKND *)
```

```
%PAGE
(* %INCLUDE DISPLSM. *)
(**)
  PROCEDURE DISPLSM(VAR POSITION:LISTPSTN;VAR LISTREF:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       DELETE SPACE ALLOCATED TO A SYSTEM LIST.                    *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          LISTREF   - POINTER TO A SYSTEM LIST WHOSE SPACE IS TO   *)
(*                      BE DEALLOCATED.                              *)
(*       OUTPUT                                                      *)
(*          LISTREF   - POINTER TO A SYSTEM LIST WITH ZERO SIZE.     *)
(*          POSITION  - POSITION IS SET TO ZERO INDICATING START OF  *)
(*                      SYSTEM LIST.                                 *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*       CHANGED:  12/10/84   J. JOHNSON  - CALL MASDSP.             *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPLSM. *)
```

```
%PAGE
(* %INCLUDE DISPNDM. *)
(**)
  PROCEDURE DISPNDM(VAR NDSREM:NDS;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       FORCE AN NDS OUT OF MEMORY.                                  *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       NETWORK PACKAGE.                                             *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NDSREM    - THE NDS TO BE FORCED OUT OF MEMORY.           *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*    CHANGE CONTROL:                                                 *)
(*       EDS - MAS VERSION 2 - 9/17/84 - REWRITTEN TO DISPOSE OF      *)
(*             ALL ELEMENTS OF A MAS WORKING FORM MODEL.              *)
(*                                                                    *)
(*       EDS - MAS VERSION 2 - 11/21/84 - TEST FOR EXISTANCE OF A     *)
(*             MODEL IN THE WORKING FORM BEFORE DELETING IT.          *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPNDM. *)
```

```
%PAGE
(* %INCLUDE DISPNM. *)
(**)
  PROCEDURE DISPNM(VAR KEYL:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       REMOVE ALL ENTITIES FROM THE LIST AND FREE THE ALLOCATED     *)
(*       SPACE. THE EMPTY LIST IS ALSO DELETED AND REMOVED FROM THE   *)
(*       LIST OF LISTS.                                               *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYL      - KEY OF THE LIST WHOSE ENTITIES ARE TO         *)
(*                      BE REMOVED.                                   *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       THE STACK_OF_LISTS IS READ.  FOR EACH LIST_OF_LISTS ON THE   *)
(*       STACK_OF_LISTS, KEYL IS REMOVED FROM THE LIST.  WHEN ALL     *)
(*       LISTS HAVE BEEN SEARCHED, KEYL IS DISPOSED.                  *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE DISPNM. *)
```

```
%PAGE                                                                   00010006
(* %INCLUDE ELDNM. *)                                                   00020000
(**)                                                                    00030000
  PROCEDURE ELDNM(VAR KEYL:LISTKEY;VAR RR:RET_REC);EXTERNAL;            00040009
(**)                                                                    00050000
(*--------------------------------------------------------------------*) 00060006
(*                                                                    *) 00070006
(*   $FUNCTION:                                                       *) 00080011
(*       CREATE A LIST WITH ALL DUPLICATE ENTITIES ELIMINATED.        *) 00090005
(*       THE FIRST REFERENCE IS MAINTAINED AND ALL SUBSEQUENT         *) 00100005
(*       ENTITIES ARE REMOVED.                                        *) 00110005
(*                                                                    *) 00120007
(*   $DESCRIPTION OF ARGUMENTS:                                       *) 00121011
(*       NAME    I/O  DESCRIPTION                                     *) 00122011
(*       KEYL     I   - KEY OF THE LIST WHICH MAY CONTAIN DUPLICATE   *) 00123011
(*                      ENTITIES.  THE LIST WILL HAVE ALL DUPLICATES  *) 00124011
(*                      REMOVED.                                      *) 00125011
(*       RR       O   - THE FUNCTION RETURN RECORD.                   *) 00126011
(*                                                                    *) 00127011
(*   $ENVIRONMENT:                                                    *) 00130011
(*       LANGUAGE: IBM PASCAL                                         *) 00140011
(*       HARDWARE: IBM 360/370/43XX                                   *) 00141011
(*                                                                    *) 00150000
(*   $PROCESSING DESCRIPTION:                                         *) 00261011
(*       EACH ENTRY ON THE SYSTEM LIST IS READ. THE ADB.PROBIT IS     *) 00262010
(*       SET ON, AND THE ENTITY KEY IS PLACED ON THE NEW SYSTEM LIST.*) 00263010
(*       IF THE ADB.PROBIT IS ALREADY SET ON, THEN THE ENTITY IS A    *) 00264010
(*       DUPLICATE AND NOT PLACED ON THE NEW LIST.                    *) 00265010
(*         THE NEW LIST REPLACES THE OLD SYSTEM LIST IN THE APPL-     *) 00266010
(*       ICATION LIST.  ALL PROBITS ARE THEN RESET TO 'OFF'.          *) 00267010
(*                                                                    *) 00268010
(*    CHANGE CONTROL:                                                 *) 00269010
(*      REVISED:  09/02/86       B. A. ULMER         W315            *) 00269113
(*         TO USE MAS INTERNAL PROCESS FLAG (MAPROB2) INSTEAD OF      *) 00269213
(*         MAPROB    (CONFLICT WITH DELRUL)                           *) 00269313
(*                                                                    *) 00269413
(*      REVISED:  04/26/85       E. D. SHREVE        W315            *) 00269513
(*         TO USE MAS INTERNAL PROCESS FLAG (MAPROB)                  *) 00269613
(*                                                                    *) 00269711
(*      REVISED:  02/07/85       E. D. SHREVE        W315            *) 00269811
(*         REWRITTEN TO PROCESS MORE EFFICIENTLY.                     *) 00269910
(*--------------------------------------------------------------------*) 00270006
(**)                                                                    00280004
(* END %INCLUDE ELDNM. *)                                              00290000
```

```
%PAGE
(* %INCLUDE ELMNODM. *)
(**)
  PROCEDURE ELMNODM(CONST KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       RETURN AN ENTBLOCK CORRESPONDING TO A KEY.                       *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       ENTITY PACKAGE.                                                  *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          KEYE       - THE KEY OF THE ENTITY.                         *)
(*       OUTPUT                                                           *)
(*          ENTDEF     - THE CORRESPONDING ENTBLOCK.                    *)
(*          RR         - THE FUNCTION RETURN RECORD.                    *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE ELMNODM. *)
```

```
%PAGE
(* %INCLUDE EXCRBE *)
(**)
  PROCEDURE EXCRBE(CONST CRB:CRBPNTR; CONST POS1:RDBSIZE;
      CONST POS2:RDBSIZE; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       EXCHANGE TWO ENTRIES IN THE CRB                             *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME    I/O  DESCRIPTION                                    *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS                 *)
(*       POS1     I   POSITION OF FIRST ENTRY TO EXCHANGE            *)
(*       POS2     I   POSITION OF SECOND ENTRY TO EXCHANGE           *)
(*       RR       O   ERROR CONDITION RETURN CODE                    *)
(*                    = 0  OK RETURN CODE                            *)
(*                    = 1  YOU BLEW IT                               *)
(*                    = 2  THE ROUTINE BLEW IT                       *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1     I   VAR1 NAME MUST BE FILLED, CHARACTER DATA     *)
(*                     MUST BE PROVIDED                              *)
(*        VAR2     I   VAR2 MUST BE SPECIFIED                       *)
(*      COM2                                                         *)
(*        VAR3     I   CHARACTER DATA MUST BE SPECIFIED             *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH       *)
(*       FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.            *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*       TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*       THE FUNCTION/EXECUTION OF THIS ROUTINE.                     *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*       YY/MM/DD  CCZZ  I. M. THECHANGER                            *)
(*          DESCRIPTION OF LATEST CHANGE MADE.                       *)
```

```
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                          *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE     *)
(*           NARRATION ON THE NEXT LINE.                               *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                 *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                         *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE EXCRBE *)
```

```
%PAGE
(* %INCLUDE EXPCLSM. *)
(**)
  PROCEDURE EXPCLSM(CONST LISTIN:LISTPNTR;VAR LISTOUT:LISTPNTR;
      VAR LSTFLG:BOOLEAN; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       EXPAND LIST WITH ALL OF ITS CONSTITUENTS AND PLACE THIS     *)
(*       EXPANDED LIST IN LISTOUT.                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME         I/O      DESCRIPTION                          *)
(*       ====         ===      ===========                          *)
(*       LISTIN        I       LIST CONTAINING ENTITIES TO BE       *)
(*                             EXPANDED.                            *)
(*       LISTOUT       O       LIST OF INCLUSIVE CONSTITUENTS       *)
(*       LSTFLG        I       FLAG TO TELL IF FIRST TIME THRU      *)
(*       RR            O       FUNCTION RETURN CODE.                *)
(*                             = 0  GOOD RETURN                     *)
(*                             > 0  CRITICAL ERROR                  *)
(*                             < 0  WARNING                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360,370,43XX                           *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE            *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THIS ROUTINE INVOKES ITSELF RECURSIVELY AND FILLS LISTOUT   *)
(*       BY ADDING EACH NEST OF CONSTITUENTS DIRECTLY AFTER THE      *)
(*       PARENT ENTITY.                                              *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 01/10/86   B. A. ULMER              W315           *)
(*                FIX BUG DEALING WITH PREVIOUS FIX                  *)
(*                                                                   *)
(*       REVISED: 05/21/85   B. A. ULMER              W315           *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING        *)
(*                                                                   *)
(*       REVISED: 04/26/85   E.D. SHREVE              W315           *)
(*                TO USE INTERNAL MAS PROCESS FLAG MAPROB            *)
(*                                                                   *)
```

```
(*       REVISED: 02/18/85   B.A. ULMER                 W315         *)
(*                IMPLEMENT THE CNST READ BLOCK                       *)
(*                                                                    *)
(*       CREATED: 06/13/84   D.J. KERCHNER              W315         *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE EXPCLSM. *)
```

```
%PAGE
(* %INCLUDE EXPCRB *)
(**)
  PROCEDURE EXPCRB(VAR CRB:CRBPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       EXPAND THE CRB                                              *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME    I/O  DESCRIPTION                                    *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS                 *)
(*       RR       O   ERROR CONDITION RETURN CODE                    *)
(*                    = 0  OK RETURN CODE                            *)
(*                    = 1  YOU BLEW IT                               *)
(*                    = 2  THE ROUTINE BLEW IT                       *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1      I   VAR1 NAME MUST BE FILLED, CHARACTER DATA     *)
(*                      MUST BE PROVIDED                             *)
(*        VAR2      I   VAR2 MUST BE SPECIFIED                       *)
(*      COM2                                                         *)
(*        VAR3      I   CHARACTER DATA MUST BE SPECIFIED             *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                        *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE   *)
(*           NARRATION ON THE NEXT LINE.                             *)
```

```
(*        YY/MM/DD  CCXX  I. M. APERSON                                  *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                           *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(**)
(* END %INCLUDE EXPCRB *)
```

```
%PAGE
(* %INCLUDE EXPSUDB *)
(**)
  PROCEDURE EXPSUDB(VAR ENTBPNTR:ENTPNTR;CONST OLDSIZE:ENTSIZE;
    CONST NEWSIZE:ENTSIZE;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       EXPAND A SYSTEM UDB (USER DATA BLOCK)                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        OLDSIZE        I   SIZE OF THE AREA TO BE EXPANDED         *)
(*        NEWSIZE        I   SIZE OF THE OUTPUT DATA AREA FOR THE    *)
(*                           EXPANDED ENTBLOCK                       *)
(*        ENTBPNTR       I   POINTER TO THE ENTBLOCK TO BE EXPANDED  *)
(*        ENTBPNTR       O   POINTER TO THE EXPANDED ENTBLOCK        *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        EXPAND THE USER DATA BLOCK (UDB)                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 07/09/85          B. A. ULMER                 FRMI    *)
(*     CHANGE TO MAKE THIS ROUTINE MORE VAX COMPATIABLE - TAKE OUT THE*)
(*     MIN FUNCTION                                                  *)
(*                                                                   *)
(*    REVISED: 12/10/84          J. JOHNSON                          *)
(*     TO CALL MASDSP                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE EXPULSM. *)
(**)
  PROCEDURE EXPULSM(CONST LISTIN:LISTPNTR; VAR LISTOUT:LISTPNTR;
      VAR LSTFLG:BOOLEAN; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       PLACE THE EXPANDED LIST WITH ALL OF ITS USERS IN LISTOUT.   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME      I/O     DESCRIPTION                               *)
(*       ====      ===     ===========                               *)
(*       LISTIN     I       LIST TO BE EXPANDED.                     *)
(*       LISTOUT    O       EXPANDED LIST.                           *)
(*       LSTFLG     I       FLAG TO TELL IF FIRST TIME THRU          *)
(*       RR         O       FUNCTION RETURN RECORD.                  *)
(*                          = 0   GOOD RETURN                        *)
(*                          > 0   CRITICAL ERROR                     *)
(*                          < 0   WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE:  IBM PASCAL                                       *)
(*       HARDWARE SYSTEM: IBM 360,370,43XX                           *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE.           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THIS ROUTINE INVOKES ITSELF RECURSIVELY AND FILLS LISTOUT   *)
(*       BY ADDING EACH NEST OF USERS DIRECTLY AFTER ITS USER        *)
(*       REFERENCE.                                                  *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 01/10/86    B. A. ULMER            W315           *)
(*                FIX BUG DEALING WITH PREVIOUS FIX                  *)
(*                                                                   *)
(*       REVISED: 05/21/85    B. A. ULMER            W315           *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING        *)
(*                                                                   *)
(*       REVISED: 04/26/85    E. D. SHREVE           W315           *)
(*                TO USE INTERNAL MAS PROCESS FLAG MAPROB            *)
(*                                                                   *)
(*       ORIGINATED: 06/13/84  D. J. KERCHNER        W315           *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE EXPULSM. *)
```

```
%PAGE
(* %INCLUDE EXPUNM. *)
(**)
  PROCEDURE EXPUNM(VAR KEYL:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       EXPAND THE LIST TO INCLUDE ALL USERS OF THE ENTITIES.        *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          LISTIN    - LIST WHOSE CONSTITUENTS ARE TO BE EXPANDED.   *)
(*          KEYL      - KEY OF THE LIST TO BE EXPANDED.               *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE EXPUNM. *)
```

```
%PAGE
(* %INCLUDE FDSCH. *)
(**)
  PROCEDURE FDSCH(CONST SCH_ROOT:ENTKEY;CONST KIND:ORD_KIND;
      VAR SCH_PTR:ENTKEY;VAR POSITION:LISTPSTN;
      VAR RR:RET_REC); EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       FIND A SCHEMA_INSTANCE_COLLECTOR OR SCHEMA_CLASS ENTITY ON  *)
(*       THE SPECIFIED SCHEMA_ROOT'S CONSTITUENT LIST.               *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       SCHEMA PACKAGE.                                             *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          NDSREM    - THE NETWORK TO BE SEARCHED.                  *)
(*          KIND      - VALUE TO BE SEARCHED FOR IN THE ENTBLOCK OF  *)
(*                      THE CLASS OR INSTANCE COLLECTOR NODE.  THIS  *)
(*                      IS THE KIND OF THE COLLECTED INSTANCES FOR   *)
(*                      INSTANCE COLLECTORS.                         *)
(*       OUTPUT                                                      *)
(*          SCH_PTR   - POINTER TO THE FOUND ENTITY WITH SPECIFIED   *)
(*                      DATA.KIND.                                   *)
(*          POSITION  - POSITION IN THE CONSTITUENT LIST OF THE LAST *)
(*                      SCHEMA CLASS OR INSTANCE COLLECTOR ENTITY    *)
(*                      WITH HEADER.KIND LESS THAN OR EQUAL TO THE   *)
(*                      SPECIFIED KIND.                              *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE FDSCH. *)
```

```
%PAGE
(* %INCLUDE FNDCRBE *)
(**)
  PROCEDURE FNDCRBE(CONST CRB:CRBPNTR; CONST EKEY:ENTKEY;
      VAR CRBPOS:RDBSIZE;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       FIND A SPECIFIC ENTRY IN THE CRB                            *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O  DESCRIPTION                                   *)
(*       CRB      I/O  CONSTITUENT READ BLOCK ADDRESS                *)
(*       EKEY      I   ENTITY KEY WHICH IS TO BE FOUND IN THE CRB    *)
(*       CRBPOS    O   POSITION IN CRB WHERE EKEY WAS FOUND          *)
(*       RR        O   ERROR CONDITION RETURN CODE                   *)
(*                     = 0  OK RETURN CODE                           *)
(*                     = 1  YOU BLEW IT                              *)
(*                     = 2  THE ROUTINE BLEW IT                      *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1      I    VAR1 NAME MUST BE FILLED, CHARACTER DATA    *)
(*                       MUST BE PROVIDED                            *)
(*        VAR2      I    VAR2 MUST BE SPECIFIED                      *)
(*      COM2                                                         *)
(*        VAR3      I    CHARACTER DATA MUST BE SPECIFIED            *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
```

```
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                          *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE     *)
(*           NARRATION ON THE NEXT LINE.                               *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                 *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                         *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE FNDCRBE *)
```

```
%PAGE
(* %INCLUDE FNDSKIND *)
(**)
  PROCEDURE FNDSKIND(CONST SCHKEY:ENTKEY;VAR KINDARY:KIND_ARRAY;
      VAR NUMKIND:INTEGER;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       BUILD AN ARRAY OF KIND VALUE COLLECTED BY A CLASS OR       *)
(*       INSTANCE COLLECTOR IN THE SCHEMA.                          *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL                                                     *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          SCHKEY    - KEY OF THE CLASS OR INSTANCE COLLECTOR NODE. *)
(*          KINDARY   - ARRAY TO STORE THE COLLECTED KINDS.        *)
(*       OUTPUT                                                     *)
(*          NUMKIND   - NUMBER OF KIND VALUES PUT INTO KINDARY.    *)
(*          RR        - THE FUNCTION RETURN RECORD.                *)
(*                                                                  *)
(*    METHOD                                                        *)
(*       1. IF SCHKEY IS AN INSTANCE COLLECTOR, THE KIND VALUE FROM *)
(*          THE 1ST CONSTITUENT'S ADB IS PUT INTO KINDARY.         *)
(*       2. IF SCHKEY IS A CLASS COLLECTOR, ALL INCLUSIVE INSTANCE  *)
(*          COLLECTORS ARE FOUND AND THEIR KINDS PUT IN KINDARY.   *)
(*          THIS IS ACCOMPLISHED BY RECURSIVE CALLS TO FNDSKIND.   *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE FNDSKIND *)
```

```
%PAGE
(* %INCLUDE FNDURUL. *)
(**)
  PROCEDURE FNDURUL(CONST ENTK:ENTKEY;VAR DEPENDENCE:BOOLEAN;
      VAR STRNGTH:BOOLEAN;VAR REQ_USER:BOOLEAN;
      VAR REQ_CNST:BOOLEAN;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GETS THE RULE FROM THE INSTANCE COLLECTOR FOR A GIVEN       *)
(*       ENTKEY.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        ENTK           I   ENTITY FOR WHICH THE DELETE RULES WILL BE*)
(*                           GOTTEN FROM IT'S INSTANCE KIND COLLECTOR *)
(*        DEPENDENCE     O   INDICATES IF THE DELETE RULE IS DEPENDENT*)
(*                           (1 - FALSE) OR INDEPENDENT (0 - FLASE)   *)
(*        STRNGTH        O   INDICATES IF THE DELETE RULE IS STRONG   *)
(*                           (1 - TRUE) OR WEAK (0 - FALSE)           *)
(*        REQ_USER       O   INDICATES WHETHER THE ENTITY REQUIRES    *)
(*                           USER(S) TO EXIST                         *)
(*        REQ_CNST       O   INDICATES WHETHER THE ENTITY REQUIRES    *)
(*                           CONSTITUENT(S) TO EXIST                  *)
(*        RC             O   EXTERNAL RETURN CODE                     *)
(*                           = 0  OK RETURN CODE                      *)
(*                           > 0  CRITICAL ERROR                      *)
(*                           < 0  WARNING                             *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*       THE SCH_ROOT IS FOUND AND THE KIND IS FOUND AND PASSED TO    *)
(*       FDSCH TO FIND THE SCHPTR THAT POINTS TO THE RULE_DEP AND     *)
(*       AND RULE_STRNGTH AND RETURNS THE RULES.                      *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
```

```
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 09/06/85          B. A. ULMER                  FRMI      *)
(*     ADDED TWO NEW PARAMETERS FOR HANDLING THE TWO NEW DELETE RULES    *)
(*     (REQ_USER, REQ_CNST)                                              *)
(*                                                                       *)
(*    REVISED: 02/18/85          B. A. ULMER                  FRMI      *)
(*     CHANGED THE STRUCTURE OF THE INTERNAL ITEM FOR IMPLEMENTATION     *)
(*     OF THE CRB                                                        *)
(*                                                                       *)
(*    ORIGINATED: 06/19/85       C. J. SAMPLE                 FRMI      *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END--------------------------------------------------------------------*)
(**)
(* END %INCLUDE FNDURUL. *)
```

```
%PAGE
(* %INCLUDE GTCRBE *)
(**)
  PROCEDURE GTCRBE(CONST CRB:CRBPNTR; VAR CRBPOS: RDBSIZE;
      CONST EKEY:ENTKEY; VAR POS:LISTPSTN; VAR DIR:LISTDIR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                 *)
(*    FUNCTION:                                                    *)
(*       GET AN ENTRY IN THE CRB                                   *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*       IBM PASCAL LANGUAGE                                       *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.  *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*       NAME     I/O  DESCRIPTION                                 *)
(*       CRB      I/O  CONSTITUENT READ BLOCK ADDRESS              *)
(*       CRBPOS    I   POSITION IN CRB OF ENTRY REQUESTED          *)
(*       EKEY      O   KEY OF ENTITY CONTAINING THE CONSTITUENT LIST *)
(*       POS       O   LIST POSITION SETTING                       *)
(*       DIR       O   DIRECTION TO READ OF LIST (FORWARD OR REVERSE) *)
(*       RR        O   ERROR CONDITION RETURN CODE                 *)
(*                     = 0  OK RETURN CODE                         *)
(*                     = 1  YOU BLEW IT                            *)
(*                     = 2  THE ROUTINE BLEW IT                    *)
(*                                                                 *)
(*    COMMONS:                                                     *)
(*      COM1                                                       *)
(*        VAR1      I    VAR1 NAME MUST BE FILLED, CHARACTER DATA  *)
(*                       MUST BE PROVIDED                          *)
(*        VAR2      I    VAR2 MUST BE SPECIFIED                    *)
(*      COM2                                                       *)
(*        VAR3      I    CHARACTER DATA MUST BE SPECIFIED          *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH    *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.         *)
(*                                                                 *)
(*    COMMENTS:                                                    *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                  *)
(*                                                                 *)
```

```
(*    CHANGE CONTROL:                                                  *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                             *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                        *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                          *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE     *)
(*           NARRATION ON THE NEXT LINE.                               *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                         *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE GTCRBE *)
```

```
%PAGE
(* %INCLUDE INDLSM *)
(**)
  PROCEDURE INDLSM(CONST KEYE:ENTKEY;CONST LISTREF:LISTPNTR;
      VAR POSITION:LISTPSTN;VAR INLST:BOOLEAN;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    AUTHOR:  UNKNOWN                 CADD    CREATED: YY/MM/DD CC *)
(*    VERSION: MAS VER 2                       REVISED: 84/10/11 CC *)
(*                                                                  *)
(*    FUNCTION:                                                     *)
(*       LOCATE AN ENTITY IN A SYSTEM LIST.                         *)
(*                                                                  *)
(*    ENVIRONMENT:                                                  *)
(*        IBM PASCAL LANGUAGE                                       *)
(*        IBM 3?XX, 43XX, DEC VAX 11/780                            *)
(*                                                                  *)
(*    DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME     I/O  DESCRIPTION                                 *)
(*        KEYE      I   ENTITY TO BE LOCATED.                       *)
(*        LISTREF   I   LIST TO BE SEARCHED.                        *)
(*        POSITION  O   POSITION OF ENTITY IN SYSTEM LIST.          *)
(*        INLST     O   TRUE IF AN ENTITY IN THE LIST CORRESPONDS   *)
(*                      TO KEYE ELSE FALSE.                         *)
(*        RR        O   ERROR CONDITION RETURN CODE.                *)
(*                      = 0  NORMAL RETURN CODE.                    *)
(*                                                                  *)
(*    COMMONS:                                                      *)
(*                                                                  *)
(*    PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*    COMMENTS:                                                     *)
(*                                                                  *)
(*    CHANGE CONTROL:                                               *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                       *)
(*           UPDATED DOCUMENTATION.                                 *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                         *)
(*           CHANGED DECLARATION OF KEYL TO VAR.                    *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE INDLSM *)
```

```
%PAGE
(* %INCLUDE INNM. *)
(**)
  FUNCTION INNM(CONST KEYE:ENTKEY;CONST KEYL:LISTKEY;
      VAR RR:RET_REC):BOOLEAN; EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       INDICATE WHETHER A LIST REFERENCES AN ENTITY.               *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          KEYE      - KEY TO LOOK FOR IN THE LIST.                 *)
(*          KEYL      - THE KEY OF THE LIST TO EXAMINE.              *)
(*       OUTPUT                                                      *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*          FUNCTION VALUE - TRUE IF ENTITY IS IN LIST ELSE FALSE.   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE INNM. *)
```

```
%PAGE
(* %INCLUDE INTLSM. *)
(**)
  PROCEDURE INTLSM(CONST LIST1:LISTPNTR;CONST LIST2:LISTPNTR;
      VAR POSITION:LISTPSTN;VAR LISTOUT:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*   $FUNCTION:                                                    *)
(*       CREATE A LIST WHICH IS THE INTERSECTION OF TWO LISTS.     *)
(*                                                                 *)
(*   $DESCRIPTION OF ARGUMENTS:                                    *)
(*        NAME          I/O  DESCRIPTION                         *)
(*        ====          ===  ===========                         *)
(*        LIST1          I   LIST TO BE INTERSECTED WITH THE SECOND *)
(*        LIST2          I   LIST TO BE INTERSECTED WITH THE FIRST  *)
(*        POSITION       I   INTEGER INDICATING THE POSITION ON    *)
(*                           LISTOUT                               *)
(*        LISTOUT        O   LIST CONTAINING COMMON ENTITIES TO THE *)
(*                           INPUT LISTS                           *)
(*        RC             O   EXTERNAL RETURN CODE                  *)
(*                           = 0  OK                               *)
(*                           > 0  CRITICAL ERROR                   *)
(*                           < 0  WARNING                          *)
(*                                                                 *)
(*   $COMMONS:                                                     *)
(*                                                                 *)
(*   $ENVIRONMENT:                                                 *)
(*        LANGUAGE: IBM PASCAL                                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                 *)
(*   $EXECUTION PROCEDURE:                                         *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE         *)
(*                                                                 *)
(*   $PROCESSING DESCRIPTION:                                      *)
(*        FIND THOSE ENTITIES WHICH ARE COMMON TO BOTH INPUT LISTS *)
(*                                                                 *)
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*                                                                 *)
(*    REVISED: 07/01/85          B. A. ULMER                FRMI   *)
(*     ELIMINATE THE MIN FUNCTION TO IMPROVE COMPATABILITY WITH VAX *)
(*                                                                 *)
(*    REVISED: 02/22/85          B. A. ULMER                FRMI   *)
(*     FIXED EMPTY LIST ELEMENT PROBLEM                            *)
(*                                                                 *)
(*    REVISED: 12/24/85          B. A. ULMER                FRMI   *)
(*     ADDED SYSTEM LIST CURRENT LENGTH INDICATOR -- LSTLNM        *)
(*                                                                 *)
```

```
%PAGE
(* %INCLUDE INTNM. *)
(**)
  PROCEDURE INTNM(CONST KEYL1:LISTKEY;CONST KEYL2:LISTKEY;
      VAR KEYLOUT:LISTKEY; VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       CREATE A LIST WHICH IS THE INTERSECTION OF LISTS           *)
(*       REFERENCED BY KEYL1 AND KEYL2.  IF AN ENTITY IS IN BOTH    *)
(*       KEYL1 AND KEYL2, IT IS ADDED TO KEYLOUT.  THE ORDER AND    *)
(*       DATA OF KEYLOUT IS THAT OF KEYL1.                          *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          KEYL1      - KEY OF THE LIST WHICH DEFINES THE ORDER AND *)
(*                       ENTITIES OF KEYLOUT.                       *)
(*          KEYL2      - KEY OF THE LIST TO BE INTERSECTED.        *)
(*       OUTPUT                                                     *)
(*          KEYLOUT    - KEY OF LIST WITH ALL ENTITIES REFERENCED BY *)
(*                       BOTH KEYL1 AND KEYL2.                      *)
(*          RR         - THE FUNCTION RETURN RECORD.               *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE INTNM. *)
```

```
%PAGE
(* %INCLUDE LSTLNM. *)
(**)
  FUNCTION LSTLNM(CONST LISTREF:LISTPNTR;VAR RR:RET_REC):LISTSIZE;
      EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       RETURN THE NUMBER OF NON-VACANT ENTITIES IN A SYSTEM LIST.       *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       LIST PACKAGE.                                                    *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          LISTREF    - POINTER TO A SYSTEM LIST.                        *)
(*       OUTPUT                                                           *)
(*          RR         - THE FUNCTION RETURN RECORD.                      *)
(*          FUNCTION VALUE - NUMBER OF ENTITIES IN THE SYSTEM LIST.       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE LSTLNM. *)
```

```
%PAGE
(* %INCLUDE LSTMXLNM. *)
(**)
  FUNCTION LSTMXLNM(CONST LISTREF:LISTPNTR;VAR RR:RET_REC):LISTSIZE;
      EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       RETURN THE NUMBER OF ENTRIES ALLOCATED TO A SYSTEM LIST.   *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          LISTREF   - POINTER TO A SYSTEM LIST.                   *)
(*       OUTPUT                                                     *)
(*          FUNCTION VALUE - SIZE OF SYSTEM LIST.                   *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE LSTMXLNM. *)
```

```
%PAGE
(* %INCLUDE MAEA. *)
(**)
  PROCEDURE MAEA(CONST KEY1:ANYKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
%PAGE
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       ACTIVATE AN ENTITY.                                         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   KEY OF THE ENTITY OR LIST OF ENTITIES TO *)
(*                           BE ACTIVATED                            *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*         FOR EACH KEY, AS AN ENTITY OR A MEMBER OF A LIST          *)
(*         RESET THE DELETE FLAG                                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 04/30/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING   *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA    *)
(*     ADDED A CHECK FOR NIL POINTER ON KEY1                         *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    ORIGINATED: 07/25/84       D. J. KERCHNER              FRMI     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE MAEA. *)
(**)
```

```
%PAGE
(* %INCLUDE MAEAI *)
(**)
  PROCEDURE MAEAI(CONST KEY1:ANYKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       ACTIVATE AN ENTITY OR A LIST OF ENTITIES AND THEIR          *)
(*       INCLUSIVE CONSTITUENTS.                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME      I/O     DESCRIPTION                               *)
(*       ====      ===     ===========                               *)
(*       KEY1       I    KEY OF THE ENTITY OR LIST OF ENTITIES TO    *)
(*                       BE ACTIVATED.                               *)
(*       RC         O    THE FUNCTION RETURN CODE.                   *)
(*                       = 0  GOOD RETURN                            *)
(*                       > 0  CRITICAL ERROR                         *)
(*                       < 0  WARNING                                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360,370,43XX                           *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       MODEL ACCESS SOFTWARE INTERFACE ROUTINE                     *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS AN ENTITY, THEN THAT ENTITY AND ITS INCLUSIVE    *)
(*         CONSTITUENT LIST WILL BE ACTIVATED.                       *)
(*       IF KEY1 IS A LIST KEY, THEN THE INCLUSIVE CONSTITUENT LISTS *)
(*         OF EACH ENTITY WILL BE ACTIVATED.                         *)
(*       NOW USES THE INTERNAL MAS PROCESS FLAG (MAPROB) IN THE      *)
(*         T_ELEMENT.IIT.                                            *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 05/01/86      B. A. ULMER          W315            *)
(*                ADDED A CALL CNVOSP TO CONVERT AN "OUT OF MEMORY"  *)
(*                CONDITION TO USER RECOGMIZEABLE FORM               *)
(*                                                                   *)
(*       REVISED: 07/11/85      B. A. ULMER          W315            *)
(*                ADD NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND  *)
(*                DEBUGGING PURPOSES                                 *)
(*                                                                   *)
(*       REVISED: 04/26/85      E. D. SHREVE         W315            *)
(*                TO USE THE INTERNAL MAS PROCESS FLAG AND TO CALL   *)
(*                EXPCLST INSTEAD OF EXPALST                         *)
(*                                                                   *)
```

```
(*       REVISED: 02/18/85      B. A. ULMER          W315            *)
(*                STRUCTURE CHANGE FOR THE CNST. READ BLOCK.         *)
(*                                                                   *)
(*       REVISED: 08/14/86      K. M. ROSS           W315            *)
(*                ADDED NIL POINTER CHECK FOR KEY1.                  *)
(*                                                                   *)
(*       ORIGINATED: 07/26/84   D. J. KERCHNER       W315            *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAEAI *)
```

```
%PAGE
(* %INCLUDE MAEAV *)
(**)
  PROCEDURE MAEAV(CONST KEY1:ENTKEY;VAR IAVAL:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       FIND THE PRESENT VALUE OF THE ACTIVATION SETTING FOR AN     *)
(*       ENTITY.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   KEY OF THE ENTITY WHOSE ACTIVATION      *)
(*                           SETTING IS TO BE CHECKED                *)
(*        IVAL           O   VALUE OF THE SWITCH -                   *)
(*                           = 1  TRUE                               *)
(*                           = 0  FALSE                              *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE ACTIVITY STATUS OF THE ENTITY IS TO BE CHECKED.         *)
(*                                                                   *)
(*       IF THE ENTITY IS ACTIVE (NOT MARKED FOR DELETE), THEN       *)
(*         THE ACTIVITY STATUS IS TRUE AND AN INTEGER FLAG VALUE     *)
(*         OF (1) WILL BE RETURNED.                                  *)
(*                                                                   *)
(*       IF THE ENTITY IS INACTIVE (MARKED FOR DELETE), THEN THE     *)
(*         ACTIVITY STATUS IS FALSE AND AN INTEGER FLAG VALUE OF     *)
(*         (0) WILL BE RETURNED.                                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
```

```
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI      *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF MEMORY" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                        *)
(*                                                                       *)
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI      *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                          *)
(*                                                                       *)
(*    ORIGINATED: 07/27/85       D. J. KERCHNER               FRMI      *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END--------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAEAV *)
(**)
```

```
%PAGE
(**)
(* %INCLUDE MAEC *)
(**)
  PROCEDURE MAEC(CONST KEY1:ANYKEY;VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
%PAGE
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATIONS LIST OF CONSTITUENT ENTITIES.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEY1           I   KEY OF AN ENTITY OR A LIST.           *)
(*        KEY2           O   RETURNED KEY OF THE APPLICATION LIST. *)
(*        RC             O   EXTERNAL RETURN CODE                  *)
(*                           < 0  WARNING                          *)
(*                           = 0  OK                               *)
(*                           > 0  CRITICAL ERROR                   *)
(*                                                                 *)
(*   $COMMONS:                                                     *)
(*                                                                 *)
(*   $ENVIRONMENT:                                                 *)
(*        LANGUAGE: IBM PASCAL                                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                 *)
(*   $EXECUTION PROCEDURE:                                         *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                  *)
(*                                                                 *)
(*   $PROCESSING DESCRIPTION:                                      *)
(*       KEY2 IS CREATED (EMPTY LIST).                             *)
(*       IF KEY1 IS AN ENTITY, THEN THE CONSTITUENT LIST OF KEY1   *)
(*       WILL BE COPIED INTO KEY2.                                 *)
(*       IF KEY1 IS A LIST KEY, THEN THE CONSTITUENT LISTS OF EACH *)
(*       ENTITY WILL BE COPIED INTO KEY2.                          *)
(*                                                                 *)
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*    REVISED: 05/01/86          B. A. ULMER          W315         *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF MEMORY" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                  *)
(*                                                                 *)
(*    REVISED: 07/11/85          B. A. ULMER          W315         *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                    *)
(*                                                                 *)
```

```
(*    REVISED: 05/15/85          B. A. ULMER           W315          *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                   *)
(*                                                                   *)
(*    REVISED: 02/18/85          B. A. ULMER           W315          *)
(*     CHANGED THE STRUCUTRE OF THE INTERNAL ITEM FOR IMPLEMENTATION *)
(*     OF THE CRB                                                    *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS            W315          *)
(*     ADDED A CHECK FOR NIL POINTER FOR KEY1                        *)
(*                                                                   *)
(*    ORIGINATED: 06/08/84       D. J. KERCHNER        W315          *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                                *)
(* END %INCLUDE MAEC    *)
```

```
%PAGE
(* %INCLUDE MAECI *)
(**)
  PROCEDURE MAECI(CONST KEY1:ANYKEY;VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATION LIST OF INCLUSIVE CONSTITUENT         *)
(*       ENTITIES.                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME       I/O     DESCRIPTION                              *)
(*       ====       ===     ===========                              *)
(*       KEY1        I      KEY OF AN ENTITY OR A LIST.              *)
(*       KEY2        O      KEY OF THE CREATED APPLICATION LIST.     *)
(*       RC          O      FUNCTION RETURN CODE.-                   *)
(*                          = 0  GOOD RETURN                         *)
(*                          > 0  CRITICAL ERROR                      *)
(*                          < 0  WARNING                             *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360, 370, 43XX                         *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       MODEL ACCESS SOFTWARE INTERFACE ROUTINE                     *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       KEY2 IS CREATED (EMPTY LIST).                               *)
(*       IF KEY1 IS AN ENTITY, THEN THE INCLUSIVE CONSTITUENT LIST   *)
(*         OF KEY1 WILL BE COPIED INTO KEY2.                         *)
(*       IF KEY1 IS A LIST KEY, THEN THE INCLUSIVE CONSTITUENT LISTS *)
(*         OF EACH ENTITY WILL BE COPIED INTO KEY2.                  *)
(*       IT IS ASSUMED THAT THE MAPROB FLAG IS INITIALLY SET TO      *)
(*         FALSE.  AFTER PROCESSING, MAPROB FLAG IS RESET.           *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 05/01/86      B.A. ULMER              W315         *)
(*                ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF MEMORY*)
(*                CONDITION TO USER RECOGNIZEABLE FORM               *)
(*                                                                   *)
(*       REVISED: 01/20/85      B.A. ULMER              W315         *)
(*                FIX BUG DEALING WITH PREVIOUS FIX                  *)
(*                                                                   *)
```

```
(*       REVISED: 11/04/85      B.A. ULMER              W315         *)
(*                NOT ALLOW ENITIES THAT ARE ON THE APPLICATION INPUT*)
(*                LIST TO BE ON THE APPLICATION OUTPUT LIST (FIX THE *)
(*                INCONSISTENCY IN THE PROCESSING)                   *)
(*                                                                   *)
(*       REVISED: 07/11/85      B.A. ULMER              W315         *)
(*                ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING    *)
(*                AND DEBUGGING PURPOSES                             *)
(*                                                                   *)
(*       REVISED: 05/15/85      B.A. ULMER              W315         *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING        *)
(*                                                                   *)
(*       REVISED: 04/29/85      E.D. SHREVE             W315         *)
(*                TO USE THE INTERNAL MAPROB FLAG                    *)
(*                                                                   *)
(*       REVISED: 02/18/85      B.A. ULMER              W315         *)
(*                IMPLEMENT CRB STRUCTURE CHANGE                     *)
(*                                                                   *)
(*       REVISED: 08/14/86      K.M. ROSS               W315         *)
(*                ADDED A NIL POINTER CHECK FOR KEY1                 *)
(*                                                                   *)
(*       ORIGINATED: 07/26/84   D.J. KERCHNER           W315         *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAECI *)
```

```
%PAGE
(* %INCLUDE MAECIK *)
(**)
  PROCEDURE MAECIK(CONST KEY1:ANYKEY;CONST ENTKIND:ORD_KIND;
      VAR KEY2:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST OF INCLUSIVE CONSTITUENTS BY KIND.            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME      I/O      DESCRIPTION                              *)
(*       ====      ===      ===========                              *)
(*       KEY1       I     THE KEY OF AN ENTITY OR A LIST OF ENTITIES *)
(*                        WHOSE INCLUSIVE CONSTITUENTS ARE TO BE     *)
(*                        SEARCHED FOR THE SPECIFIED KIND.           *)
(*       KIND       I     THE KIND CODE OF AN ENTITY OR AN ENTITY    *)
(*                        CLASS.                                     *)
(*       KEY2       O     THE KEY OF THE LIST WHICH WILL CONTAIN ALL *)
(*                        ENTITIES OF THE SPECIFIED KIND FOUND WITHIN*)
(*                        THE INCLUSIVE CONSTITUENTS OF KEY1.        *)
(*       RC         O     THE FUNCTION RETURN CODE.                  *)
(*                        = 0   GOOD RETURN                          *)
(*                        > 0   CRITICAL ERROR                       *)
(*                        < 0   WARNING                              *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360, 370, 43XX                         *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       MODEL ACCESS SOFTWARE INTERFACE ROUTINE                     *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       A NEW LIST IS CREATED TO CONTAIN THE INCLUSIVE CONSTITUENTS,*)
(*       OR LIST MEMBERS.  FOR EACH LIST MEMBER WHOSE KIND MATCHES   *)
(*       THE GIVEN KIND, THAT MEMBER IS ADDED TO THE OUTPUT LIST     *)
(*       POINTED TO BY KEY2.                                         *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 05/01/86     B.A. ULMER              W315          *)
(*                ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF MEMORY*)
(*                CONDITION TO USER RECOGNIZEABLE FORM               *)
(*       REVISED: 07/11/85     B.A. ULMER              W315          *)
(*                ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING    *)
(*                AND DEBUGGING PURPOSES                             *)
```

```
(*       REVISED: 05/15/85     B.A. ULMER              W315         *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING       *)
(*       REVISED: 04/29/85     E.D. SHREVE             W315         *)
(*                TO USE INTERNAL MAS PROCESS FLAG (MAPROB)         *)
(*       REVISED: 02/18/85     B.A. ULMER              W315         *)
(*                TO IMPLEMENT NEW CRB STRUCTURE                    *)
(*       REVISED: 09/11/84      R. A. MCCLUSKEY        W315         *)
(*                    CHANGED PROCESSING OF SYSUSE FLAG.  DROPPED   *)
(*           ROUTINE EXPCLSTK TO USE EXPCLST INSTEAD.              *)
(*       REVISED: 08/14/86      K. M. ROSS             W315         *)
(*                 ADDED A NIL POINTER CHECK FOR KEY1               *)
(*       ORIGINATED: 08/20/84  R.A. MCCLUSKEY          W315         *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAECIK *)
```

```
%PAGE
(* %INCLUDE MAECMP *)
(**)
  PROCEDURE MAECMP(CONST KEY1:ENTKEY;VAR KEY2:LISTKEY;
     VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GIVEN AN ENTITY DETERMINE WHICH OF ITS CONSTITUENTS IT      *)
(*       COMPRESSES WITH                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   USER ENTITY WHOSE COMPRESSIBILITY IS    *)
(*                           DETERMINED BY THE CONSTITUENT ENTITY    *)
(*        KEY2           I   CONSTITUENT ENTITY BEING COMPRESSED     *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MAECQY *)
(**)
  PROCEDURE MAECQY(CONST KEY1:ENTKEY;CONST KEY2:ENTKEY;VAR CMPFLG:
      INTEGER;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GIVEN AN ENTITY AND ITS USER DETERMINE IF THE USER SHOULD BE *)
(*       COMPRESSED WITH THE ENTITY WHEN THE ENTITY IS COMPRESSED     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   USER ENTITY WHOSE COMPRESSIBILITY IS    *)
(*                           DETERMINED BY THE CONSTITUENT ENTITY    *)
(*        KEY2           I   CONSTITUENT ENTITY BEING COMPRESSED     *)
(*        CMPFLG         O   FLAG WHICH TELLS IF THE USER IS         *)
(*                           COMPRESSED WITH THE CONSTITUENT         *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *) 1
```

```
%PAGE
(* %INCLUDE MAECR *)
(**)
  PROCEDURE MAECR(VAR ENTDEF:ENTBLOCK;CONST KEYC:ANYKEY;
      VAR KEYE:ENTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN ENTITY.                                           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        ENTDEF         I   APPLICATION DATA DEFINING THE ENTITY TO  *)
(*                           BE CREATED                              *)
(*        KEYC           I   CONSTITUENT OR LIST OF CONSTITUENTS TO   *)
(*                           BE CONNECTED TO THE ENTITY              *)
(*        KEYE           O   KEY OF CREATES ENTITY                   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 10/11/84          D. J. KERCHNER              FRMI    *)
(*     UPDATE DOCUMENTATION                                          *)
(*                                                                   *)
```

```
(*     REVISED: 10/04/84          E. D. SHREVE                FRMI     *)
(*      INPUT PARAMETER ENTDEF CHANGED TO VAR FROM CONST FOR COMPATAB- *)
(*      ABILITY WITH THE DEC VAC SYSTEM                               *)
(*                                                                    *)
```

```
%PAGE
(* %INCLUDE MAECTK *)
(**)
PROCEDURE   MAECTK(VAR KNDCNT:LISTSIZE;VAR RC:EXT_RET_CODE);
            SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       TO RETURN THE NUMBER OF 'KIND' VALUES IN THE                *)
(*       WORKING-FORM MODEL.                                         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        KNDCNT          O   COUNT OF THE NUMBER OF ENTITIES IN THIS *)
(*                            WORKING FORM MODEL OF A SPECIFIC KIND  *)
(*        RC              O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK                                *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*     NDSREM                                                        *)
(*       KEY               I   KEY OF THE ROOT ELEMENT - MUST BE     *)
(*                             PROVIDED                              *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       RETRIEVES THE VALUE OF THE STD_ARY_USED_LENGTH IN THE       *)
(*       ADB OF THE SCHEMA_ROOT ELEMENT.                             *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    ORIGINATED: 10/26/84       E. D. SHREVE                FRMI     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
%PAGE                                                                   *)
(*----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                  *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE MAECTK *)
```

```
%PAGE
(* %INCLUDE MAECXQ *)
(**)
  PROCEDURE MAECXQ(CONST KEY1:ANYKEY;VAR DATAREC:BLKDATA;
      CONST PROCNAME:ROUTINE;VAR KEY2:LISTKEY;VAR RCC:EXT_RET_CODE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       EXECUTE A PROCEDURE ON THE CONSTITUENTS OF AN ENTITY, OR LIST*)
(*       OF ENTITIES.  IF AN OUTPUT LIST IS NOT PASSED, CONSTRUCT ONE *)
(*       IN ORDER TO PUT ENTITIES ON IT AS DETERMINED BY THE         *)
(*       APPLICATION PROCEDURE.                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENITIY OR LIST OF ENTITIES WHOSE CONSTIT-*)
(*                           UENTS ARE TO BE PROCESSED               *)
(*        DATAREC       I/O  APPLICATION DEFINED DATA STRUCTURE WHICH *)
(*                           EITHER SUPPLIES OR RECIEVES VALUES      *)
(*                           OPERATED ON BY THE APPLICATION PROCEDURE *)
(*        PROC           I   ENTRY POINT OF APPLICATION DEFINED      *)
(*                           PROCEDURE                               *)
(*        KEY2           O   KEY OF THE LIST CREATED                 *)
(*                           FOR THIS ROUTINE                        *)
(*        RCC            O   USER DEFINED PROCEDURE RETURN CODE      *)
(*                           = 0,1 OK RETURN CODE                    *)
(*                           = 2-7 PROCEDURE WARNING CODE            *)
(*                           = 8-15 PROCEDURE ERROR CODE             *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE USER SENDS IN THE NECESSARY INFORMATION, THEN THIS      *)
(*       ROUTINE REFERENCES THE USER'S SPECIFIED PROCEDURE TO ACT    *)
(*       UPON THE INFORMATION HE HAS SUPPLIED TO THE PROCEDURE.      *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 09/09/86          B. A. ULMER          DBMA               *)
(*     FIX PROBLEM WITH DELTING EMPTY PASSED IN APPL LIST                *)
(*                                                                       *)
(*    ORIGINATED: 06/16/86       B. A. ULMER          W315               *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(**)
(* END %INCLUDE MAECXQ *)
```

```
%PAGE
(**)
  PROCEDURE MAED(CONST KEY1:ANYKEY;VAR KEYL:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
%PAGE                                                                  *)
(* %INCLUDE MAED. *)
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*       DELETE AN ENTITY OR LIST OF ENTITIES.                        *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE DELETED *)
(*        KEYL           O   LIST OF ENTITIES UNABLE TO DELETE       *)
(*        RC             O   EXTERNAL RETURN CODE-                   *)
(*                           = 0  OK RETURN CODE                      *)
(*                           < 0  WARNING                             *)
(*                           > 0  CRITICAL ERROR                      *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                     *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*       IF KEY1 IS AN ENTKEY THEN                                    *)
(*         TRY TO DELETE THE ENTITY ACCORDING TO IT'S USER'S RULES.   *)
(*       IF KEY1 IS A LISTKEY THEN                                    *)
(*         SORT THE LIST IN A DELETABLE ORDER.                        *)
(*         TRY TO DELETE EACH ENTITY ON THE LIST ACCORDING TO ITS     *)
(*           USER'S DELETE RULES.                                     *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: 5/01/86           B. A. ULMER            W315         *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                     *)
(*                                                                    *)
(*    REVISED: 4/11/86           E. D. SHREVE           W315         *)
(*     CHANGED  TO TEST FOR NIL LIST POINTER BEFORE READING SORTLST.. *)
(*                                                                    *)
```

```
(*    REVISED: 12/30/85          B. A. ULMER              W315          *)
(*     CHANGE TO READ THE SORT LIST IN REVERSE ORDER - REMOVE THE      *)
(*     CALLS TO ELDNL AND CPYNL (NO LONGER NECESSARY SINCE SORTDLST    *)
(*     HAS BEEN IMPROVED FOR EFFICIENCY )                              *)
(*                                                                     *)
(*    REVISED: 09/  /85          B. A. ULMER              W315          *)
(*     ADD CODE TO HANDLE THE TWO NEW DELETE RULES                     *)
(*                                                                     *)
(*    REVISED: 08/  /85          L. J. BEHAN              W315          *)
(*     ADD A NEW PARAMETER TO DELRUL, DELENTY TO HANDLE APPLICATION    *)
(*     LIST POSITION PROBLEM                                           *)
(*                                                                     *)
(*    REVISED: 07/11/85          B. A. ULMER              W315          *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING   *)
(*     PURPOSES                                                        *)
(*                                                                     *)
(*    REVISED: 05/15/85          B. A. ULMER              W315          *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                     *)
(*                                                                     *)
(*    ORIGINATED: 03/08/84       C. J. SAMPLE             W315          *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAED. *)
```

```
%PAGE
(* %INCLUDE MAEDI. *)
(**)
  PROCEDURE MAEDI(CONST KEY1:ANYKEY; VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       DELETE INCLUSIVELY AN ENTITY OR LIST OF ENTITIES.           *)
(*       ENTITIES AND THEIR DIRECT AND INDIRECT CONSTITUENTS WILL    *)
(*       BE DELETED.                                                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE      *)
(*                           INCLUSIVELY DELETED -                  *)
(*        KEY2           O   LIST OF ENTITIES UNABLE TO DELETE      *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                    *)
(*                           < 0  WARNING                           *)
(*                           > 0  CRITICAL ERROR                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS AN ENTKEY THEN                                   *)
(*         AN INCLUSIVE LIST OF THE ENTITY'S CONSTITUENTS IS CREATED *)
(*         AND THE ENTITY IS ALSO PLACED ON THE INCLUSIVE LIST.      *)
(*                                                                   *)
(*       IF KEY1 IS A LISTKEY THEN                                   *)
(*         AN INCLUSIVE LIST OF THE LIST OF ENTITIES' CONSTITUENTS   *)
(*         IS CREATED AND THE LIST OF ENTITIES ARE ALSO PLACED ON THE *)
(*         INCLUSIVE LIST.                                           *)
(*                                                                   *)
(*       THE INCLUSIVE LIST IS SORTED IN A USER-CONSTITUENT ORDER.   *)
(*                                                                   *)
(*       FOR EACH ENTITY ON THE INCLUSIVE LIST, AN ATTEMPT IS MADE   *)
(*         TO DELETE THE ENTITY ACCORDING TO THE DELETE RULES OF     *)
(*         THEIR USERS.                                              *)
```

```
(*                                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*    REVISED: 05/01/86          B. A. ULMER           W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                     *)
(*                                                                    *)
(*    REVISED: 12/30/85          B. A. ULMER           W315           *)
(*     CHANGE TO READ SORT LIST IN REVERSE ORDER                      *)
(*                                                                    *)
(*    REVISED: 09/  /85          B. A. ULMER           W315           *)
(*     ADD CODE TO HANDLE THE TWO NEW DELETE RULES                    *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    REVISED: 08/  /85          L. J. BEHAN           W315           *)
(*     ADD A NEW PARAMETER TO DELRUL, DELENTY TO HANDLE APPLICATION   *)
(*     LIST POSITION PROBLEM                                          *)
(*                                                                    *)
(*    REVISED: 07/11/85          B. A. ULMER           W315           *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    REVISED: 05/15/85          B. A. ULMER           W315           *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                    *)
(*                                                                    *)
(*    ORIGINATED: 08/20/84       C. J. SAMPLE          W315           *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE
(**)
(* END %INCLUDE MAEDI. *) 1
```

```
%PAGE
(* %INCLUDE MAEDT. *)
(**)
  PROCEDURE MAEDT(CONST KEY1:ANYKEY;VAR KEYDL:LISTKEY;
      VAR KEYML:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       TEST DELETE AN ENTITY OR LIST OF ENTITIES.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE TEST   *)
(*                           DELETED                                 *)
(*        KEYDL          O   LIST OF ENTITIES WHICH WOULD BE DELETED *)
(*                           OR MARKED FOR DELETE BY MAED            *)
(*        KEYML          O   LIST OF ENTITIES WHICH WOULD BE MARKED  *)
(*                           MAED                                    *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       SIMILAR TO MAED, EXCEPT NO DELETION NOR MARK FOR DELETION   *)
(*       IS PERFORMED.                                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 06/19/86          B. A. ULMER          W315           *)
(*     CHANGE DETRUL CALLING PARAMETERS & EXCEPTION LIST TO MARK LIST *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER          W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 12/30/85          B. A. ULMER          W315          *)
(*     CHANGE TO READ THE SORT LIST IN REVERSE ORDER - REMOVE THE    *)
(*     CALLS TO ELDNL AND CPYNL  (NOT NECCESSARY  SORTDLST HAS BEEN  *)
(*     IMPROVED FOR EFFICIENCY)                                      *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315          *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 05/15/85          B. A. ULMER          W315          *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                   *)
(*                                                                   *)
(*    ORIGINATED: 06/27/84       C. J. SAMPLE         W315          *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                               *)
(**)
(* END %INCLUDE MAEDT. *)
```

```
%PAGE
(* %INCLUDE MAEDTI. *)
(**)
  PROCEDURE MAEDTI(CONST KEY1:ANYKEY;VAR KEYDL:LISTKEY;
      VAR KEYML:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       TEST FOR INCLUSIVE DELETION OF AN ENTITY OR LIST OF ENTITIES *)
(*       ENTITIES AND THEIR DIRECT AND INDIRECT CONSTITUENTS WILL BE  *)
(*       TESTED FOR DELETION.                                        *)
(*                                                                   *)
(*                                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE        *)
(*                           INCLUSIVELY TEST DELETED                *)
(*        KEYDL          O   LIST OF ENTITIES WHICH WOULD BE DELETED  *)
(*                           BY MAEDI                                *)
(*        KEYML          O   LIST OF ENTITIES WHICH WOULD BE MARKED BY*)
(*                           MAEDI                                   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS AN ENTKEY THEN                                   *)
(*         AN INCLUSIVE LIST OF THE ENTITY'S CONSTITUENTS IS CREATED  *)
(*         AND THE ENTITY IS ALSO PLACED ON THE INCLUSIVE LIST.      *)
(*                                                                   *)
(*       IF KEY1 IS A LISTKEY THEN                                   *)
(*         AN INCLUSIVE LIST OF THE LIST OF ENTITIES' CONSTITUENTS   *)
(*         IS CREATED AND THE LIST OF ENTITIES ARE ALSO PLACED ON THE *)
(*         INCLUSIVE LIST.                                           *)
(*                                                                   *)
(*       THE INCLUSIVE LIST IS SORTED IN A USER-CONSTITUENT ORDER.   *)
(*                                                                   *)
```

```
(*       FOR EACH ENTITY ON THE INCLUSIVE LIST, AN ATTEMPT IS MADE    *)
(*         TO TEST DELETE THE ENTITY ACCORDING TO THE DELETE RULES    *)
(*         OF THEIR USERS.                                            *)
(*                                                                    *)
(*       THE LIST OF MARKABLE ENTITIES IS MERGED WITH THE LIST OF     *)
(*         NON DELETABLE ENTITIES.                                    *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL                                                 *)
(*                                                                    *)
(*     REVISED: 05/01/86          B. A. ULMER          W315           *)
(*      ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*      TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                    *)
(*     REVISED: 06/19/86          B. A.  ULMER         W315           *)
(*      CHANGE PARAMETERS TO DETRUL AND EXCEPTION LIST TO MARK LIST   *)
(*                                                                    *)
(*     REVISED: 01/13/85          E. D. SHREVE         W315           *)
(*      CHANGED TO INITIALIZE A LIST POSITION VARIABLE                *)
(*                                                                    *)
(*     REVISED: 12/30/85          B. A. ULMER          W315           *)
(*      CHANGE TO READ SORT LIST IN REVERSE ORDER                     *)
(*                                                                    *)
(*     REVISED: 07/11/85          B. A. ULMER          W315           *)
(*      ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*      PURPOSES                                                      *)
(*                                                                    *)
(*     REVISED: 05/15/85          B. A. ULMER          W315           *)
(*      FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                   *)
(*                                                                    *)
(*     ORIGINATED: 08/21/84       C. J. SAMPLE         W315           *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(**)
(* END %INCLUDE MAEDTI. *)
```

```
%PAGE
(* %INCLUDE MAEDTS. *)
(**)
  PROCEDURE MAEDTS(CONST KEY1:ANYKEY;VAR KEYDL:LISTKEY;
      VAR KEYEL:LISTKEY;VAR KEYML:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*       TEST DELETE AN ENTITY OR LIST OF ENTITIES, AND RETURN THREE       *)
(*       LISTS.                                                            *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE TEST         *)
(*                           DELETED                                       *)
(*        KEYDL          O   LIST OF ENTITIES WHICH WOULD BE DELETED       *)
(*                           OR MARKED FOR DELETE BY MAED                  *)
(*        KEYEL          O   LIST OF ENTITIES WHICH WOULD NOT BE           *)
(*                           DELETED BY MAED                               *)
(*        KEYML          O   LIST OF ENTITIES WHICH WOULD BE MARKED_       *)
(*                           FOR_DELET BY MAED                             *)
(*        RC             O   EXTERNAL RETURN CODE                          *)
(*                           = 0  OK RETURN CODE                           *)
(*                           < 0  WARNING                                  *)
(*                           > 0  CRITICAL ERROR                           *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*       SIMILAR TO MAEDT, EXCEPT THREE LISTS ARE RETURNED. KEYDL AND      *)
(*       KEYML CAN BE SUBMITTED TO DIRECTLY DELETE AND MARK ENTITIES       *)
(*       USING MAS DELETE ROUTINES THAT DO NOT CHECK THE DELETE RULES.     *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED:  05/01/86          B. A. ULMER            W315              *)
(*      ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION      *)
(*      TO USER RECOGNIZEABLE FORM                                         *)
(*                                                                         *)
```

```
(*    ORIGINATED: 04/22/86       E. D. SHREVE          W315           *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(**)
(* END %INCLUDE MAEDTS *)
```

```
%PAGE
(* %INCLUDE MAEGKN *)
(**)
  PROCEDURE MAEGKN(CONST KEYE:ENTKEY;VAR KIND:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        RETRIEVE THE KIND VALUE OF AN ENTITY.                     *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        KEYE           I   KEY OF AN ENTITY                       *)
(*        KIND           O   KIND VALUE OF THE ENTITY (INTEGER)     *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                    *)
(*                           > 0  CRITICAL ERROR                    *)
(*                           < 0  WARNING                           *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*      NONE                                                        *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                   *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        ACCESS THE KIND VALUE FROM THE ENTITY ADB AND RETURN IT.  *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*        NONE                                                      *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
(*    REVISED: 05/01/86          B. A. ULMER              W315      *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                   *)
(*                                                                  *)
(*    REVISED: 07/11/85          B. A. ULMER              W315      *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                     *)
(*                                                                  *)
(*    ORIGINATED: 03/25/85       E. D. SHREVE             W315      *)
(*                                                                  *)
(*END-------------------------------------------------------------*)
(* END %INCLUDE MAEGKN *)
```

```
%PAGE
(* %INCLUDE MAEGTK *)
(**)
  PROCEDURE MAEGTK(CONST KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       RETRIEVE THE ENTITY BLOCK WHICH CORRESPONDS TO KEYE.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   KEY OD TH ENETITY TO BE RETRIEVED       *)
(*        ENTDEF         O   APPLICATION DATA ASSOCIATED WITH KEYE   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        APPLICATION PROVIDES ENTITY KEY.  MAS WILL RETRIEVE THE    *)
(*        ENTITY                                                     *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA    *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                            *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 11/15/84          D. J. KERCHNER              FRMI    *)
(*     CHECK FOR VALID ENTITY KEY    IF NOT RETURN RC < 0            *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MAEKND *)
(**)
PROCEDURE   MAEKND(CONST KNDPOS:LISTINDX;VAR KNDVAL:ORD_KIND;
            VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       TO RETURN A 'KIND' VALUE FROM THE LIST OF KINDS IN THE      *)
(*       WORKING-FORM MODEL.                                         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        KNDPOS          I   SEQUENCE # OF THE KIND VALUE REQUESTED *)
(*        KNDVAL          O   KIND VALUE AT THE 'KNDPOS' POSITION    *)
(*        RC              O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK RETURN CODE                    *)
(*                            = 1  YOU BLEW IT                       *)
(*                            = 2  THE ROUTINE BLEW IT               *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*     NDSREM                                                        *)
(*        KEY             I   KEY OF THE ROOT ELEMENT                *)
(*                            MUST BE PROVIDED                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       RETRIEVES THE 'KIND' VALUE STORED AT THE 'KNDPOS' POSITION  *)
(*       IN THE STD_ARRAY OF THE SCH_ROOT ADB.                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER          W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    ORIGINATED: 10/26/84       E. D. SHREVE              FRMI     *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                               *)
(*-------------------------------------------------------------------*)
```

```
(*    DATA STRUCTURES/MAJOR VARIABLES:                                    *)
(*--------------------------------------------------------------------------*)
(*                                                                          *)
(*END-------------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAEKND *)
(**)
```

```
%PAGE
(* %INCLUDE MAERST  *)
(**)
  PROCEDURE MAERST(CONST FLGNAME:NAMTYP; VAR RC:EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        RESET THE GIVEN FLAG IN ALL ENTITIES IN THE WORKING FORM   *)
(*        MODEL                                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        FLGNAME        I   THE NAME OF THE FLAG WHICH WILL BE RESET *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NDSREM                                                       *)
(*        VAR1      I   VAR1 NAME MUST BE FILLED, CHARACTER DATA     *)
(*                      MUST BE PROVIDED                             *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    ORIGINATED: 08/12/85       B. A. ULMER                 FRMI    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                               *)
(*-------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                               *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE MAERST  *)
```

```
%PAGE
(* %INCLUDE MAESVL. *)
(**)
  PROCEDURE MAESVL(CONST KEY1:ENTKEY;VAR ISET:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       FIND THE CURRENT BINARY SWITCH SETTING OF AN ENTITY.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEY1           I   KEY OF THE ENTITY WHOSE SETTING IS TO BE *)
(*                           DETERMINED                              *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE INPUT KEY MUST BE AN ENTITY KEY.  IF THE SWITCH IS      *)
(*       TRUE, THEN THE VALUE "1" IS RETURNED.  IF THE SWITCH IS     *)
(*       FALSE, THEN THE VALUE "0" IS RETURNED.                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA    *)
(*     ADDED A CHECK FOR NIL POINTER FOR KEY1                        *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MAESWA *)
(**)
  PROCEDURE MAESWA(VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       SETS THE PROCESS BIT 'OFF' IN ALL ENTITIES IN THE MODEL.    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        RC              0   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK                                *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NDSREM                                                       *)
(*        KEY       I   KEY OF THE MODEL ROOT ELEMENT                *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        EACH ENTRY IN THE SCHEMA-ROOT CONSTITUENT LIST IS READ.    *)
(*        IF IT IS AN INSTANCE_COLLECTOR NODE, THEN EACH ENTITY      *)
(*        ON THE CONSTITUENT LIST OF THE COLLECTOR NODE IS READ      *)
(*        AND THE ADB.SYSUSE FIELD IS SET TO 'TRUE'.                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDTION  *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    ORIGINATED: 02/06/85 CCWW  E. D. SHREVE                FRMI    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
```

```
%PAGE                                                                    *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE MAESWA *)
(**)
```

```
%PAGE
(* %INCLUDE MAESWT. *)
(**)
  PROCEDURE MAESWT(CONST KEY1:ANYKEY;CONST ISWT:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       SET AN ENTITY SWITCH OR THE SWITCHES FOR EACH ENTITY IN A   *)
(*       LIST AS REQUESTED BY THE USER.                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   KEY OF THE ENTITY WHOSE SWITCH IS TO BE *)
(*                           SET OR KEY OF THE LIST ALL OF WHOSE     *)
(*                           ENTITY SWITCHES ARE TO BE SET           *)
(*        ISWT           I   SWITCH VALUE REQUESTED                  *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE TYPE OF KEY IS CHECKED FOR.                             *)
(*       IF AN ENTITY, THEN THE ENTITY'S SWITCH IS RESET.            *)
(*       IF A LIST, THEN EACH ENTITY ON THE LIST HAS ITS SWITCH      *)
(*       RESET.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI   *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MAEU. *)
(**)
  PROCEDURE MAEU(CONST KEY1:ANYKEY;VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST OF USER ENTITY REFERENCES.                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES FOR WHICH A  *)
(*                           LIST OF DIRECT USERS IS REQUESTED       *)
(*        PARM2          O   LIST OF USER REFERENCES                 *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       A NEW LIST, KEY2, IS CREATED THAT CONTAINS THE LIST OF      *)
(*       DIRECT USERS. IF KEY1 IS AN ENTITY KEY, THE DIRECT USERS    *)
(*       OF KEY1 ARE PLACED IN THE LIST. IF KEY1 IS A LISTKEY, THE   *)
(*       DIRECT USERS OF ALL ENTITIES IN THE LIST ARE PLACED INTO     *)
(*       KEY2.                                                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER          W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315           *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    REVISED: 05/15/85          B. A. ULMER          W315          *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                  *)
(*                                                                  *)
(*    REVISED: 08/14/86          K. M. ROSS           W315          *)
(*     ADDED A CHECK FOR NIL POINTER FOR KEY1                       *)
(*                                                                  *)
(*    ORIGINATED: 06/21/84       D. J. KERCHNER       W315          *)
(*                                                                  *)
(*------------------------------------------------------------------*)
%PAGE                                                               *)
(**)
(* END %INCLUDE MAEU. *)
```

```
%PAGE
(* %INCLUDE MAEUD *)
(**)
  PROCEDURE MAEUD(VAR KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       UPDATE THE ENTITY BLOCK CORRESPONDING TO A KEY.             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   KEY OF THE ENTITY TO BE UODATED         *)
(*        ENTDEF         I   APPLICATION DATA ASSOCIATED WITH KEYE   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        CALL REVNODM                                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       IT IS ILLEGAL FOR THE APPLICATION TO CHANGE KIND ON UPDATE. *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 10/11/84          D. J. KERCHNER              FRMI    *)
(*     UPDATED THE INCLUDE DOCUMENTATION                             *)
(*                                                                   *)
```

```
(*    REVISED: 10/04/84          E. D. SHREVE                FRMI     *)
(*     CHANGED THE DECLARATION FOR KEYE AND ENTDEF TO VAR              *)
(*                                                                     *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA     *)
(*     ADDED A CHECK FOR NIL POINTER FOR KEY1                          *)
(*                                                                     *)
```

```
%PAGE
(* %INCLUDE MAEUI *)
(**)
  PROCEDURE MAEUI(CONST KEY1:ANYKEY;VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATION LIST OF INCLUSIVE USER ENTITIES.      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O      DESCRIPTION                               *)
(*       ====     ===      ===========                               *)
(*       KEY1      I       KEY OF AN ENTITY OR A LIST.               *)
(*       KEY2      O       RETURNED KEY OF THE APPLICATION LIST.     *)
(*       RC        O       FUNCTION RETURN CODE.                     *)
(*                         = 0   GOOD RETURN                         *)
(*                         > 0   CRITICAL ERROR                      *)
(*                         < 0   WARNING                             *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360, 370, 43XX                         *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       MODEL ACCESS SOFTWARE INTERFACE PROCEDURE                   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       KEY2 IS CREATED (EMPTY LIST).                               *)
(*       IF KEY1 IS AN ENTITY, THEN THE INCLUSIVE USER LIST OF KEY1  *)
(*       WILL BE COPIED INTO KEY2.                                   *)
(*       IF KEY1 IS A LIST KEY, THEN THE INCLUSIVE USER LISTS OF     *)
(*       EACH ENTITY WILL BE COPIED INTO KEY2.                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 05/01/86        B. A. ULMER          W315         *)
(*                ADDED TO CNVOSP TO CONVERT AN "OUT OF SPACE"       *)
(*                CONDITION TO USER RECOGNIZEABLE FORM               *)
(*       REVISED: 11/08/85        B. A. ULMER          W315         *)
(*                FIX BUG DEALING WITH PREVIOUS FIX                  *)
(*       REVISED: 11/08/85        B. A. ULMER          W315         *)
(*                NOT ALLOW ENTITIES THAT ARE ON THE APPLICATION     *)
(*                INPUT LIST TO BE ON THE APPLICATION OUTPUT LIST    *)
(*                (FIX THE INCONSISTENCIES IN THE PROCESSING)        *)
(*       REVISED: 07/11/85        B. A. ULMER          W315         *)
(*                ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING    *)
(*                AND DEBUGGING PURPOSES                             *)
(*       REVISED: 05/15/85        B. A. ULMER          W315         *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING        *)
```

```
(*       REVISED: 04/29/85        E. D. SHREVE          W315        *)
(*                TO USE MAS INTERNAL PROCESS FLAG (MAPROB)         *)
(*       REVISED: 08/14/86        K. M. ROSS            W315        *)
(*                ADDED A NIL POINTER CHECK FOR KEY1                *)
(*       ORIGINATED: 06/13/84     D. J. KERCHNER        W315        *)
(*                                                                  *)
(**)
(* END %INCLUDE MAEUI *)
```

```
%PAGE
(* %INCLUDE MAEUIK *)
(**)
  PROCEDURE MAEUIK(CONST KEY1:ANYKEY;CONST ENTKIND:ORD_KIND;
      VAR KEY2:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST OF INCLUSIVE USERS BY KIND.                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O    DESCRIPTION                                 *)
(*       ====     ===    ===========                                 *)
(*       KEY1      I     THE KEY OF AN ENTITY OR A LIST OF ENTITIES  *)
(*                       WHOSE INCLUSIVE USERS ARE TO BE SEARCHED    *)
(*                       FOR THE SPECIFIED KIND.                     *)
(*       KIND      I     THE KIND CODE OF AN ENTITY OR AN ENTITY     *)
(*                       CLASS.                                      *)
(*       KEY2      O     THE KEY OF THE LIST WHICH WILL CONTAIN ALL  *)
(*                       ENTITIES OF THE SPECIFIED KIND FOUND WITHIN *)
(*                       THE INCLUSIVE USERS OF KEY1.                *)
(*       RC        O     THE FUNCTION RETURN CODE.                   *)
(*                       = 0   GOOD RETURN                           *)
(*                       > 0   CRITICAL ERROR                        *)
(*                       < 0   WARNING                               *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360, 370, 43XX                         *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       FOR THE GIVEN ENTKEY OR LISTKEY EXPAND ITS USERS INCLU      *)
(*       SIVLEY. FOR EACH MEMBER OF THE EXPANDED LIST WHOSE KIND     *)
(*       MATCHES THE KIND VALUE DESIRED ADD IT TO THE LIST POINTED   *)
(*       TO BY KEY2                                                  *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED: 05/01/86       B.A. ULMER              W315        *)
(*                ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE"*)
(*                CONDITION TO USER RECOGNIZEABLE FORM               *)
(*       REVISED: 07/11/85       B.A. ULMER              W315        *)
(*                ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING    *)
(*                AND DEBUGGING                                      *)
```

```
(*       REVISED: 05/15/85       B.A. ULMER              W315        *)
(*                FIX INCONSISTENCY IN OUTPUT LIST PROCESSING        *)
(*       REVISED: 04/29/85       E.D. SHREVE             W315        *)
(*                TO USE THE INTERNAL MAS PROCESS FLAG (MAPROB)      *)
(*       REVISED: 08/14/86       K.M. ROSS               W315        *)
(*                ADDED A NIL POINTER CHECK FOR KEY1                 *)
(*       ORIGINATED: 08/20/84    R. A. MCCLUSKEY         W315        *)
(*-----------------------------------------------------------------*)
(**)
(* END %INCLUDE MAEUIK *)
```

```
%PAGE
(* %INCLUDE MAEUSR *)
(**)
  PROCEDURE MAEUSR(CONST KEYE:ENTKEY; VAR UEXIST:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DETERMINES IF AN ENTITY HAS ANY USERS.                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   ENTITY KEY                              *)
(*        UEXIST         O   INTEGER VALUE INDICATING IF KEYE HAS    *)
(*                           ANY USERS.                              *)
(*                           = 0 NO USERS EXIST -                    *)
(*                           = 1 USERS EXIST                         *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING MESSAGE                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        EVALUATES THE USER POINTER IN THE ENTITY BLOCK FOR A NIL.   *)
(*        IF NIL, THEN NO USERS EXIST.                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        THIS PROCEDURE DEVELOPED SPECIFICALLY FOR THE IDB PACKAGE  *)
(*        BUT IS FUNCTIONAL FOR ALL APPLICATIONS.                    *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                         *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 04/07/85          B. A. ULMER                         *)
(*     CHANGED TO CHECK FOR THE SYSTEM LIST HAVING NO ENTRIES -  IF IT*)
(*     DOES, THEN NO USERS EXIST                                     *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/86          B. A. ULMER                                  *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING         *)
(*     PURPOSES                                                              *)
(*                                                                           *)
(*    ORIGINATED: 03/25/85       E. D. SHREVE                                 *)
(*                                                                           *)
(*                                                                           *)
(*END-------------------------------------------------------------------------*)
(* END %INCLUDE MAEUSR *)
```

```
%PAGE
(* %INCLUDE MAEUXQ *)
(**)
  PROCEDURE MAEUXQ(CONST KEY1:ANYKEY;VAR DATAREC:BLKDATA;
      CONST PROCNAME:ROUTINE;VAR KEY2:LISTKEY;VAR RCC:EXT_RET_CODE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       EXECUTE A PROCEDURE ON THE USERS OF AN ENTITY, OR LIST      *)
(*       OF ENTITIES.  IF AN OUTPUT LIST IS NOT PASSED, CONSTRUCT ONE *)
(*       IN ORDER TO PUT ENTITIES ON IT AS DETERMINED BY THE         *)
(*       APPLICATION PROCEDURE.                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENITIY OR LIST OF ENTITIES WHOSE USERS  *)
(*                           ARE TO BE PROCESSED                     *)
(*        DATAREC       I/O  APPLICATION DEFINED DATA STRUCTURE WHICH *)
(*                           EITHER SUPPLIES OR RECIEVES VALUES      *)
(*                           OPERATED ON BY THE APPLICATION PROCEDURE *)
(*        PROC           I   ENTRY POINT OF APPLICATION DEFINED      *)
(*                           PROCEDURE                               *)
(*        KEY2           O   KEY OF THE LIST CREATED                 *)
(*                           FOR THIS ROUTINE                        *)
(*        RCC            O   USER DEFINED PROCEDURE RETURN CODE      *)
(*                           = 0,1 OK RETURN CODE                    *)
(*                           = 2-7 PROCEDURE WARNING CODE            *)
(*                           = 8-15 PROCEDURE ERROR CODE             *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*       THE USER SENDS IN THE NECESSARY INFORMATION, THEN THIS        *)
(*       ROUTINE REFERENCES THE USER'S SPECIFIED PROCEDURE TO ACT      *)
(*       UPON THE INFORMATION HE HAS SUPPLIED TO THE PROCEDURE.        *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
(*    REVISED: 09/09/86          B. A. ULMER          DBMA            *)
(*     FIX PROBLEM WITH DELETING EMPTY PASSED IN APPL LIST            *)
(*                                                                     *)
(*    ORIGINATED: 06/16/86       B. A. ULMER          W315            *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
%PAGE                                                                 *)
(**)
(* END %INCLUDE MAEUXQ *)
```

```
%PAGE
(* %INCLUDE MAEXEQ *)
(**)
  PROCEDURE MAEXEQ(CONST KEY1:ANYKEY;VAR DATAREC:BLKDATA;
      CONST PROCNAME:ROUTINE;VAR RCC:EXT_RET_CODE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF THE ENTITY OR APPLICATION LIST*)
(*                           OF ENTITIES TO BE PROCESSED             *)
(*        DATAREC       I/O  THE APLLICATION DEFINED DATA STRUCTURE   *)
(*                           WHICH EITHER SUPPLIES OR RECEIVES VALUES *)
(*                           OPERATED ON BY THE APPLICATION USER      *)
(*                           DEFINED PROCEDURE                        *)
(*        PROCNAME       I   THE NAME OF THE USER DEFINED PROCEDURE   *)
(*        RCC            O   THE USER DEFINED PROCEDURE'S RETURN CODE *)
(*                           RCC < 0 & RCC > 15  PROC_OUT_OF_RANGE    *)
(*                           RCC=> 0 & RCC=<  7  CONTINUE PROCESSING  *)
(*                           RCC=> 8 & RCC=< 15  PROC_CODE_ERROR      *)
(*        RC             O   EXTERNAL RETURN CODE                     *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE USER SENDS IN THE NECESSARY INFORMATION, THEN THIS      *)
(*       ROUTINE REFERENCES THE USER'S SPECIFIED PROCEDURE TO ACT    *)
(*       UPON THE INFORMATION HE HAS SUPPLIED TO THE PROCEDURE.      *)
(*       THE PROCEDURE RETURNS ITS OWN RETURN CODE TO THE USER       *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 05/01/86          B. A.ULMER                  FRMI      *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                      *)
(*                                                                     *)
(*    REVISED: 01/20/86          B. A.ULMER                  FRMI      *)
(*     ADD NEW CAPABILITY TO ALLOW READING LIST IN REVERSE IN ORDER   *)
(*     TO PROCESS                                                      *)
(*                                                                     *)
(*    REVISED: 07/11/85          B. A.ULMER                  FRMI      *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                        *)
(*                                                                     *)
(*    REVISED: 03/06/85          B. A.ULMER                  FRMI      *)
(*     FIXED APPLICATION LIST PROBLEM                                  *)
(*                                                                     *)
(*    REVISED: 11/28/84          D. J. KERCHNER              FRMI      *)
(*     CHANGED MANNER OF ACCESSING USER DEFINED PROCEDURE - NOW        *)
(*     ACCESSED VIA ASSEMBLER CSECT PASASM                             *)
(*                                                                     *)
(*    ORIGINATED: 04/11/84       D. J. KERCHNER              FRMI      *)
(*                                                                     *)
(*-------------------------------------------------------------------*)
%PAGE                                                                  *)
(*-------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                 *)
(*-------------------------------------------------------------------*)
(*                                                                     *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE MAEXEQ *)
(**)
```

```
%PAGE
(* %INCLUDE MAINIT. *)
(**)
  PROCEDURE MAINIT(VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       INITIALIZE THE MAS NETWORK.                                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER         W315            *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*    REVISED: 07/11/85          B. A. ULMER         W315            *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*    REVISED: 05/21/85          B. A. ULMER         W315            *)
(*     ADD CALL TO INITIALIZE THE APPLICATION ACCESSIBLE FLAG TABLE  *)
(*                                                                   *)
(*    ORIGINATED: 03/08/84       D. J. KERCHNER      W315            *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                               *)
%PRINT ON
(* END %INCLUDE MAINIT *)
```

```
%PAGE
(* %INCLUDE MAKCNT *)
(**)
  PROCEDURE MAKCNT(CONST KINDX:INTEGER;VAR COUNT:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         DETERMINE THE NUMBER OF ENTITIES IN THE WORKING FORM MODEL *)
(*         OF A SPECIFIED KIND                                       *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                            *)
(*         ====          ===  ===========                            *)
(*         KINDX          I   KIND VALUE FOR WHICH A COUNT IS TO BE  *)
(*                            DETERMINED                             *)
(*         COUNT          O   NUMBER OF ENITITES IN THE MODEL OF THE *)
(*                            SPECIFIED KIND                         *)
(*         RC             O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK RETURN CODE                    *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         MODEL ACCESS SOFTWARE INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         DETERMINE IF THE KIND SPECIFIED IS IN THE WORKING FORM    *)
(*         MODEL.  IF SO, RETURN THE LENGTH VALUE OF THE CONSTITUENT *)
(*         LIST FOR THE COUNT.  IF NOT, RETURN A VALUE OF ZERO FOR   *)
(*         THE COUNT.                                                *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: 05/01/86          B. A. ULMER              W315      *)
(*      ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*      TO USER RECOGNIZEABLE FORM                                   *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/85          B. A. ULMER                 W315      *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDi ING AND DEBUGGING  *)
(*     PURPOSES                                                        *)
(*                                                                     *)
(*    ORIGINATED: 05/10/85       B. A. ULMER                 W315      *)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MAKCNT *)
```

```
%PAGE
(* %INCLUDE MAKILL. *)
(**)
  PROCEDURE MAKILL(VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DELETE THE WORKING FORM MODEL.                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        RC              O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK RETURN CODE                    *)
(*                            < 0  WARNING                           *)
(*                            > 0  CRITICAL ERROR                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NDSGVR                                                       *)
(*        LIST_OF_ROOTS  0  POINTER TO LIST OF ROOTS                 *)
(*        STACK_OF_LISTS 0  POINTER TO STACK_OF_ROOTS                *)
(*      NDSREM                                                       *)
(*        KEY            0  POINTER TO THE WORKING FORM ROOT NODE    *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                   *)
(*        THIS VERSION USED WITH THE MAS MEMORY MANAGER.             *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DELETES THE WORKING FORM USING PROCEDURE 'NDSREL'.         *)
(*        RESETS POINTERS IN THE COMMONS TO NIL.                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        THIS VERSION FOR USE WITH THE MAS MEMORY MANAGER. THE OLD  *)
(*        VERSION MUST BE USED IF THE PASCAL MEMORY MANAGER IS USED. *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 W315    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 W315    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    REVISED: 04/05/85          E. D. SHREVE                 W315     *)
(*     CHANGED TO DELETE THE WORKING FORM USING 'NDSREL'.               *)
(*                                                                      *)
(*    ORIGINATED: 02/02/84       D. KERCHNER                  K315     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*END-----------------------------------------------------------------*)
(**)
%PRINT ON
(* END %INCLUDE MAKILL *)
```

```
%PAGE
(* %INCLUDE MAKXEQ *)
(**)
  PROCEDURE MAKXEQ(CONST KIND:ORD_KIND;VAR DATAREC:BLKDATA;
      CONST PROCNAME:ROUTINE;VAR RCC:EXT_RET_CODE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*   $FUNCTION                                                           *)
(*       EXECUTE A PROCEDURE ON ALL ENTITIES OF A SPECIFIED KIND.        *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS                                           *)
(*       NAME      I/O      DESCRIPTION                                  *)
(*       ====      ===      ===========                                  *)
(*       KIND       I     KIND VALUE OF THE ENTITIES TO BE PROCESSED.    *)
(*       DATAREC    I     THE APPLICATION DEFINED DATA STRUCTURE WHICH   *)
(*                        EITHER SUPPLIES OR RECEIVES VALUES OPERATED    *)
(*                        ON BY THE APPLICATION DEFINED PROCEDURE.       *)
(*       PROCNAME   I     THE NAME OF THE USER DEFINED PROCEDURE.        *)
(*       DATAREC    O     THE DATA STRUCTURE THAT RESULTS FROM USING     *)
(*                        THE USER DEFINED PROCEDURE.                    *)
(*       RCC        O     THE USER DEFINED PROCEDURE'S RETURN CODE.      *)
(*       RC         O     THE FUNCTION RETURN CODE.                      *)
(*                        =0  EXPECTED RESULT                            *)
(*                        >0  CRITICAL ERROR                             *)
(*                        <0  WARNING                                    *)
(*    $COMMONS:                                                          *)
(*       NONE                                                            *)
(*                                                                       *)
(*    $ENVIRONMENT:                                                      *)
(*       LANGUAGE: IBM PASCAL                                            *)
(*       HARDWARE SYSTEM: IBM 360/370                                    *)
(*                                                                       *)
(*    $EXECUTION PROCEDURE:                                              *)
(*       MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                        *)
(*                                                                       *)
(*    $PROCESSING DESCRIPTION:                                           *)
(*       THE CONSITIUENT LIST OF THE INPUT 'KIND' INSTANCE COLLECTOR     *)
(*       IS READ IN LIFO ORDER AND THE INPUT PROCEDURE IS CALLED         *)
(*       WITH EACH ENTRY IN THE CONSTITUENT LIST.                        *)
(*                                                                       *)
(*    $COMMENTS:                                                         *)
(*       THE ROUTINE PASASM IS CALLED TO PROVIDE A METHOD OF PASSING     *)
(*       ARGUMMENTS FROM A FORTRAN ROUTINE.                              *)
(*       THE LIST IS READ IN LIFO ORDER IN CASE THE INPUT PROCEDURE      *)
(*       DELETES ENTITIES THAT AFFECT THE LIST BEING READ. WITH THE      *)
(*       LIFO ORDER, THE LIST POSITION IS NOT AFFECTED.                  *)
(*                                                                       *)
```

```
(*    $CHANGE CONTROL:                                                    *)
(*       REVISED:  05/01/86  B. A. ULMER                                  *)
(*        ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE"             *)
(*        CONDITION TO USER RECOGNIZEABLE FORM                            *)
(*       REVISED:  07/29/85  B. A. ULMER                                  *)
(*        FIX LOCAL LIST PROBLEM                                          *)
(*       REVISED:  07/11/85  B. A. ULMER                                  *)
(*        ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND             *)
(*        DEBUGGING PURPOSES                                              *)
(*       REVISED:  03/27/85  E. SHREVE                                    *)
(*        TO READ THE LIST IN LIFO ORDER                                  *)
(*       REVISED:  03/11/85  B. ULMER                                     *)
(*        FIX PROBLEM OF LIST POSITION.                                   *)
(*       REVISED:  02/18/85  B. ULMER                                     *)
(*        CHANGED THE STRUCTURE OF THE INTERNAL ITEM FOR IMPLEMENT-       *)
(*        ATION OF THE CRB                                                *)
(*       ORIGINATED:  01/20/85   E. SHREVE                                *)
(*-----------------------------------------------------------------------*)
(**)
(* END %INCLUDE MAKXEQ *)
```

```
%PAGE
(* %INCLUDE MAL *)
(**)
  PROCEDURE MAL(VAR KEYL:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*       CREATE AN EMPTY LIST.                                             *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        RC             O   EXTERNAL RETURN CODE                         *)
(*                           = 0  OK                                       *)
(*                           > 0  CRITICAL ERROR                           *)
(*                           < 0  WARNING                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: 05/01/86          B. A.ULMER              FRMI              *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION       *)
(*     TO USER RECOGNIZEABLE FORM                                          *)
(*                                                                         *)
```

```
%PAGE
(* %INCLUDE MALAND. *)
(**)
  PROCEDURE MALAND(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
     VAR KEY3:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATION LIST OF ENTITIES COMMON TO TWO INPUT  *)
(*       LISTS.                                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES WHICH WILL BE *)
(*                           'ANDED' - IF ENTITY, USE CONSTITUENT LIST*)
(*        KEY2           I   ENTITY OR LIST OF ENTITIES WHICH WILL BE *)
(*                           'ANDED' - IF ENTITY, USE CONSTITUENT LIST*)
(*        KEY3           O   LIST OF ENTITIES COMMON TO KEY1 AND KEY2 *)
(*        RC             O   EXTERNAL RETURN CODE                     *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE INPUT LIST KEY1 WILL BE COMPARED WITH THE INPUT LIST    *)
(*       KEY2.  THOSE ENTITIES WHICH APPEAR ON KEY1 AND KEY2 WILL    *)
(*       BE PUT ON THE OUTPUT LIST KEY3.                             *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER          W315          *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN :OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315          *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    REVISED: 08/14/86          K. M. ROSS          W315            *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                            *)
(*                                                                   *)
(*    ORIGINATED: 03/09/84       D. J. KERCHNER      W315            *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                                *)
(**)
(* END %INCLUDE MALAND. *)
```

```
%PAGE
(* %INCLUDE MALATC *)
(**)
  PROCEDURE MALATC(VAR KEY1:ANYKEY;CONST KEY2:ANYKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*       APPEND AN ENTITY OR LIST (KEY2) TO AN ENTITY OR LIST (KEY1). *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        KEY1           I   THE KEY OF THE ENTITY OR LIST OF ENTITIES*)
(*                           TO WHICH KEY2 IS APPENDED               *)
(*        KEY2           I   THE KEY OF THE ENTITY OR LIST OF ENTITIES*)
(*                           TO BE APPENDED RO KEY1                  *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 AND KEY2 ARE BOTH ENTITIES, THEN                    *)
(*          KEY2 IS ADDED TO THE CONSTITUENT LIST OF KEY1.           *)
(*       IF KEY1 IS AN ENTITY AND KEY2 IS A LIST, THEN               *)
(*          ALL ENTITIES OF KEY2 ARE ADDED TO THE CONSTITUENT LIST   *)
(*          OF KEY1.                                                 *)
(*       IF KEY1 IS A LIST AND KEY2 IS AN ENTITY, THEN               *)
(*          KEY2 IS ADDED TO THE END OF KEY1.                        *)
(*       IF KEY1 AND KEY2 ARE BOTH LISTS, THEN                       *)
(*          ALL ENTITIES OF KEY2 ARE ADDED TO THE END OF KEY1.       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    REVISED: 10/11/84          D. J. KERCHNER               FRMI     *)
(*     INPUT PARAMETER KEY2 CHANGED TO VAR FROM CONST FOR COMPATA-    *)
(*     BILITY WITH DEC VAX SYSTEM  - UPDATE DOC                       *)
(*                                                                    *)
```

```
%PAGE
(* %INCLUDE MALCPY. *)
(**)
  PROCEDURE MALCPY(CONST KEY1:LISTKEY;VAR KEY2:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         MAKE A COPY OF A LIST.                                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                            *)
(*         ====          ===  ===========                            *)
(*         KEY1           I   THE KEY OF THE LIST TO BE COPIED       *)
(*         KEY2           I   THE KEY OF THE NEW LIST THAT WILL CONTAIN*)
(*                            A COPY OF KEY1                         *)
(*         RC             O   EXTERNAL RETURN CODE                   *)
(*                            = 0  OK                                *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         MODEL ACCESS SOFTWARE INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: 05/01/86          B. A. ULMER                FRMI    *)
(*      ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*      TO USER RECOGNIZEABLE FORM                                   *)
(*                                                                   *)
(*     REVISED: 07/11/85          B. A. ULMER                FRMI    *)
(*      ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*      PURPOSES                                                     *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALD. *)
(**)
  PROCEDURE MALD(CONST KEY1:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       DELETE AN APPLICATION LIST.                                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF THE APPLICATION LIST TO BE   *)
(*                           DELETED                                 *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR -                   *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       1. KEY1 MUST BE A LISTKEY.                                  *)
(*       2. KEY1 IS DELETED AND CAN NOT BE RECOVERED.                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA    *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                            *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALDA *)
(**)
  PROCEDURE MALDA(VAR  RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DELETE ALL APPLICATION LISTS THAT ARE NOT 'LOCKED'.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        RC             O   EXTERNAL RETURN CODE                  *)
(*                           = 0  OK RETURN CODE                    *)
(*                           > 0  CRITICAL ERROR                    *)
(*                           < 0  WARNING MESSAGE                   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NDSGVR                                                       *)
(*        STACK_OF_LISTS    I   KEY OF STACK_OF_LISTS                *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        READS THE STACK_OF_LISTS AND CALLS THE APPROPRIATE ROUTINE *)
(*        TO DELETE ALL LISTS FROM THE LIST_OF_LISTS.  IF THE LIST_  *)
(*        OF_LISTS IS EMPTY, THE SYSTEM LIST IS DISPOSED.            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        ONLY APPLICATION LISTS THAT ARE NOT LOCKED (DELTFLG = 0)   *)
(*        ARE DELETED.                                               *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER              W315      *)
(*       ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION*)
(*       TO USER RECOGNIZEABLE FORM                                  *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER              W315      *)
(*       ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING*)
(*       PURPOSES                                                    *)
(*    REVISED: 04/23/85          E.D. SHREVE              W315      *)
(*       TO DELETE ONLY UN_LOCKED APPLICATION LISTS.                 *)
(*    ORIGINATED: 03/21/84       R. A. MCCLUSKEY          W315      *)
(*                                                                   *)
(*                                                                   *)
(*END---------------------------------------------------------------*)
(* END %INCLUDE MALDA *)
```

```
%PAGE
(* %INCLUDE MALDI *)
(**)
  PROCEDURE MALDI(CONST KEY1:ANYKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DELETE AN APPLICATION LIST AND ALL LISTS AFTER IT THAT ARE  *)
(*        NOT LOCKED.                                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   LIST TO START THE DELETE                *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR -                   *)
(*                           < 0  WARNING MESSAGE                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NDSGVR                                                       *)
(*        STACK_OF_LISTS    I   KEY OF STACK_OF_LISTS                *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        READS THE STACK_OF_LISTS AND CALLS THE APPROPRIATE ROUTINE  *)
(*        TO DELETE ALL LISTS FROM THE LIST_OF_LISTS AFTER A SPECI-   *)
(*        FIED LIST.                                                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        ONLY APPLICATION LISTS THAT ARE NOT LOCKED (DELTFLG = 0)    *)
(*        ARE DELETED.                                               *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                W315     *)
(*       ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION*)
(*       TO USER RECOGNIZEABLE FORM                                  *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                W315     *)
(*       ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING*)
(*       PURPOSES                                                    *)
(*    REVISED: 04/23/85          E.D. SHREVE                W315     *)
(*       TO DELETE ONLY UN_LOCKED APPLICATION LISTS.                 *)
```

```
(*    REVISED: 84/09/27          D. KERCHNER                              *)
(*           CHG TO DECREMENT POSITION FOR READ FROM LIST, CHG TO        *)
(*           CHECK FOR VALID POSITION NUMBER, CHG TO DELETE EACH         *)
(*           EACH ENTITY FROM LIST_OF_LISTS.                             *)
(*    REVISED: 86/08/14          K. ROSS                                  *)
(*           ADDED A NIL POINTER CHECK FOR KEY1                          *)
(*    ORIGINATED: 03/21/84       R. A. MCCLUSKEY            W315          *)
(*                                                                       *)
(*                                                                       *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MALDI *)
```

```
%PAGE
(* %INCLUDE MALFND. *)
(**)
  PROCEDURE MALFND(CONST KEY1:ANYKEY;CONST KEY2:ENTKEY;
      CONST IFIRST:LISTPSTN;VAR IPOS:LISTPSTN;VAR RC:EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*   $FUNCTION                                                     *)
(*       FIND THE POSITION OF AN ENTITY (KEY2) IN AN APPLICATION   *)
(*       LIST (KEY1).  IF KEY1 IS AN ENTITY THEN FIND ITS POSITION *)
(*       IN THE CONSTITUENT LIST OF THAT ENTITY.                   *)
(*                                                                 *)
(*   $DESCRIPTION OF ARGUMENTS:                                    *)
(*       NAME    I/O    DESCRIPTION                                *)
(*       ====    ===    ===========                                *)
(*       KEY1    I      THE KEY OF THE LIST IN WHICH KEY2 IS TO BE *)
(*                      FOUND.                                     *)
(*       KEY2    I      THE KEY OF THE ENTITY TO BE FOUND IN KEY1. *)
(*       IFIRST  I      THE POSITION IN KEY1 WHERE THE FIND        *)
(*                      OPERATION IS TO START.                     *)
(*       IPOS    O      THE POSITION IN KEY1 FWHERE KEY2 IS FOUND. *)
(*       RC      O      THE FUNCTION RETURN CODE.                  *)
(*                                                                 *)
(*   $COMMONS                                                      *)
(*       NONE                                                      *)
(*                                                                 *)
(*   $ENVIRONMENT:                                                 *)
(*       LANGUAGE: IBM PASCAL                                      *)
(*       HARDWARE SYSTEM: IBM 360/370                              *)
(*                                                                 *)
(*   $EXECUTION PROCEDURE:                                         *)
(*       INTERNAL MODEL ACCESS SOFTWARE PROCEDURE                  *)
(*                                                                 *)
(*   $PROCESSING DESCRIPTION:                                      *)
(*       KEY1 IS EITHER AN ENTITY KEY OR A LIST KEY.  IF KEY1 IS A *)
(*       LIST, THEN KEY2 IS FOUND IN THE LIST.  IF KEY1 IS AN ENTITY *)
(*       THEN KEY2 IS FOUND IN THE CONSTITUENT LIST OF KEY1.  KEY2 *)
(*       IS AN ENTITY KEY THAT IS TO BE MATCHED.                   *)
(*       THE SEARCH STARTS AT POSITION IFIRST.  EACH ENTITY IN KEY1 *)
(*       IS CHECKED FOR A MATCH WITH KEY2.  IF MATCHED, THEN THE   *)
(*       POSITION IS RETURNED IN IPOS.  IF NO MATCH, THEN IPOS IS  *)
(*       RETURNED AS ZERO AND THE RETURN CODE SIGNALS AN ERROR.    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*       REVISED:  05/01/86     B. A. ULMER              W315      *)
(*        ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE"      *)
(*        CONDITION TO USER RECOGNIZEABLE FORM                     *)
(*                                                                 *)
```

```
(*       REVISED:  07/11/85     B. A. ULMER                W315       *)
(*        ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND         *)
(*        DEBUGGING PURPOSES                                          *)
(*                                                                    *)
(*       REVISED:  03/25/85     E.D. SHREVE                W315       *)
(*       TO CALL RDLST FROM OUTSIDE THE WHILE LOOP TO SET THE EOL.    *)
(*                                                                    *)
(*       REVISED:  08/14/86     K.M. ROSS                  W315       *)
(*       ADDED A NIL POINTER CHECK FOR KEY1                           *)
(*                                                                    *)
(*    ORIGINATED:  05/07/85           D. KERCHNER          W315       *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE MALFND. *)
```

```
%PAGE
(* %INCLUDE MALGTK. *)
(**)
  PROCEDURE MALGTK(CONST KEY1:ANYKEY;CONST IPOS:INTEGER;
      VAR KEY2:ENTKEY; VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GET THE NTH KEY FROM A LIST.                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF ENTITY OF LIST OF ENTITIES   *)
(*                           WHOSE NTH KEY IS TO BE GOTTEN           *)
(*        IPOS           I   POSITION IN THE LIST WHERE THE TARGET   *)
(*                           ENTRY IS LOCATED                        *)
(*        KEY2           O   THE KSY OF THE ENTITY AT THE NTH POSITION*)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       1.  IF KEY1 IS A LIST, GET THE IPOS ENTRY FROM THE LIST.    *)
(*       2.  IF KEY2 IS AN ENTITY, GET THE IPOS ENTRY IN THE         *)
(*           CONSTITUENT LIST OF KEY1.                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CCNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 08/28/85          B. A. ULMER                 FRMI    *)
(*     CHANGE WHEN KEY2 IS SET TO NIL - BU FIX FOR HANDLING 1ST AND  *)
(*     3RD PARAMETERS AS SAME KEY                                    *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/85          B. A. ULMER                    FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
```

```
%PAGE
(* %INCLUDE MALINS. *)
(**)
  PROCEDURE MALINS(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
      CONST IPOS:INTEGER;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       INSERT AN ENTITY OR LIST INTO A LIST.                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF ENTITY OR LIST OF ENTITIES   *)
(*                           INTO TO WHICH KEY2 IS TO BE INSERTED    *)
(*        KEY2           I   THE KEY OF ENTITY OR LIST OF ENTITIES TO *)
(*                           BE INSERTED INTO KEY1                   *)
(*        IPOS           I   THE POSITION IN KEY1 TO INSERT KEY2     *)
(*                           (NOTE: THE INSERT BEGINS AT IPOS-1)     *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       1. KEY1 AND KEY2 MAY BE LIST OR ENTITY KEYS.                *)
(*       2. IF KEY1 IS AN ENTITY KEY, KEY2 IS INSERTED INTO THE      *)
(*          CONSTITUENT LIST OF KEY1.                                *)
(*       3. IF KEY2 IS A LIST KEY, ALL ENTITIES IN THE LIST ARE      *)
(*          INSERTED INTO KEY1.                                      *)
(*       4. THE INSERT TAKES PLACE STARTING AT THE POSITION 'BEFORE' *)
(*          IPOS IN KEY1.                                            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/85          B. A. ULMER                    FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
```

```
%PAGE
(* %INCLUDE MALK. *)
(**)
  PROCEDURE MALK(CONST KIND:ORD_KIND;VAR KEY1:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST OF ALL ENTITIES OF A SPECIFIED KIND.          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KIND           I   KIND CODE OF A CLASS COLLECTOR OR AN   *)
(*                           INSTANCE COLLECTOR                      *)
(*        KEY1           O   KEY OF THE CREATED LIST OF ENTITIES    *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE ELEMENTS OF THE LIST WILL BE A CONCATENATION  OF THE    *)
(*       CONTENT OF EACH ENTITY CLASS AS THEY ARE ENCOUNTERED IN     *)
(*       THE ENTITY CLASS STRUCTURE.                                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER          W315          *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER REOCGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315          *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
(*    REVISED: 05/15/85          B. A. ULMER          W315          *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                   *)
(*                                                                   *)
```

```
(*    ORIGINATED: 04/24/84       D. J. KERCHNER       W315            *)
(*                                                                     *)
(*-------------------------------------------------------------------*)
%PAGE                                                                  *)
(**)
(* END %INCLUDE MALK. *)
```

```
%PAGE
(* %INCLUDE MALKL. *)
(**)
  PROCEDURE MALKL(CONST KEY1:ANYKEY;CONST KIND:ORD_KIND;
      VAR KEY2:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST OF AN ENTITY KIND WHICH ARE FOUND WITHIN      *)
(*       ANOTHER LIST.                                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF ENTITY OR LIST OF ENTITIES   *)
(*                           WHOSE IMMEDIATE CONSTITUENTS ARE TO BE  *)
(*                           SEARCHED                                *)
(*        KIND           I   THE KIND VALUE OF AN ENTITY OR CLASS    *)
(*        KEY2           O   THE KEY OF THE LIST THAT CONTAINS THE   *)
(*                           SELECTED ENTITIES                       *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       1. IF KEY1 IS AN ENTKEY, THEN ALL CONSTITUENTS OF KEY1      *)
(*          THAT MATCH ON KIND ARE PUT INTO KEY2.                    *)
(*       2. IF KEY1 IS A LISTKEY, THEN ALL ENTITIES ON KEY1 THAT     *)
(*          MATCH ON KIND ARE PUT INTO KEY2.                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 12/10/85          B. A. ULMER                  FRMI     *)
(*     RETURN WARNING WHEN OUTPUT LIST IS NIL                           *)
(*                                                                      *)
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING    *)
(*     PURPOSES                                                         *)
(*                                                                      *)
```

```
%PAGE
(* %INCLUDE MALN *)
(**)
  PROCEDURE MALN(CONST LSIZE:INTEGER;VAR KEYL:LISTKEY;
                 VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN EMPTY LIST OF A SPECIFIED SIZE.                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        LSIZE          I   NUMBER OF ENTITIES IN THE LIST          *)
(*        KEYL           O   INITIALIZED TO KEY OF EMPTY LIST        *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       A NEW APPLICATION LIST WILL BE CREATED, WITH SUFFICIENT     *)
(*       SPACE TO ACCOMODATE 'LSIZE' ENTRIES. ALL ENTRIES ARE        *)
(*       INITIALIZED TO NIL.                                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALNO. *)
(**)
  PROCEDURE MALNO(CONST KEY1:ANYKEY;VAR KOUNT:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       COUNT THE ENTITIES ON A LIST.                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE LIST WHOSE ENTRIES ARE TO BE COUNTED *)
(*        KOUNT          O   THE NUMBER OF ENTRIES IN KEY1           *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS A LIST, RETURN THE NUMBER ON THE LIST.  IF KEY1  *)
(*       IS AN ENTITY, RETURN THE NUMBER OF CONSTITUENTS.            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI   *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALNOT. *)
(**)
  PROCEDURE MALNOT(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
      VAR KEY3:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATION LIST OF ENTITIES IN KEY1 BUT NOT IN   *)
(*       KEY2.                                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES WHCIH WILL BE *)
(*                           'NOTED' - IF ENTITY, USE CONSTITUENT LIST*)
(*        KEY2           I   ENTITY OR LIST OF ENTITIES EHICH WILL BE *)
(*                           'NOTED' - IF ENTITY, USE CONSTITUENT LIST*)
(*        KEY3           O   LIST OF ENTITIES WHICH KEY1 HAS BUT KEY2 *)
(*                           DOES NOT                                *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE KEY1 LIST IS COMPARED TO THE KEY2 LIST.  IF AN ENTITY   *)
(*       IS IN THE KEY1 LIST, THEN IT IS PUT ON THE OUTPUT KEY3      *)
(*       LIST.  THE OUTPUT LIST WILL CONSIST OF ONLY THOSE ENTITIES  *)
(*       FOUND IN KEY1 BUT NOT IN KEY2.                              *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER          W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315           *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    REVISED: 05/15/85          B. A. ULMER          W315          *)
(*     FIX INCONSISTENCY IN OUTPUTLIST PROCESSING                   *)
(*                                                                  *)
(*    REVISED: 08/14/86          K. M. ROSS           W315          *)
(*     ADDED NIL POINTER CHECK FOR KEY1                             *)
(*                                                                  *)
(*    ORIGINATED: 03/09/84       D. J. KERCHNER       W315          *)
(*                                                                  *)
(*------------------------------------------------------------------*)
%PAGE                                                               *)
(**)
(* END %INCLUDE MALNOT. *)
```

```
%PAGE
(* %INCLUDE MALOCK *)
(**)
  PROCEDURE MALOCK(VAR LKEY:LISTKEY;CONST LOCK:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        SET AN APPLICATION LIST FOR DELETE OR NON-DELETE STATUS.   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        LKEY           I   LISTKEY                               *)
(*        LOCK           I   INTEGER VALUE INDICATING LOCK SETTING  *)
(*                           = 0 SET TO 'DELETE'                    *)
(*                           = 1 SET TO 'NON-DELETE'                *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                    *)
(*                           > 0  CRITICAL ERROR                    *)
(*                           < 0  WARNING MESSAGE                   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE.                   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        SETS A FLAG IN THE INPUT LIST TO DELETE OR NON-DELETE.     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        THE DELETE/NON-DELETE STATUS AFFECTS ONLY THE MALDA AND    *)
(*        MALDI INTERFACE ROUTINES.  THESE ROUTINES WILL CHECK THE   *)
(*        STATUS AND NOT DELETE THE LIST IF STATUS = 1.  ALL OTHER   *)
(*        DELETE FUNCTIONS (EG. MALD) DO NOT CHECK THE STATUS WHEN   *)
(*        DELETING.                                                  *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER                         *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                         *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    ORIGINATED: 04/23/85       E. D. SHREVE                          *)
(*                                                                     *)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MALOCK *)
```

```
%PAGE
(* %INCLUDE MALOR *)
(**)
  PROCEDURE MALOR(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
      VAR KEY3:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE AN APPLICATION LIST FROM A BOOLEAN 'OR' ON TWO       *)
(*       INPUT LISTS.                                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES WHICH WILL BE *)
(*                           'ORED' - IF ENTITY, USE CONSTITUENT LIST *)
(*        KEY2           I   ENTITY OR LIST OF ENTITIES WHICH WILL BE *)
(*                           'ORED' - IF ENTITY, USE CONSTITUENT LIST *)
(*        KEY3           O   LIST OF ENITIES WHICH ARE EITHER IN KEY1 *)
(*                           OR KEY2                                 *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       KEY1 AND KEY2 MAY BE EITHER ENTKEYS OR LISTKEYS.            *)
(*       IF KEY1 IS AN ENTITY KEY, THEN ITS CONSTITUENT LIST WILL BE *)
(*       'OR'ED WITH KEY2.                                           *)
(*       IF KEY2 IS AN ENTITY KEY, THEN ITS CONSTITUENT LIST WILL BE *)
(*       'OR'ED WITH KEY1.                                           *)
(*       CREATE AN APPLICATION LIST, KEY3, CONTAINING ALL ENTITIES   *)
(*       IN EITHER OR BOTH OF TWO INPUT LISTS.                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 05/01/86          B. A. ULMER         W315            *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 07/11/85          B. A. ULMER         W315              *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    REVISED: 05/15/85          B. A. ULMER         W315              *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                    *)
(*                                                                    *)
(*    REVISED: 08/14/86          K. M. ROSS          W315              *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                             *)
(*                                                                    *)
(*    ORIGINATED: 03/09/85       D. J. KERCHNER      W315              *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(**)
(* END %INCLUDE MALOR *)
```

```
%PAGE
(* %INCLUDE MALPUT. *)
(**)
  PROCEDURE MALPUT(VAR KEYL:LISTKEY;CONST EKEY:ENTKEY;
      CONST IPOS:INTEGER);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*  **** FOR IDB USE ONLY. 12/15/84  E. SHREVE   ****                *)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       INSERT AN ENTITY INTO THE IDB BIG LIST.                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYL           I   KEY OF THE IDB APPLICATION LIST OF ALL  *)
(*                           ENTITIES                                *)
(*        KEYE           I   KEY OF THE ENTITY TO BE INSERTED        *)
(*        IPOS           I   THE POSITION IN KEYL TO INSERT THE KEY  *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *) 1
```

```
%PAGE
(* %INCLUDE MALRD *)
(**)
  PROCEDURE MALRD(CONST KEY1:ANYKEY;VAR KEY2:ENTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*        READ THE NEXT ENTRY IN A DIRECTED LIST.                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                            *)
(*         ====          ===  ===========                            *)
(*         KEY1           I   THE KEY OF THE DIRECTED LIST TO BE READ *)
(*         KEY2           O   KEY OF THE ENTITY READ                  *)
(*         RC             O   EXTERNAL RETURN CODE                    *)
(*                            = 0  OK                                *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         MODEL ACCESS SOFTWARE INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*        THE LIST IS READ IN THE DIRECTION AS SET BY MALSTF OR      *)
(*        MALSTR.  IF KEY1 IS AN ENTKEY THEN THE NEXT CONSTITUENT    *)
(*        IS READ.  IF KEY1 IS AN  APPLICATION LIST THE NEXT ENTITY  *)
(*        IS READ.                                                   *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: 05/01/86          B. A. ULMER                FRMI    *)
(*      ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*      TO USER RECOGNIZEABLE FORM                                   *)
(*                                                                   *)
(*     REVISED: 07/11/85          B. A. ULMER                FRMI    *)
(*      ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*      PURPOSES                                                     *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALRDE. *)
(**)
  PROCEDURE MALRDE(CONST KEYL:LISTKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REMOVE DUPLICATE ENTRIES IN A LIST.                         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYL           I   THE KEY OF THE LIST WHOSE DUPLICATE     *)
(*                           ENTITIES WILL BE REMOVED                *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       ANY DUPLICATE ENTITIES FOUND IN THE INPUT LIST WILL BE      *)
(*       REMOVED.  THE CHANGE IS MADE IN-PLACE.                      *)
(*       CALLS ELDNL IN THE NDS PACKAGE.                             *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALREP *)
(**)
  PROCEDURE MALREP(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REPLACE A LIST.  IF KEY1 IS AN ENTITY, THEN REPLACE THE     *)
(*       CONSTITUENT LIST OF THAT ENTITY.                            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF THE ENTITY OR LIST OF ENTITIES*)
(*                           TO BE REPLACED  -  IF AN ENTITY, THEN   *)
(*                           USE THE CONSTITUENT LIST OF KEY1        *)
(*        KEY2           I   THE KEY OF THE ENTITY OR LIST OF ENTITIES*)
(*                           TO REPLACE KEY1  -  IF AN ENTITY, THEN  *)
(*                           USE THE CONSTITUENT LIST OF KEY1        *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       KEY1 MAY BE EITHER AN ENTITY KEY OR A LIST KEY.             *)
(*       IF KEY1 IS A LIST KEY, THEN KEY2 REPLACES KEY1.             *)
(*       IF KEY1 IS AN ENTITY, THEN THE CONSTITUENT LIST OF KEY1 IS  *)
(*       REPLACED BY KEY2.                                           *)
(*       KEY2 MAY BE EITHER AN ENTITY KEY OR A LIST KEY.             *)
(*       IF KEY2 IS A LIST KEY, THEN KEY2 REPLACES KEY1.             *)
(*       IF KEY2 IS AN ENTITY, THEN THE CONSTITUENT LIST OF KEY2     *)
(*       REPLACES KEY1.                                              *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI       *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION     *)
(*     TO USER RECOGNIZEABLE FORM                                        *)
(*                                                                       *)
(*    REVISED: 10/30/85          B. A. ULMER                 FRMI       *)
(*     TAKE OUT CHECK OF DELETE RULES                                    *)
(*                                                                       *)
(*    REVISED: 09/05/85          B. A. ULMER                 FRMI       *)
(*     ADDED NEW PARAMETERS TO FNDURUL FOR THE TWO NEW DELETE RULES.     *)
(*                                                                       *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI       *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING     *)
(*     PURPOSES                                                          *)
(*                                                                       *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA       *)
(*     ADDED A NIL POINTER CHECK KEY1                                    *)
(*                                                                       *)
```

```
%PAGE
(* %INCLUDE MALRMV *)
(**)
  PROCEDURE MALRMV(CONST KEY1:ANYKEY;CONST IPOS:LISTINDX;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REMOVE AN ENTITY FROM A LIST.                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF ENTITY OF LIST OF ENTITIES   *)
(*                           WHICH AN ENITTY WILL BE REMOVED         *)
(*        IPOS           I   THE POSITION IN KEY1 LIST WHICH THE     *)
(*                           ENTITY WILL BE REMOVED                  *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        1. KEY1 MAY BE AN ENTITY OR LIST KEY.                      *)
(*        2. IF KEY1 IS A LIST KEY, THEN AN ENTITY IS REMOVED        *)
(*           FROM THE LIST.                                          *)
(*        3. IF KEY1 IS AN ENTITY KEY, THEN AN ENTITY IS REMOVED     *)
(*           FROM THE CONSTITUENT LIST OF KEY1. THE DELETE RULES     *)
(*           FOR KEY1 ARE TESTED TO INSURE THAT THE REMOVAL FROM     *)
(*           KEY1 IS PERMITTED.                                      *)
(*        4. IPOS IS THE POSITION NUMBER OF THE ENTITY TO BE         *)
(*           REMOVED.                                                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                  FRMI   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AAN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    REVISED: 09/05/85          B. A. ULMER                  FRMI     *)
(*     ADDED NEW PARAMETERS TO FNDURUL FOR THE TWO NEW DELETE RULES.  *)
(*                                                                    *)
(*    REVISED: 07/11/85          B. A. ULMER                  FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    REVISED: 10/31/84          D. J. KERCHNER               FRMI     *)
(*     INITIALIZED THE POSITION TO AN ARBITRARY #100 FOR THE DELRLST  *)
(*     AND DELPLST CALLS                                              *)
(*                                                                    *)
(*    REVISED: 02/06/85          E. D. SHREVE                 FRMI     *)
(*     TEST FOR INVALID IPOS ARGUMENT                                 *)
(*                                                                    *)
(*    ORIGINATED: 06/28/84       E. D. SHREVE                 FRMI     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MALRMV *)
(**)
```

```
%PAGE
(* %INCLUDE MALROR *)
(**)
  PROCEDURE MALROR(VAR KEYL:LISTKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*       REORDER THE APPLICATION LIST IN USER CONSTITUENT ORDER            *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        KEYL          I/O  LIST TO BE REORDERED                          *)
(*        RC             O   EXTERNAL RETURN CODE                          *)
(*                           = 0  OK                                       *)
(*                           > 0  CRITICAL ERROR                           *)
(*                           < 0  WARNING                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
%PAGE
(* %INCLUDE MALRPL. *)
(**)
  PROCEDURE MALRPL(CONST KEY1:ANYKEY;CONST KEY2:ENTKEY;
      CONST IPOS:LISTPSTN;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REPLACE AN ENTITY IN A LIST.                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF AN ENTITY OR LIST OF ENTITIES *)
(*                           WHICH WILL BE REPLACED                  *)
(*        KEY2           I   KEY OF THE ENTITY TO BE MOVED INTO KEY1  *)
(*        IPOS           I   THE POSITION IN KEY1 WHERE KEY2 IS TO BE *)
(*                           PLACED                                  *)
(*        RC             O   EXTERNAL RETURN CODE                     *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       1. KEY1 MAY BE AN ENTITY OR LIST KEY.                       *)
(*                                                                   *)
(*       2. IF KEY1 IS AN ENTITY KEY, THEN KEY2 WILL REPLACE THE     *)
(*          ENTITY AT IPOS IN KEY1'S CONSTITUENT LIST.               *)
(*                                                                   *)
(*       3. THE KEY AT IPOS POSITION IN THE LIST IS REPLACED.        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       IF THE ENTITY BEING REPLACED IN A CONSTITUENT LIST IS       *)
(*       'MARKED FOR DELETE', THEN AN ATTEMPT WILL BE MADE TO        *)
(*       DELETE THE ENTITY.                                          *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
```

```
(*                                                                    *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI     *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                     *)
(*                                                                    *)
(*    REVISED: 03/20/86          B. A. ULMER                 FRMI     *)
(*     CHANGE DELRLST TO INDLST AND DELPLST WHEN TRYING TO REMOVE     *)
(*     THE USER KEY FROM THE REPLACED ENTITY'S USER LIST              *)
(*                                                                    *)
(*    REVISED: 08/  /85          L. J. BEHAN                 FRMI     *)
(*     ADD NEW PARAMETER TO DELRUL, DELENTY TO HANDLE APPLICATION     *)
(*     LIST POSITION PROBLEM                                          *)
(*                                                                    *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
```

```
%PAGE
(* %INCLUDE MALRVS.*)
(**)
  PROCEDURE MALRVS(VAR KEYA:ANYKEY; VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        REVERSE THE ORDER OF THE INPUT LIST.                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYA          I/O  A LIST OR ENTITY KEY                    *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        IF THE INPUT KEY IS AN APPLICATION LIST, THE LIST IS       *)
(*        REVERSED.  IF THE INPUT IS AN ENTITY, THE CONSTITUENT      *)
(*        LIST OF THE ENTITY IS REVERSED.                            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED:  05/01/86          B. A. ULMER                 W315   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    ORIGINATED: 04/11/86 MAS2  E. D. SHREVE                 W315   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*END---------------------------------------------------------------*)
(* END %INCLUDE MALRVS. *)
```

```
%PAGE
(* %INCLUDE MALSRT *)
(**)
  PROCEDURE MALSRT(CONST KEY:ANYKEY; CONST PROCNAME:ROUTINE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:  GIVEN THE USER DEFINED ORDER FUNCTION THE LIST PASSED*)
(*   IN AS INPUT WILL BE SORTED USING THIS FUNCTION                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEY            I   THE KEY OF THE ENTITY OR APPLICATION LIST*)
(*                           OF ENTITIES TO BE SORTED               *)
(*        PROCNAME       I   THE NAME OF THE USER DEFINED FUNCTION  *)
(*                           FOR THE ORDERING OF THE LIST           *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                *)
(*                           > 0  CRITICAL ERROR                    *)
(*                           < 0  WARNING                           *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                   *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*       THE USER SENDS IN THE ORDER FUNCTION, THEN THIS ROUTINES   *)
(*       REFERENCES THE USER DEFINED FUNCTION TO ACT UPON THE ENTITIES*)
(*       BEING SORTED.                                              *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
(*    REVISED: MM/DD/YY          I M THECHANGER            GROUP    *)
(*     REASON FOR CHANGING THE ROUTINE                              *)
(*                                                                  *)
(*    ORIGINATED: 04/  /86       B. A. ULMER               FRMI     *)
```

```
(*                                                                    *)
(*-------------------- ----------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MALSRT *)
(**)
```

```
%PAGE
(* %INCLUDE MALSTF *)
(**)
  PROCEDURE MALSTF(CONST KEY1:ANYKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       INITALIZE FOR READING A DIRECTED LIST IN FORWARD ORDER.     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEY1           I   THE KEY OF AN ENTITY OR LIST OF ENTITIES *)
(*                           WHOSE READ DIRECTION WILL BE SET TO     *)
(*                           FORWARD                                 *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS AN ENTITY THEN THE CONSTITUENT LIST OF KEY1      *)
(*       WILL BE INITIALIZED.  IF KEY1 IS A LISTKEY THEN THE LIST    *)
(*       POINTED TO WILL BE INITIALIZED.  IN EITHER CASE THE         *)
(*       <.POSITION> ELEMEMT IS SET TO THE VALUE 1 AND THE           *)
(*       <.DIRECTION> ELEMENT IS SET TO THE VALUE <FORWARD>.         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       USES NDS FUNCTION LSTLNM.                                   *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
```

```
(*                                                                  *)
(*    REVISED: 08/14/86          K. M. ROSS                 DBMA   *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                           *)
(*                                                                  *)
```

```
%PAGE
(* %INCLUDE MALSTR *)
(**)
  PROCEDURE MALSTR(CONST KEY1:ANYKEY;VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       INITALIZE FOR READING A DIRECTED LIST IN REVERSE ORDER.     *)
(*       MAS INTERFACE PACKAGE.                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF AN ENTITY OR LIST OF ENTITIES *)
(*                           WHSOE READ DIRECTION WILL BE SET TO     *)
(*                           REVERSE                                 *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       IF KEY1 IS AN ENTITY THEN THE CONSTITUENT LIST OF KEY1      *)
(*       WILL BE INITIALIZED.  IF KEY1 IS A LISTKEY THEN THE LIST    *)
(*       POINTED TO WILL BE INITIALIZED.  IN EITHER CASE THE         *)
(*       <.POSITION> ELEMEMT IS SET TO THE LENGTH OF THE LIST AND    *)
(*        THE <.DIRECTION> ELEMENT IS SET TO THE VALUE <REVERSE>.    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       USES NDS FUNCTION LSTLNM.                                   *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 08/14/86          K. M. ROSS                  DBMA    *)
(*     ADDED A NIL POINTER CHECK FOR KEY1                            *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MALXEQ *)
(**)
  PROCEDURE MALXEQ(CONST KEY1:ANYKEY;VAR DATAREC:BLKDATA;
      CONST PROCNAME:ROUTINE;VAR KEY2:LISTKEY;VAR RCC:EXT_RET_CODE;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       EXECUTE A PROCEDURE ON AN ENTITY, OR A LIST OF ENTITIES.    *)
(*       CONSTRUCT AN OUTPUT LIST OF ENTITIES AS DETERMINED BY THE   *)
(*       APPLICATION PROCEDURE.                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENITIY OR LIST OF ENTITIES TO BE        *)
(*                           PROCESSED                               *)
(*        DATAREC       I/O  APPLICATION DEFINED DATA STRUCTURE WHICH *)
(*                           EITHER SUPPLIES OR RECIEVES VALUES      *)
(*                           OPERATED ON BY THE APPLICATION PROCEDURE *)
(*        PROC           I   ENTRY POINT OF APPLICATION DEFINED      *)
(*                           PROCEDURE                               *)
(*        KEY2           O   KEY OF THE LIST CREATED                 *)
(*                           FOR THIS ROUTINE                        *)
(*        RCC            O   USER DEFINED PROCEDURE RETURN CODE      *)
(*                           = 0,1 OK RETURN CODE                    *)
(*                           = 2-7 PROCEDURE WARNING CODE            *)
(*                           = 8-15 PROCEDURE ERROR CODE             *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       THE USER SENDS IN THE NECESSARY INFORMATION, THEN THIS      *)
(*       ROUTINE REFERENCES THE USER'S SPECIFIED PROCEDURE TO ACT    *)
(*       UPON THE INFORMATION HE HAS SUPPLIED TO THE PROCEDURE.      *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*    REVISED: 05/01/86          B. A. ULMER          W315               *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION     *)
(*     TTO USER RECOGNIZEABLE FORM                                       *)
(*                                                                       *)
(*    REVISED: 01/20/86          B. A. ULMER          W315               *)
(*     ADD CAPABILITY TO READ THE INPUT LIST IN REVERSE IN ORDER         *)
(*     TO PROCESS                                                        *)
(*                                                                       *)
(*    REVISED: 07/11/85          B. A. ULMER          W315               *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING     *)
(*     PURPOSES                                                          *)
(*                                                                       *)
(*    REVISED: 05/15/85          B. A. ULMER          W315               *)
(*     FIX INCONSISTENCY IN OUTPUT LIST PROCESSING                       *)
(*                                                                       *)
(*    REVISED: 03/06/85          B. A. ULMER          W315               *)
(*     FIX APPLICATION LIST PROBLEM                                      *)
(*                                                                       *)
(*    REVISED: 11/28/84          D. J.KERCHNER        W315               *)
(*     MALXEQ MADE FROTRAN CALLABLE BY USING INTERMEDIATE ASSEMBLER      *)
(*     ROUTINE (PASASM)                                                  *)
(*                                                                       *)
(*    ORIGINATED: 04/24/84       D. J. KERCHNER       W315               *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(**)
(* END %INCLUDE MALXEQ *)
```

```
%PAGE
(* %INCLUDE MAQURY  *)
(**)
  PROCEDURE MAQURY(CONST KEY1:ENTKEY; CONST FLGNAME:NAMTYP; VAR
      FLGVAL:INTEGER; VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DESCRIPTION OF WHAT THIS ROUTINE DOES.                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   ENTITY WHOSE SPECIFIED FLAG VALUE IS TO *)
(*                           DETERMINED                              *)
(*        FLGNAME        I   FLAG NAME (STRING(6))                   *)
(*        FLGVAL         O   VALUE OF THE SPECIFIED FLAG             *)
(*                           =1 TRUE                                 *)
(*                           =0 FALSE                                *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                     *)
(*                           < 0  WARNING                            *)
(*                           > 0  CRITICAL ERROR                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DETERMINE WHICH APPLICATION ACCESSIBLE FLAG'S VALUE IS TO  *)
(*        BE GOTTEN AND THEN GET THE FLAG VALUE                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER          W315           *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER          W315           *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
(*    ORIGINATED: 05/21/85       B. A. ULMER          W315          *)
(*                                                                  *)
(*------------------------------------------------------------------*)
%PAGE                                                               *)
(* END %INCLUDE MAQURY  *)
```

```
%PAGE
(* %INCLUDE MASALOC *)
(**)
  PROCEDURE MASALOC(CONST SIZE:INTEGER; VAR REGVAL:POINTER;
      VAR RC:INTEGER);FORTRAN;
(**)
(* END %INCLUDE MASALOC *)
```

```
%PAGE
(* %INCLUDE MASDSP *)
(**)
PROCEDURE MASDSP(  VAR ENT_PTR:    POINTER;
                 CONST TYPE_SIZE:  INTEGER);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       DISPOSE OF A MAS DYNAAMICALLY ALLOCATED MEMORY AREA.        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        TYPE_SIZE      I   THE SIZE OF THE AREA TO BE DISPOSED    *)
(*        ENT_PTR        I   POINTER TO THE MEMORY AREA TO BE DISPOSED*)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      $PCMGR     HOLDS THE DESCRIPTORS FOR THE MAS MEMORY AREAS.   *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DELETE A BLOCK AND COMBINE IT WITH ANY CONTIGIOUS BLOCKS   *)
(*        OF FREED MEMORY.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 02/06/86          B. A. ULMER                 FRMI    *)
(*     ADDED CODE TO HANDLE WHEN THE 8K OVERFLOW BLOCK NEEDS FREED   *)
(*     (JUST REMOVE IF FROM THE BLOCK CHAIN AND SET OVERFLOW FLAG TO *)
(*     FALSE)                                                        *)
(*                                                                   *)
(*    REVISED: 08/  /85          B. A. ULMER                 FRMI    *)
(*     FIX BUG DEALING WITH THE PRESENCE OF AN INFINITE LOOP         *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ELIMINATE THE LEAVE AND MAX FUNCTIONS FOR BETTER COMPATABILITY*)
(*     WITH THE DEC VAX                                              *)
(*                                                                   *)
```

```
(*    ORIGINATED: 12/10/84       J. J. JOHNSON                FRMI     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE MASDSP *)
(**)
```

```
%PAGE
(* %INCLUDE MASMSZ *)
(**)
  PROCEDURE MASMSZ(VAR MODSIZ:INTEGER; VAR FRESIZ:INTEGER;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        RETURNS THE ACTUAL MODEL SPACE USED AND THE AMOUNT OF      *)
(*        FREE SPACE IN THE ALLOCATED MEMORY BLOCKS OF THE MODEL.    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        MODSIZ         O   TOTAL BYTES OF USED MODEL SPACE        *)
(*        FRESIZ         O   TOTAL BYTES OF FREE SPACE IN THE       *)
(*                           ALLOCATED MODEL BLOCKS.                 *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK RETURN CODE                     *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*        USED ONLY WITH THE MAS MEMORY MANAGER. CAN NOT BE USED     *)
(*        WITH THE PASCAL MEMORY MANAGER.                            *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        CALLS THE INTERNAL MAS ROUTINES THAT CALCULATE FREESPACE   *)
(*        AND MODEL SPACE USING THA MAS MEMORY MANAGER CONTROL BLOCKS.*)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        IF THIS PROCEDURE IS TO BE USED WITH THE PASCAL MEMORY     *)
(*        MANAGER, THEN A SPECIAL PROCEDURE 'NDSFCT' IS REQUIRED.    *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED:  05/01/86          B. A. ULMER                 W315   *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*    ORIGINATED: 04/09/85       E. SHREVE                    W315     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE MASMSZ *)
```

```
%PAGE
(* %INCLUDE MASNEW *)
(**)
PROCEDURE MASNEW(  VAR ENT_PTR:    POINTER;
                 CONST TYPE_SIZE:  INTEGER;
                 VAR   RR:         RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       ALLOCATES A NEW DYNAMIC MEMORY AREA FOR MAS ELEMENTS.       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        TYP_SIZE       I   THE SIZE OF THE MEMORY REGION REQUIRED  *)
(*        ENT_PTR        O   POINTER TO THE AREA OBTAINED            *)
(*        RR             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      $PCMGR    HOLDS THE DESCRIPTORS FOR THE MAS MEMORY SPACE.    *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ATTEMPTS TO LOCATE A FREE SPACE, STARTING AT THE FIRST     *)
(*        ALLOCATED REGION, AND CONTINUES THRU ALL ALLOCATED REGIONS. *)
(*        IF FOUND, IT REMOVES THE REGION FROM THE FREE SPACE CHAIN.  *)
(*        IF NO SPACE EXISTS, IT ALLOCATES A NEW REGION AND CONNECTS  *)
(*        THE NEW REGION TO THE LAST.                                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 02/06/86          B. A. ULMER                 FRMI    *)
(*     ADDED CODE TO HANDLE A FAILURE ON GETMAIN IN ROUTINE MASALOC  *)
(*     AND PROCESSING OF 8K OVERFLOW BLOCK                           *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ELIMINATE THE LEAVE AND MAX FUNCTIONS TO BETTER COMPATIBILITY *)
(*     WITH THE DEC VAX                                              *)
(*                                                                   *)
```

```
(*    ORIGINATED: 12/10/84       J. J. JOHNSON                FMRI     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                    *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE MASNEW *)
(**)
```

```
%PAGE
(* %INCLUDE MASOVR *)
(**)
PROCEDURE MASOVR( CONST $IZE: INTEGER; VAR ENT_PTR: POINTER;
                  VAR OSPACE: $CBP);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*                                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      $PCMGR     HOLDS THE DESCRIPTORS FOR THE MAS MEMORY AREAS.   *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ELIMINATE THE LEAVE AND MAX FUNCTIONS FOR BETTER COMPATABILITY *)
(*     WITH THE DEC VAX                                              *)
(*                                                                   *)
(*    ORIGINATED:  3/21/86       B. A. ULMER                 FRMI    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
%PAGE                                                                *)
(*-------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                               *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE MASDSP *)
(**)
```

```
%PAGE                                                                   00010000
(* %INCLUDE MAUPDT  *)                                                  00020000
(**)                                                                    00030000
  PROCEDURE MAUPDT(VAR KEY1:ANYKEY; CONST FLGNAME:NAMTYP; CONST         00093005
      FLGVAL:INTEGER; VAR RC:EXT_RET_CODE);SUBPROGRAM;                  00094000
(**)                                                                    00096000
(*-------------------------------------------------------------------*)00097000
(*                                                                   *)00098000
(*   $FUNCTION:                                                      *)00099000
(*        UPDATE A SPECIFIED APPLICATION ACCESSIBLE FLAG VALUE       *)00099107
(*                                                                   *)00099200
(*   $DESCRIPTION OF ARGUMENTS:                                      *)00099300
(*        NAME          I/O  DESCRIPTION                             *)00099400
(*        ====          ===  ===========                             *)00099500
(*        KEY1           I   ENTITY OR LIST OF ENTITIES WHOSE        *)00099604
(*                           SPECIFIED FLAG VALUE IS TO BE UPDATED   *)00099704
(*        FLGNAME        I   FLAG NAME  (STRING(6))                  *)00099804
(*        FLGVAL         I   VALUE TO BE USED WHEN UPDATING THE FLAG *)00099904
(*                           = 1 TRUE                                *)00100004
(*                           = 0 FALSE                               *)00100107
(*        RC             O   EXTERNAL RETURN CODE                    *)00100200
(*                           = 0  OK RETURN CODE                     *)00100300
(*                           < 0  WARNING                            *)00100404
(*                           > 0  CRITICAL ERROR                     *)00100500
(*                                                                   *)00100800
(*   $COMMONS:                                                       *)00100900
(*                                                                   *)00101500
(*   $ENVIRONMENT:                                                   *)00101600
(*        LANGUAGE: IBM PASCAL                                       *)00101700
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)00101800
(*                                                                   *)00102300
(*   $EXECUTION PROCEDURE:                                           *)00102400
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)00102500
(*                                                                   *)00102800
(*   $PROCESSING DESCRIPTION:                                        *)00102900
(*        DETERMINE WHICH OF THE APPLICATION ACCESSIBLE FLAGS IS TO BE*)00103007
(*        UPDATED AND THEN UPDATE IT WITH THE INPUT VALUE            *)00103104
(*                                                                   *)00103204
(*   $COMMENTS:                                                      *)00103300
(*                                                                   *)00103500
(*   $CHANGE CONTROL:                                                *)00103600
(*    REVISED: 08/21/85          B. A. ULMER                 FRMI    *)00104508
(*     CHANGED TO NOT ALLOW APPLICATION TO SET AN ENTITY FOR MARK    *)00104608
(*     DELETE                                                        *)00104708
(*                                                                   *)00104800
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)00104908
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)00105008
(*     PURPOSES                                                      *)00105108
(*                                                                   *)00105208
```

```
(*    ORIGINATED: 05/21/85       B. A. ULMER                  FRMI     *)00105304
(*                                                                     *)00105400
(*---------------------------------------------------------------------*)00105500
%PAGE                                                                  *)00105600
(* END %INCLUDE MAUPDT  *)                                              00630000
```

```
%PAGE
(**)
  PROCEDURE MIDBD(VAR KEY1:ANYKEY; VAR RC:EXT_RET_CODE);SUBPROGRAM;
%PAGE                                                                  *)
(* %INCLUDE MIDBD. *)
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   WARNING:  FOR IDB USE ONLY                                       *)
(*   MAY CONTAMINATE MODEL IF USING DELETE WITH NO DELETE RULES       *)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*       DELETE AN ENTITY OR LIST OF ENTITIES BUT DO NOT CONSIDER     *)
(*       THE DELETE RULES                                             *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        KEY1           I   ENTITY OR LIST OF ENTITIES TO BE DELETED *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK RETURN CODE                      *)
(*                           < 0  WARNING                             *)
(*                           > 0  CRITICAL ERROR                      *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                     *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*       IF KEY1 IS AN ENTKEY THEN                                    *)
(*         DELETE THE ENTITY                                          *)
(*       IF KEY1 IS A LISTKEY THEN                                    *)
(*         DELETE EACH ENTITY ON THE LIST                             *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*    REVISED: 05/01/86          B. A. ULMER            W315         *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION  *)
(*     TO USER RECOGNIZEABLE FORM                                     *)
(*                                                                    *)
(*    REVISED: 04/22/86          E. D. SHREVE           W315         *)
(*     CHANGED TO CALL XIELM INSTEAD OF DELENTY TO PERFORM THE DELETE *)
(*     AND CHANGE INPUT TO VAR.                                       *)
(*                                                                    *)
```

```
(*    REVISED: 08/04/85          L. J. BEHAN              W315          *)
(*     ADD NEW PARAMETER TO DELENTY FOR HANDLING OF APPLICATION        *)
(*     LIST POSITION PROBLEM                                           *)
(*                                                                     *)
(*    REVISED: 07/11/85          B. A. ULMER              W315          *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING   *)
(*     PURPOSES                                                        *)
(*                                                                     *)
(*    ORIGINATED: 06/17/85       B. A. ULMER              W315          *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE MIDBD. *)
```

```
%PAGE
(* %INCLUDE MIDBRV *)
(**)
  PROCEDURE MIDBRV(CONST KEY1:ANYKEY;CONST IPOS:LISTINDX;
      VAR RC:EXT_RET_CODE);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*    WARNING: FOR IDB USE ONLY                                      *)
(*    MAY CONTAMINATE MODEL IF USING REMOVE WITHOUT DELETE RULES     *)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       REMOVE AN ENTITY FROM A LIST WITHOUT CONSIDERING THE        *)
(*       DELETE RULES                                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEY1           I   THE KEY OF AN ENTITY OR LIST OF ENTITIES *)
(*                           FROM WHICH AN ENTITIY WILL BE REMOVED   *)
(*        IPOS           I   THE POSITION IN KEY1 FROM WHICH THE     *)
(*                           ENTITIY WILL BE REMOVED                 *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        1. KEY1 MAY BE AN ENTITY OR LIST KEY.                      *)
(*        2. IF KEY1 IS A LIST KEY, THEN AN ENTITY IS REMOVED        *)
(*           FROM THE LIST.                                          *)
(*        3. IF KEY1 IS AN ENTITY KEY, THEN AN ENTITY IS REMOVED     *)
(*           FROM THE CONSTITUENT LIST OF KEY1.                      *)
(*        4. IPOS IS THE POSITION NUMBER OF THE ENTITY TO BE         *)
(*           REMOVED.                                                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 05/01/86          B. A. ULMER                 FRMI    *)
(*     ADDED A CALL TO CNVOSP TO CONVERT AN "OUT OF SPACE" CONDITION *)
(*     TO USER RECOGNIZEABLE FORM                                    *)
(*                                                                   *)
```

```
(*                                                                    *)
(*    REVISED: 12/30/85          B. A. ULMER                 FRMI     *)
(*     CHANGE CALL FROM DELENTY TO DELRUL FOR THE CASE WHEN ENTITY IS *)
(*     MARKED FOR DELETE                                              *)
(*                                                                    *)
(*    REVISED: 08/  /85          L. J. BEHAN                 FRMI     *)
(*     ADD NEW PARAMETERS TO DELENTY FOR HANDLING OF APPLICATION      *)
(*     LIST POSITION PROBLEM                                          *)
(*                                                                    *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI     *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING  *)
(*     PURPOSES                                                       *)
(*                                                                    *)
(*    ORIGINATED: 06/19/85       B. A. ULMER                 FRMI     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MIDBRV *)
(**)
```

```
%PAGE
(* %INCLUDE MOVRLSM *)
(**)
  PROCEDURE MOVRLSM(CONST FROM_LIST:LISTPNTR;
      CONST FROM_POSITION:LISTPSTN;VAR TO_LIST:LISTPNTR;
      CONST TO_POSITION:LISTPSTN;CONST ENTCOUNT:LISTSIZE;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    AUTHOR:  UNKNOWN                  CADD    CREATED: YY/MM/DD CC  *)
(*    VERSION: MAS VER 2                        REVISED: 84/10/11 CC  *)
(*                                                                  *)
(*    FUNCTION:                                                     *)
(*       MOVE ENTITIES BETWEEN SYSTEM LISTS.                        *)
(*                                                                  *)
(*    ENVIRONMENT:                                                  *)
(*        IBM PASCAL LANGUAGE                                       *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                            *)
(*                                                                  *)
(*    DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME     I/O  DESCRIPTION                                 *)
(*        FROM_LIST                                                 *)
(*                 I  POINTER TO A SYSTEM LIST.                     *)
(*        FROM_POSITION                                             *)
(*                 I  THE RELATIVE POSITION OF THE FIRST ENTITY TO  *)
(*                    BE MOVED.                                     *)
(*        TO_LIST  I  POINTER TO A SYSTEM LIST.                     *)
(*        TO_POSITION                                               *)
(*                 I  THE RELATIVE POSITION IN THE LIST TO WHICH    *)
(*                    THE ENTITIES WILL BE MOVED.                   *)
(*        ENTCOUNT I  THE NUMBER OF ENTITIES TO MOVE.               *)
(*        RR       O   ERROR CONDITION RETURN CODE.                 *)
(*                     = 0  NORMAL RETURN CODE.                     *)
(*                     = 14 BAD_LIST_POSITION                       *)
(*                     = 16 BAD_LIST_MOVE_COUNT                     *)
(*                     = 17 BAD_LIST_REFERENCE                      *)
(*                                                                  *)
(*    COMMONS:                                                      *)
(*                                                                  *)
(*    PROCESSING DESCRIPTION:                                       *)
(*       MOVRLSM USES AMPXMOVE A SYSTEM ROUTINE. AMPXMOVE MOVES     *)
(*       DATA FROM MEMORY TO MEMORY (NUMBER OF BYTES TO BE MOVED    *)
(*       HAS TO BE SPECIFIED).                                      *)
(*                                                                  *)
(*    COMMENTS:                                                     *)
(*                                                                  *)
(*    CHANGE CONTROL:                                               *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                       *)
(*           UPDATED DOCUMENTATION.                                 *)
```

```
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED DECLARATION OF 'TO_LIST' TO VAR.                *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MOVRLSM *)
```

```
%PAGE
(* %INCLUDE MRGTLSM. *)
(**)
  PROCEDURE MRGTLSM(VAR LIST1:LISTPNTR;CONST LIST2:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       CONCATENATE THE ENTITIES IN LIST2 TO LIST1.                 *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          LIST1, LIST2 - TWO LIST POINTERS.                        *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MRGTLSM. *)
```

```
%PAGE
(* %INCLUDE MRGTNM. *)
(**)
  PROCEDURE MRGTNM(CONST KEYL1:LISTKEY;CONST KEYL2:LISTKEY;
      VAR RR:RET_REC); EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CONCATENATE THE ENTITIES IN LIST2 TO LIST1.                  *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE1     - KEY OF THE APPLICATION LIST.  IF ENTITY KEY, *)
(*                      THEN USE CONSTITUENT LIST.                    *)
(*          KEYE2     - KEY OF THE APPLICATION LIST TO BE            *)
(*                      CONCATENATED.  IF ENTITY KEY, THEN USE        *)
(*                      CONSTITUENT LIST.                             *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE MRGTNM. *)
```

```
%PAGE
(* %INCLUDE MRKNM. *)
(**)
  PROCEDURE MRKNM(VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       MARK THE STACK OF LISTS SO THAT THE NEXT RELEASE LIST        *)
(*       WILL ONLY DESTROY LISTS CREATED AFTER THIS MARK OPERATION.   *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NONE      -                                               *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE MRKNM. *)
```

```
%PAGE
(* %INCLUDE MSTART. *)
(**)
  PROCEDURE MSTART(CONST ID:INTEGER);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       START STATISTICS GENERATION.                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        ID             I   INDICATION OF THE STATISTICS BEING KEPT *)
(*                           THIS FIELD MUST CORRESPOND TO ID INPUT  *)
(*                           TO MSTOP                                *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       WHEN MSTART IS CALLED, THE INTEGER EQUIVALENT VALUE OF THE  *)
(*       MAS ROUTINE ID IS ENTERED INTO A COMMON FIELD.  ALSO, A     *)
(*       FLAG IS SET TO ON INDICATING THAT THIS PARTICULAR MAS       *)
(*       ROUTINE IS THE ONE CURRENTLY BEING PROCESSED.               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 07/24/86          B. A. ULMER                 FRMI    *)
(*     CHANGE ID_FLAG FIELD OF MSTATUS TO AN INTEGER SO THAT AN APPL. *)
(*     USER CAN KNOW HOW MANY LEVELS HE IS NESTED                    *)
(*                                                                   *)
(*    REVISED: 07/11/85          B. A. ULMER                 FRMI    *)
(*     ADD A NEW PARAMETER TO CNVRR FOR ERROR HANDLING AND DEBUGGING *)
(*     PURPOSES                                                      *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE MSTOP. *)
(**)
  PROCEDURE MSTOP(CONST ID:INTEGER);SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       STOP STATISTICS GENERATION.                                 *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       STATISTICS PACKAGE.                                         *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          ID         - INDICATION OF TYPE OF STATISTICS BEING KEPT. *)
(*                       THIS FIELD MUST CORRESPOND TO ID INPUT TO   *)
(*                       CALL TO MSTART.                             *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          NONE       -                                           *)
(*                                                                   *)
(*    METHOD                                                         *)
(*       WHEN MSTOP IS CALLED, THE INTEGER EQUIVALENT VALUE OF THE   *)
(*       MAS ROUTINE ID IS ENTERED INTO A COMMON FIELD.  ALSO, A     *)
(*       FLAG IS SET TO OFF INDICATING THAT THIS PARTICULAR MAS      *)
(*       ROUTINE IS NO LONGER ACTIVELY BEING PROCESSED, BUT THE ID   *)
(*       WILL INDICATE THAT IT WAS THE LAST ONE CALLED.              *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE MSTOP. *)
```

```
%PAGE
(* %INCLUDE NDSCMM. *)
(**)
  PROCEDURE NDSCMM;EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       DUMMY PROGRAM DEFINES NDSREM COMMON.                         *)
(*       USED AS THE 'SEED' OF THE MAS NDS.                           *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       NETWORK PACKAGE.                                             *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NONE      -                                               *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          NONE      -                                               *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       SYSTEM INCONGRUITIES FORCE NESTING OF DEF WITHIN A           *)
(*       PROCEDURE.                                                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NDSCMM. *)
```

```
%PAGE
(* %INCLUDE NDSFCT *)
(**)
  PROCEDURE NDSFCT(VAR MODSIZ:INTEGER; VAR FRESIZ:INTEGER;
                   VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         COMPUTES THE AMOUNT OF USED MODEL SPACE AND THE AMOUNT OF *)
(*         FREESPACE IN THE ALLOCATED MEMORY BLOCKS.                 *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         MODSIZ         O   TOTAL BYTES OF USED MODEL SPACE       *)
(*         FRESIZ         O   NUMBER OF BYTES OF FREE SPACE.        *)
(*         RR             O   RETURN CODE                           *)
(*                            = 0  OK RETURN CODE                    *)
(*                            > 0  CRITICAL ERROR                    *)
(*                            < 0  WARNING                           *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*        PCMGT                                                      *)
(*         PTR       I    POINTER TO THE 1ST ALLOCATED MEMORY BLOCK. *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE          *)
(*         USED ONLY WITH THE MAS MEMORY MANAGER.  CAN NOT BE USED   *)
(*         WITH THE PASCAL MEMORY MANAGER.                           *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         EACH ALLOCATED BLOCK IS FOUND USING THE BLOCK CHAIN OF THE*)
(*         SPACE CONTROL BLOCK ($CB). THE FREE CHAIN IS USED TO SUM  *)
(*         THE SIZE OF EACH FREED ENTRY.  THE BLOCK SIZES OF ALL     *)
(*         ALLOCATED BLOCKS ARE ALSO TOTALED.                        *)
(*          MODSIZ = TOTAL SPACE ALLOCATED - FREESIZE                *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         THE STRUCTURE OF THE MEMORY MANAGER CONTROL BLOCKS ARE    *)
(*         DESCRIBED IN THE INCLUDE MEMBER 'PCMGT'.                  *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     CHANGED:    07/16/85       B. A. ULMER                 W315  *)
(*     REASON:     CHANGED $PCMGT TO PCMGT FOR VAX COMPATABILITY     *)
(*                                                                   *)
```

```
(*    ORIGINATED: 04/09/85       E. SHREVE                    W315     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*END----------------------------------------------------------------*)
(* END %INCLUDE NDSFCT *)
```

```
(* %INCLUDE NDSGBM. *)
(**)
  PROCEDURE NDSGBM;EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       DUMMY PROCEDURE FOR COMPILE TIME INITIALIZATION OF NDS       *)
(*       GLOBAL AREA.  CONTAINS NDS GLOBAL VARIABLE.                  *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       NETWORK PACKAGE.                                             *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NONE      -                                               *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          NONE      -                                               *)
(*                                                                    *)
(*    COMMENT                                                         *)
(*       DEFINED WITHIN THIS PROCEDURE ARE THE LIST OF ALL NETWORKS   *)
(*       AND THE LIST OF ALL LISTS.                                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NDSGBM. *)
```

```
%PAGE
(* %INCLUDE NDSRML *)
(**)
  PROCEDURE NDSRML;EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        RELEASE ALL MEMORY BLOCKS ALLOCATED TO THE WORKING FORM.   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        NONE                                                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      $PCMGR                                                       *)
(*        PTR      I   POINTER TO THE FIRST ALLOCATED BLOCK.        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381 - MAS PACKAGE USING *)
(*                 THE MODEL ACCESS MEMORY MANAGER.                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*        THIS ROUTINE CAN ONLY BE USED WITH THE MAS MEMORY MANAGER. *)
(*        IF THE PASCAL MEMORY MANAGER IS USED, THE ROUTINE DISPNDM  *)
(*        MUST BE SUBSTITUTED FOR NDSRML.                            *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        BEGINNING WITH THE POINTER IN $PCMGR, EACH MEMORY BLOCK    *)
(*        ALLOCATED TO THE WORKING FORM IS LOCATED AND FREED.        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        THE 1ST WORD OF EACH MEMORY AREA CONTAINS THE POINTER THAT *)
(*        CHAINS ALL WORKING FORM MEMORY AREAS.                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*    REVISED: 07/11/85          B. A. ULMER                 W315    *)
(*     CHANGED $PCMGT TO PCMGT FOR VAX COMPATABILITY                 *)
(*                                                                   *)
(*    ORIGINATED: 04/05/85       E.D.  SHREVE                W315    *)
(*                                                                   *)
```

```
(*-------------------------------------------------------------------*)
%PAGE                                                                *)
(*-------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                               *)
(*        THE INCLUDE MEMBER '$PCMGR' DESCRIBES THE STRUCTURE OF THE *)
(*        CONTROL BLOCKS THAT CONTROL THE MEMORY AREAS AND LINKS THEM*)
(*        TOGETHER.                                                  *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE NDSRML *)
```

```
%PAGE
(* %INCLUDE NEWCRB *)
(**)
  PROCEDURE NEWCRB(VAR CRB:CRBPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       CREATE A CRB                                                *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME    I/O  DESCRIPTION                                    *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS                 *)
(*       RR       O   ERROR CONDITION RETURN CODE                    *)
(*                    = 0  OK RETURN CODE                            *)
(*                    = 1  YOU BLEW IT                               *)
(*                    = 2  THE ROUTINE BLEW IT                       *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      COM1                                                         *)
(*        VAR1      I   VAR1 NAME MUST BE FILLED, CHARACTER DATA     *)
(*                      MUST BE PROVIDED                             *)
(*        VAR2      I   VAR2 MUST BE SPECIFIED                       *)
(*      COM2                                                         *)
(*        VAR3      I   CHARACTER DATA MUST BE SPECIFIED             *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                        *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE   *)
(*           NARRATION ON THE NEXT LINE.                             *)
(*        YY/MM/DD  CCXX  I. M. APERSON                              *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                       *)
```

```
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWCRB *)
```

```
%PAGE
(* %INCLUDE NEWEMM. *)
(**)
  PROCEDURE NEWEMM(VAR KEYE:ENTKEY;CONST FORM:ENTITIES;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CREATE A NEW NDS OBJECT.  FORM DETERMINES WHAT IS CREATED.   *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       ENTITY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          FORM      - THE FORM OF THE ENTITY TO CREATE.             *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          KEYE      - THE POINTER TO THE CREATED ENTITY.            *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*   CHANGE CONTROL:                                                  *)
(*      CHANGED: 12/10/84 J. JOHNSON  - TO CALL 'MASNEW'.             *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWEMM. *)
```

```
%PAGE
(* %INCLUDE NEWIIM *)
(**)
  PROCEDURE NEWIIM(CONST ROOT:ENTKEY;VAR KEYE:ENTKEY;
        VAR ENTDEF:ENTBLOCK;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  UNKNOWN                  CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 2                        REVISED: 84/10/11 CC   *)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A NEW ENTITY AND COPY THE APPLICATION ENTDATA INTO   *)
(*       IT.  CALLING PROCEDURE MUST CONNECT ENTITY TO PROPER POINT  *)
(*       IN NDS.                                                     *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME     I/O  DESCRIPTION                                  *)
(*        ROOT      I   THE NDS INTERNAL ROOT TO BE THE OWNER OF     *)
(*                      THE ENTITY.                                  *)
(*        ENTDEF    I   CONTAINS THE DATA TO BE COPIED INTO THE      *)
(*                      NEW ENTITY.                                  *)
(*        KEYE      O   THE KEY OF THE NEW ENTITY.                   *)
(*        RR        O   ERROR CONDITION RETURN CODE.                 *)
(*                      = 0  NORMAL RETURN CODE.                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       ALLOCATES A NEW T_ENTITY AND CREATS EMPTY USER AND CNSTS    *)
(*       LISTS AND POINTS TO THEM. IT CREATES THE ADB.               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        04/26/85              E. D. SHREVE               W315      *)
(*              TO INITIALIZE THE CRBEXIT AND MAPROB FIELDS          *)
(*                                                                   *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                        *)
(*           UPDATED DOCUMENTATION.                                  *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED DECLARATION OF ENTDEF TO VAR.                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWIIM *)
```

```
%PAGE
(* %INCLUDE NEWLSM. *)
(**)
  PROCEDURE NEWLSM(CONST SIZE:LISTSIZE;VAR POSITION:LISTPSTN;
      VAR LISTREF:LISTPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       LISTREF IS INITIALIZED AND ALLOCATED ENOUGH SPACE TO HOLD   *)
(*       SIZE ENTITIES.  IF ALREADY INITIALIZED, LISTREF IS DELETED  *)
(*       PRIOR TO ALLOCATION OF SPACE.  IF SIZE IS ZERO, NO SPACE    *)
(*       IS ALLOCATED.                                               *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          SIZE      - NUMBER OF ENTITIES TO BE ALLOCATED.          *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          POSITION  - POSITION OF LIST.                            *)
(*          LISTREF   - POINTER TO A SYSTEM LIST WITH SIZE ENTITIES  *)
(*                      ALLOCATED TO IT.                             *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*    CHANGE CONTROL                                                 *)
(*          CHANGED: 12/10/84  J. JOHNSON - TO CALL 'MASNEW'.        *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWLSM. *)
```

```
%PAGE
(* %INCLUDE NEWNDM. *)
(**)
  PROCEDURE NEWNDM(VAR NDSREM:NDS;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CREATE A NEW EMPTY MODEL IN MEMORY.                          *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       NETWORK PACKAGE.                                             *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NONE      -                                               *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          NDSREM    - CONNECTED TO THE NEW NDS.                     *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*    CHANGE CONTROL:                                                 *)
(*      EDS - MAS VERSION 2 - 9/17/84  REMOVE 'MARK' FUNCTION.        *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWNDM. *)
```

```
%PAGE
(* %INCLUDE NEWNM. *)
(**)
  PROCEDURE NEWNM(VAR KEYL:LISTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION                                                      *)
(*       CREATE AN EMPTY APPLICATION LIST.                           *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS                                      *)
(*       NAME          I/O       DESCRIPTION                       *)
(*       ====          ===       ===========                       *)
(*       KEYL           O        KEY OF THE CREATED APPL LIST      *)
(*       RR             O        RETURN CODE                       *)
(*                               =0  GOOD RETURN                   *)
(*                               >0  CRITICAL ERROR                *)
(*                               <0  WARNING                       *)
(*                                                                 *)
(*    $COMMONS                                                     *)
(*      NDSGVM                                                     *)
(*        STACK_OF_LISTS    I   USED TO FIND LIST_OF_LISTS TO ADD  *)
(*                              THE NEW LIST KSY.                  *)
(*                                                                 *)
(*    $ENVIRONMENT:                                                *)
(*        LANGUAGE:  IBM PASCAL                                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                 *)
(*    $EXECUTION PROCEDURE:                                        *)
(*        INTERNAL PROCEDURE OF THE MODEL ACCESS SOFTWARE          *)
(*                                                                 *)
(*    $PROCESSING DESCRIPTION:                                     *)
(*        CREATES A NEW APPLICATION LIST ELEMENT AND ATTACHES IT   *)
(*        TO THE LIST_OF_LISTS.  IT CREATES A NEW SYSTEM LIST THAT *)
(*        IS EMPTY AND ATTACHES IT TO THE APPLICATION LIST ELEMENT. *)
(*        THE FIELDS OF THE ELEMENTS ARE INITIALIZED.              *)
(*                                                                 *)
(*    $CHANGE CONTROL:                                             *)
(*        REVISED: 04/23/85      E.D. SHREVE           W315        *)
(*         CHANGED TO INITIALIZE THE NEW 'DELTFLG' FIELD.          *)
(*                                                                 *)
(*        ORIGINATED:  ORIGINAL NDS PACKAGE                        *)
(*-------------------------------------------------------------------*)
(**)
(*END %INCLUDE NEWNM. *)
```

```
%PAGE
(* %INCLUDE NEWNODE *)
(**)
  PROCEDURE NEWNODE(CONST NDSREM:NDS;VAR KEYE:ENTKEY;
      VAR ENTDEF:ENTBLOCK;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    AUTHOR:  UNKNOWN                 CADD    CREATED: YY/MM/DD CC *)
(*    VERSION: MAS VER 2                       REVISED: 84/10/11 CC *)
(*                                                                  *)
(*    FUNCTION:                                                     *)
(*       CREATE A NEW ENTITY IN THE NDS AND COPY THE APPLICATION    *)
(*       ENTDATA INTO IT.                                           *)
(*                                                                  *)
(*    ENVIRONMENT:                                                  *)
(*        IBM PASCAL LANGUAGE                                       *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                            *)
(*                                                                  *)
(*    DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME    I/O  DESCRIPTION                                  *)
(*        NDSREM   I   THE NDS TO BE THE OWNER OF THE ENTITY.       *)
(*        ENTDEF   I   CONTAINS THE DATA TO BE COPIED INTO THE      *)
(*                     NEW ENTITY.                                  *)
(*        KEYE     O   THE KEY OF THE NEW ENTITY.                   *)
(*        RR       O   ERROR CONDITION RETURN CODE.                 *)
(*                     = 0  NORMAL RETURN CODE.                     *)
(*                                                                  *)
(*    COMMONS:                                                      *)
(*                                                                  *)
(*    PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*    COMMENTS:                                                     *)
(*                                                                  *)
(*    CHANGE CONTROL:                                               *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                       *)
(*           UPDATED DOCUMENTATION.                                 *)
(*        84/10/04  MAS VER 2  E. D. CHREVE                         *)
(*           CHANGED DECLARATION FOR ENTDEF TO VAR.                 *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWNODE *)
```

```
%PAGE
(* %INCLUDE NEWNSC. *)
(**)
  PROCEDURE NEWNSC(VAR ROOT:ENTKEY;VAR KEYE:ENTKEY;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       CREATE AN EMPTY SCHEMA CLASS COLLECTOR ATTACHED TO THE     *)
(*       SCHEMA ROOT.                                               *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       SCHEMA PACKAGE.                                            *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED     *)
(*                      INSTANCE COLLECTOR WILL BE ATTACHED.        *)
(*                                                                  *)
(*       OUTPUT                                                     *)
(*          KEYE      - KEY OF THE CREATED CLASS COLLECTOR ENTITY. *)
(*          RR        - THE FUNCTION RETURN RECORD.                *)
(*                                                                  *)
(*    METHOD                                                        *)
(*       THIS PROGRAM IS CALLED FOR NO OTHER REASON THAN TO AVOID   *)
(*       PASCAL TYPE CHECKING BY USING A DIFFERENT DEFINITION OF    *)
(*       ENTBLOCK.                                                  *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWNSC. *)
```

```
%PAGE
(* %INCLUDE NEWNSI. *)
(**)
  PROCEDURE NEWNSI(VAR ROOT:ENTKEY;VAR KEYE:ENTKEY;VAR RR:RET_REC);
      EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       CREATE AN EMPTY SCHEMA INSTANCE COLLECTOR ATTACHED TO THE   *)
(*       SCHEMA ROOT.                                                *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       SCHEMA PACKAGE.                                             *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED       *)
(*                      INSTANCE COLLECTOR WILL BE ATTACHED.         *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          KEYE      - KEY OF THE CREATED INSTANCE COLLECTOR ENTITY.*)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                   *)
(*    METHOD                                                         *)
(*       THIS PROGRAM IS CALLED FOR NO OTHER REASON THAN TO AVOID    *)
(*       PASCAL TYPE CHECKING BY USING A DIFFERENT DEFINITION OF     *)
(*       ENTBLOCK.                                                   *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWNSI. *)
```

```
%PAGE
(* %INCLUDE NEWNSR. *)
(**)
  PROCEDURE NEWNSR(VAR ROOT:ENTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       CREATE A NEW NULL SCHEMA ROOT AND ATTACH IT TO THE NDS.      *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       SCHEMA PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED       *)
(*                      SCHEMA ROOT WILL BE ATTACHED.                 *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       THIS PROGRAM IS CALLED FOR NO OTHER REASON THAN TO AVOID     *)
(*       PASCAL TYPE CHECKING BY USING A DIFFERENT DEFINITION OF      *)
(*       ENTBLOCK.                                                    *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWNSR. *)
```

```
%PAGE
(* %INCLUDE NEWSADB *)
(**)
  PROCEDURE NEWSADB(CONST SIZE:ENTSIZE;VAR ENTBPNTR:ENTPNTR;
      VAR RR:RET_REC); EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    AUTHOR:  UNKNOWN                 CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 1                       REVISED: 12/10/84      *)
(*                                                                    *)
(*    FUNCTION:                                                       *)
(*       ALLOCATE SPACE FOR DATA TO A SYSTEM UDB.                     *)
(*                                                                    *)
(*    ENVIRONMENT:                                                    *)
(*        IBM PASCAL LANGUAGE                                         *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                              *)
(*                                                                    *)
(*    DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME     I/O  DESCRIPTION                                   *)
(*        SIZE      I   SIZE OF ENTDATA TO BE COPIED.                 *)
(*        ENTBPNTR  O   POINTER TO ENTBLOCK CREATED.                  *)
(*        RR        O   ERROR CONDITION RETURN CODE.                  *)
(*                      = 0  NORMAL RETURN CODE.                      *)
(*                                                                    *)
(*    COMMONS:                                                        *)
(*                                                                    *)
(*    PROCESSING DESCRIPTION:                                         *)
(*       NEWSADB USES THE PASCAL/VS COMPILER SUPPORT ROUTINE AMPXNEW.*)
(*                                                                    *)
(*    COMMENTS:                                                       *)
(*                                                                    *)
(*    CHANGE CONTROL:                                                 *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                         *)
(*           UPDATED DOCUMENTATION.                                   *)
(*        84/12/10  MAS VER 2  J. JOHNSON                             *)
(*           TO CALL MASNEW.                                          *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWSADB *)
```

```
%PAGE
(* %INCLUDE NEWSCHC. *)
(**)
  PROCEDURE NEWSCHC(VAR ROOT:ENTKEY;VAR KEYE:ENTKEY;
      VAR ENTDEF:ENTBLOCK;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       CREATE AN EMPTY SCHEMA CLASS ENTITY ATTACHED TO THE        *)
(*       SCHEMA ROOT.                                               *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       SCHEMA PACKAGE.                                            *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED      *)
(*                      CLASS ENTITY WILL BE ATTACHED.              *)
(*                                                                  *)
(*       OUTPUT                                                     *)
(*          KEYE      - KEY OF THE CREATED ENTITY.                  *)
(*          ENTDEF    - WORK AREA TO BE PASSED TO NEWIIT.           *)
(*          SCH_PTR   - POINTER TO THE CREATED CLASS ENTITY.        *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWSCHC. *)
```

```
%PAGE
(* %INCLUDE NEWSCHI. *)
(**)
  PROCEDURE NEWSCHI(CONST ROOT:ENTKEY;VAR KEYE:ENTKEY;
      VAR ENTDEF:ENTBLOCK;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       CREATE AN EMPTY SCHEMA INSTANCE COLLECTOR ENTITY ATTACHED        *)
(*       TO THE SCHEMA ROOT.                                              *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       SCHEMA PACKAGE.                                                  *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED            *)
(*                      INSTANCE COLLECTOR WILL BE ATTACHED.              *)
(*                                                                        *)
(*       OUTPUT                                                           *)
(*          KEYE      - KEY OF THE INITIALIZED ENTITY.                    *)
(*          ENTDEF    - WORK AREA TO PASS TO NEWIIM.                      *)
(*          SCH_PTR   - POINTER TO THE CREATED INSTANCE COLLECTOR         *)
(*                      ENTITY.                                           *)
(*          RR        - THE FUNCTION RETURN RECORD.                       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWSCHI. *)
```

```
%PAGE
(* %INCLUDE NEWSCHR. *)
(**)
  PROCEDURE NEWSCHR(VAR ROOT:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       CREATE AN EMPTY ROOT COLLECTOR ENTITY ATTACHED TO THE NDS.  *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       SCHEMA PACKAGE.                                             *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          ROOT      - THE INTERNAL ROOT TO WHICH THE CREATED       *)
(*                      SCHEMA_ROOT WILL BE ATTACHED.                *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          ENTDEF    - WORK AREA TO BE PASSED TO NEWIIM.            *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE NEWSCHR. *)
```

```
%PAGE
(* %INCLUDE NODECNM. *)
(**)
  PROCEDURE NODECNM(CONST KEYE:ENTKEY;VAR KEYLOUT:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*       CREATE A LIST WHICH CONTAINS A COPY OF THE ENTITY'S               *)
(*       CONSTITUENT LIST.                                                 *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        KEYE           I   KEY OF THE ENTITY.                            *)
(*        KEYLOUT        O   LIST OF THE ENTITY'S CONSTITUENT ENTITIES     *)
(*        RR             O   THE FUNCTION RETURN RECORD.                   *)
(*                           = 0  OK RETURN CODE                           *)
(*                           = 1  YOU BLEW IT                              *)
(*                           = 2  THE ROUTINE BLEW IT                      *)
(*                           = ?  ERRORS FROM INTERNALLY CALLED            *)
(*                                FUNCTIONS                                *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                          *)
(*                OR                                                       *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE                 *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: 06/28/85 CCXX     B. A. ULMER                  FRMI        *)
(*     CHANGE THE RETURN CODE FROM (END_OF_LIST TO NO_LIST_CREATED)        *)
(*                                                                         *)
```

```
%PAGE
(* %INCLUDE NODEUNM. *)
(**)
  PROCEDURE NODEUNM(CONST KEYE:ENTKEY;VAR KEYLOUT:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       CREATE A LIST WHICH CONTAINS A COPY OF THE ENTITY'S         *)
(*       USER LIST.                                                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE           I   KEY OF THE ENTITY.                      *)
(*        KEYLOUT        O   LIST OF THE ENTITY'S USER ENTITIES       *)
(*        RR             O   THE FUNCTION RETURN RECORD.             *)
(*                           = 0  OK RETURN CODE                     *)
(*                           = 1  YOU BLEW IT                        *)
(*                           = 2  THE ROUTINE BLEW IT                *)
(*                           = ?  ERRORS FROM INTERNALLY CALLED      *)
(*                                FUNCTIONS                          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        MODEL ACCESS SOFTWARE INTERFACE ROUTINE                    *)
(*                OR                                                 *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 06/28/85 CCXX     B. A. ULMER                 FRMI    *)
(*     CHANGE THE RETURN CODE FROM (END_OF_LIST TO NO_LIST_CREATED)  *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE OCOUNT *)
(**)
  PROCEDURE OCOUNT(VAR SIZE:INTEGER);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 86/03/13  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       COUNT THE NUMBER OF TIMES THE OVERFLOW BUFFER HAS BEEN USED *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*       IBM PASCAL LANGUAGE                                         *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.    *)
(*                                                                   *)
(*    EXECUTION PROCEDURE:                                           *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                  *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME     I/O  DESCRIPTION                                   *)
(*       SIZE     I/O  SIZE TO BE STORED IN THE REQUESTED SIZE ARRAY  *)
(*                      IN THE MSTATUS COMMON                        *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*      MSTATUS                                                      *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH      *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.           *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                           *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                      *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                        *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE   *)
(*           NARRATION ON THE NEXT LINE.                             *)
(*        YY/MM/DD  CCXX  I. M. APERSON                              *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                       *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE OCOUNT *)
```

```
%PAGE
(* %INCLUDE ORDRLST. *)
(**)
  PROCEDURE ORDRLST(VAR IN_LIST:LISTPNTR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GIVEN AN APPLICATION LIST OF ENTITIES REORDER THEN SO THAT  *)
(*       THEY ARE IN USER TO CONSTITUENT ORDER                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IN_LIST        I   SYSTEM LIST THAT IS TO BE REORDERED    *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       CREATE A COPY OF IN_LIST IN SRT_LST.                        *)
(*       REPEAT FOR EACH ENTITY OF SRT_LST:                          *)
(*         GET ALL USERS OF I-TH ENTITY OF SRT_LST.                  *)
(*           IF A USER OF SRT_LST(I) APPEARS AT SRT_LST(J) AND I<J   *)
(*           THEN                                                    *)
(*             SWAP SRT_LST(I) AND SRT_LST(J).                       *)
(*           ELSE                                                    *)
(*             GET NEXT SRT_LST(I)                                   *)
(*       UNTIL END OF LIST IN SRT_LST.                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE OSTART *)
(**)
  PROCEDURE OSTART; EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 86/03/13  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                    *)
(*    FUNCTION:                                                       *)
(*       INITIALIZE THE INFORMATION DEALING WITH THE OVERFLOW BUFFER  *)
(*       IN THE MSTATUS COMMON                                        *)
(*                                                                    *)
(*    ENVIRONMENT:                                                    *)
(*       IBM PASCAL LANGUAGE                                          *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.     *)
(*                                                                    *)
(*    EXECUTION PROCEDURE:                                            *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                   *)
(*                                                                    *)
(*    DESCRIPTION OF ARGUMENTS:                                       *)
(*       NAME    I/O  DESCRIPTION                                     *)
(*                                                                    *)
(*    COMMONS:                                                        *)
(*      MSTATUS                                                       *)
(*                                                                    *)
(*    PROCESSING DESCRIPTION:                                         *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH       *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.            *)
(*                                                                    *)
(*    COMMENTS:                                                       *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                     *)
(*                                                                    *)
(*    CHANGE CONTROL:                                                 *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                            *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                       *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                         *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE    *)
(*           NARRATION ON THE NEXT LINE.                              *)
(*        YY/MM/DD  CCXX  I. M. APERSON                               *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                        *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE OSTART *)
```

```
%PAGE
(* %INCLUDE PASAM. *)
(**)
  PROCEDURE PASAM(CONST KEYP:ENTKEY; VAR BLOCK:ENTBLOCK; VAR DATAREC:BLKDATA;
    VAR RC:EXT_RET_CODE; CONST NAME:ROUTINE);FORTRAN;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    AUTHOR:  D. KERCHNER       PDDI        CREATED:  84/09/11     *)
(*    VERSION:  MAS2                         REVISED:  YY/MM/DD     *)
(*                                                                  *)
(*    FUNCTION:                                                     *)
(*        THIS ROUTINE SERVES AS A LINK ROUTINE BETWEEN THE MAS     *)
(*        INTERFACE PACKAGE AND THE USER'S APPLICATION DEFINED      *)
(*        PROCEDURE MAKING IT FORTRAN CALLABLE                      *)
(*                                                                  *)
(*    ENVIRONMENT:                                                  *)
(*        IBM ASSEMBLER LANGUAGE                                    *)
(*        IBM 4341/3083 VAX 11/780 SYSTEMS                          *)
(*                                                                  *)
(*    EXECUTION PROCEDURE:                                          *)
(*        THIS ROUTINE IS INVOKED BY A CALL FROM A MAS INTERFACE    *)
(*        ROUTINE SUCH AS MALXEQ OR MAEXEQ, IN ORDER TO INVOKE A    *)
(*        USER DEFINED PROCEDURE WHICH IS IN THE USER'S MODULE      *)
(*                                                                  *)
(*    DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME    TYPE   I/O  DESCRIPTION                           *)
(*        KEYP            I   ENTITY KEY                            *)
(*        ENTBLOCK        I   APPLICATION DEFINED BLOCK             *)
(*        DATAREC        I/O  USER PASSED DATA (I/O)                *)
(*        RC              O   ERROR CONDITION RETURN CODE (PASSED   *)
(*                            ONLY, NOT PASSED                      *)
(*        ROUTINE         I   NAME OF THE USER DEFINED PROCEDURE    *)
(*                                                                  *)
(*    COMMONS:                                                      *)
(*                                                                  *)
(*    PROCESSING DESCRIPTION:                                       *)
(*        PASAM RECEIVES CALL FROM MAS PASCAL ROUTINE.  THIS        *)
(*        ASSEMBLER CSECT THEN BRANCHES TO THE USER DEFINED ROUTINE *)
(*        BY PASSING THE ADDRESS OF THAT ROUTINE IN A BRANCH REGISTER)
(*        INSTRUCTION.  WHEN THE USER ROUTINE COMPLETES PROCESSING, *)
(*        CONTROL IS RETURNED TO THE MAS PASCAL ROUTINE VIA THE     *)
(*        USER DEFINED ROUTINE.                                     *)
(*                                                                  *)
(*    COMMENTS:                                                     *)
(*        VARIABLES ARE NOT ACCESSED, BUT ARE PASSED THROUGH TO THE *)
(*        USER DEFINED ROUTINE.                                     *)
(*                                                                  *)
(*    DATA STRUCTURES/MAJOR VARIABLES:                              *)
(*                                                                  *)
```

```
(*    REGISTER USAGE:                                          *)
(*        R1  - PARAMETER LIST                                 *)
(*        R15 - BRANCHING REGISTER                             *)
(*                                                             *)
(*    CHANGE CONTROL:                                          *)
(*                                                             *)
(*END %INCLUDE PASAM                                          *)
```

```
%PAGE
(* %INCLUDE RDLSM. *)
(**)
  PROCEDURE RDLSM(VAR POSITION:LISTPSTN; CONST LISTREF:LISTPNTR;
      VAR KEYE:ENTKEY; VAR EOL:BOOLEAN; VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       READ A SYSTEM LIST AS A FIRST IN FIRST OUT ORDER.          *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          POSITION  - INDICATING NEXT ENTITY IN LISTREF TO BE READ.*)
(*          LISTREF   - LIST TO BE READ.                            *)
(*                                                                  *)
(*       OUTPUT                                                     *)
(*          POSITION  - UPDATED TO NEXT ENTITY.                     *)
(*          KEYE      - ENTITY READ FROM LIST.                      *)
(*          EOL       - TRUE IF ENTITY WAS READ ELSE FALSE.         *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*    METHOD                                                        *)
(*       IF THERE IS AN ENTITY AT INDICATED POSITION THEN PLACE     *)
(*           NEXT ENTITY INDICATED BY POSITION IN KEYE, ADJUST      *)
(*           POSITION TO INDICATE NEXT ENTITY, RETURN EOL SET TO    *)
(*           FALSE,                                                 *)
(*       ELSE                                                       *)
(*           RETURN EOL SET TO TRUE.                                *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE RDLSM. *)
```

```
%PAGE
(* %INCLUDE RDNM. *)
(**)
  PROCEDURE RDNM(CONST ENTRNG:ENTRANGE;VAR KEYL:LISTKEY;
      VAR KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;VAR EOLIST:BOOLEAN;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       READ THE NEXT ENTITY IN THE KIND RANGE FROM AN APPLICATION       *)
(*       LIST.                                                            *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       LIST PACKAGE.                                                    *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          ENTRNG     - ALL ENTITIES OUTSIDE THIS INCLUSIVE RANGE       *)
(*                       ARE IGNORED.                                     *)
(*          KEYL       - KEY OF LIST TO READ.                            *)
(*                                                                        *)
(*       OUTPUT                                                           *)
(*          KEYE       - KEY OF NEXT ENTITY ON LIST.                     *)
(*          ENTDEF     - DATA FOR NEXT ENTITY ON LIST.                   *)
(*          EOLIST     - TRUE IF NO ENTITY IN THE RANGE EXISTS.          *)
(*          RR         - THE FUNCTION RETURN RECORD.                     *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE RDNM. *)
```

```
%PAGE
(* %INCLUDE RDRLSM. *)
(**)
  PROCEDURE RDRLSM(CONST POSITION:LISTPSTN;CONST LISTREF:LISTPNTR;
      VAR KEYE:ENTKEY;VAR EOL:BOOLEAN;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       READ THE LAST ENTITY KEY FROM LISTREF.                      *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          POSITION  - RELATIVE POSITION IN LISTREF OF ENTITY       *)
(*                      TO BE READ.                                  *)
(*          LISTREF   - LIST WHOSE POSITION-TH ENTITY IS TO BE READ. *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          KEYE      - KEY OF POSITION-TH ENTITY IN LISTREF.       *)
(*          EOL       - TRUE IF NO POSITION-TH ENTITY IN LISTREF.   *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE RDRLSM. *)
```

```
1
 %PAGE
 (* %INCLUDE RDTLSM. *)
 (**)
   PROCEDURE RDTLSM(CONST LISTREF:LISTPNTR;VAR KEYE:ENTKEY;
       VAR EMPTY:BOOLEAN;VAR RR:RET_REC);EXTERNAL;
 (**)
 (*-------------------------------------------------------------------*)
 (*                                                                   *)
 (*    FUNCTION                                                       *)
 (*       READ THE LAST ENTITY KEY FROM LISTREF.                      *)
 (*                                                                   *)
 (*    LANGUAGE                                                       *)
 (*       PASCAL.                                                     *)
 (*                                                                   *)
 (*    PACKAGE                                                        *)
 (*       LIST PACKAGE.                                               *)
 (*                                                                   *)
 (*    ARGUMENTS                                                      *)
 (*       INPUT                                                       *)
 (*          LISTREF   - LIST WHOSE LAST ENTITY IS TO BE READ.        *)
 (*                                                                   *)
 (*       OUTPUT                                                      *)
 (*          KEYE      - RETURNS LAST ENTITY IN LISTREF.              *)
 (*          EMPTY     - TRUE IF NO ENTITIES IN LIST, ELSE FALSE.     *)
 (*          RR        - THE FUNCTION RETURN RECORD.                  *)
 (*                                                                   *)
 (*-------------------------------------------------------------------*)
 (**)
 (* END %INCLUDE RDTLSM. *)
```

```
%PAGE
(* %INCLUDE REVAADB. *)
(**)
  PROCEDURE REVAADB(CONST ENTBPNTR:ENTPNTR;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       ASSIGN THE VALUE OF A SYSTEM UDB TO AN APPLICATION ENTBLOCK. *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       UDB PACKAGE.                                                 *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          ENTBPNTR  - POINTER TO ENTBLOCK CREATED.                  *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          ENTDEF    - THE ENTBLOCK WITH THE VALUE OF SYSUDB         *)
(*                      ASSIGNED TO IT.                               *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       REVAADB USES SYSTEM ROUTINE AMPXMOVE TO MOVE DATA IN         *)
(*       MEMORY.  THE NUMBER OF BYTES TO MOVE MUST BE SPECIFIED.      *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE REVAADB. *)
```

```
%PAGE
(* %INCLUDE REVNODM *)
(**)
  PROCEDURE REVNODM(VAR KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  UNKNOWN                  CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 2                        REVISED: 84/10/11 CC   *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       REVISE AN ENTITY'S USER DATA BLOCK.                         *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                             *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME     I/O  DESCRIPTION                                  *)
(*        KEYE      I   KEY OF ENTITY TO BE REVISED.                 *)
(*        ENTDEF    I   NEW DATA FOR ENTITY TO BE REVISED.           *)
(*        RR        O   ERROR CONDITION RETURN CODE.                 *)
(*                      = 0  NORMAL RETURN CODE.                     *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                        *)
(*           UPDATED DOCUMENTATION.                                  *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED DECLARATION ON KEYE AND ENTDEF TO VAR.          *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE REVNODM *)
```

```
%PAGE
(* %INCLUDE REVRLSM. *)
(**)
  PROCEDURE REVRLSM(CONST POSITION:LISTPSTN;CONST KEYE:ENTKEY;
      CONST LISTREF:LISTPNTR;VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*    FUNCTION                                                      *)
(*       CHANGE AN ENTITY IN A SYSTEM LIST.                         *)
(*                                                                  *)
(*    LANGUAGE                                                      *)
(*       PASCAL.                                                    *)
(*                                                                  *)
(*    PACKAGE                                                       *)
(*       LIST PACKAGE.                                              *)
(*                                                                  *)
(*    ARGUMENTS                                                     *)
(*       INPUT                                                      *)
(*          POSITION  - THE RELATIVE POSITION OF THE ENTITY IN      *)
(*                      THE LIST.                                   *)
(*          KEYE      - THE NEW ENTITY KEY.                         *)
(*          LISTREF   - A POINTER TO A SYSTEM LIST.                 *)
(*                                                                  *)
(*       OUTPUT                                                     *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(**)
(* END %INCLUDE REVRLSM. *)
```

```
%PAGE
(* %INCLUDE REVSADB *)
(**)
  PROCEDURE REVSADB(VAR ENTDEF:ENTBLOCK;VAR ENTBPNTR:ENTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  UNKNOWN                CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 2                      REVISED: 84/10/11 CC   *)
(*                                            REVISED: 84/12/10      *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       REPLACE THE VALUE OF A SYSTEM ENTBLOCK WITH THE VALUE OF    *)
(*       ENTDEF.                                                     *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                             *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME     I/O  DESCRIPTION                                  *)
(*        ENTDEF    I   THE APPLICATION ENTBLOCK VALUE TO ASSIGN     *)
(*                      TO A SYSTEM ENTBLOCK.                        *)
(*        ENTBPNTR  O   POINTER TO THE SYSTEM ENTBLOCK TO BE REVISED.*)
(*        RR        O   ERROR CONDITION RETURN CODE.                 *)
(*                      = 0  NORMAL RETURN CODE.                     *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*       REVSADB USES SYSTEM ROUTINE AMPXMOVE TO MOVE DATA IN        *)
(*       MEMORY.   THE NUMBER OF BYTES TO MOVE MUST BE SPECIFIED.    *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                        *)
(*           UPDATED DOCUMENTATION.                                  *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED ENTDEF FROM CONST TO VAR.                       *)
(*        84/12/10  MAS VER 2  J. JOHNSON                            *)
(*           TO CALL MASDSP.                                         *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE REVSADB *)
```

```
%PAGE
(* %INCLUDE RLSNM. *)
(**)
  PROCEDURE RLSNM(VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       RELEASE ALL THE LISTS ON THE CURRENT LIST OF LISTS.          *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          NONE      -                                               *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE RLSNM. *)
```

```
%PAGE
(* %INCLUDE RSTLSM. *)
(**)
  PROCEDURE RSTLSM(VAR POSITION:LISTPSTN;CONST LISTREF:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       RESETS POSITION TO INDICATE THE BEGINNING OF A LIST.        *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL.                                                     *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       LIST PACKAGE.                                               *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          LISTREF   - POINTER TO A LIST.                           *)
(*                                                                   *)
(*       OUTPUT                                                      *)
(*          POSITION  - RESET TO INDICATE BEGINNING OF LIST.         *)
(*          RR        - THE FUNCTION RETURN RECORD.                  *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE RSTLSM. *)
```

```
%PAGE
(* %INCLUDE RSTSFLG *)
(**)
  PROCEDURE RSTSFLG(CONST LISTP:LISTPNTR;
      CONST SETTING:BOOLEAN;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       RESET THE REQUESTED POSITION IN THE INTERNAL MAS PROCESS    *)
(*       FLAG (MAPROB) IN THE IIT TO THE REQUESTED BOOLEAN VALUE.    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME       I/O     DESCRIPTION                              *)
(*       ====       ===     ===========                              *)
(*       LISTP       I    THE LIST OF ENTITIES THAT ARE TO HAVE A BYTE *)
(*                        IN THE SYSUSE FLAG SET.                    *)
(*       SETTING     I    BOOLEAN VALUE THE SYSUSE(FLG_POS) BYTE IS TO *)
(*                        BE SET TO.  (IE: TRUE OR FALSE)            *)
(*       RR          O    FUNCTION RETURN CODE                       *)
(*                        = 0  GOOD RETURN                           *)
(*                        > 0  CRITICAL ERROR                        *)
(*                        < 0  WARNING                               *)
(*   $COMMONS                                                        *)
(*       NONE                                                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*       LANGUAGE: IBM PASCAL                                        *)
(*       HARDWARE SYSTEM: IBM 360,370,43XX                           *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*       INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE            *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       FOR EACH ENTITY ON THE LIST OF ENTITIES, THE MAPROB         *)
(*       BYTE IS SET TO THE INPUT SETTING.                           *)
(*                                                                   *)
(*   $COMMENTS                                                       *)
(*       USES THE MAPROB FLAG IN THE T_ENTITY.                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*       REVISED:  04/26/85     E.D. SHREVE             W315         *)
(*                 TO SET THE MAPROB BYTE IN THE T_ENTITY INSTEAD    *)
(*                 OF THE SYSUSE OF THE ADB. FOR INTERNAL MAS.       *)
(*                                                                   *)
(*       ORIGINATED: 07/10/84     C. J. SAMPLE           W315        *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE RSTSFLG *)
```

```
%PAGE
(* %INCLUDE RVRLSM.*)
(**)
  PROCEDURE RVRLSM(VAR KEYIN:LISTPNTR; VAR KEYOUT:LISTPNTR;
                   VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        CREATE AN OUTPUT LIST THAT CONTAINS THE ENTITIES ON THE    *)
(*        INPUT LIST IN REVERSE ORDER.                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        KEYIN          I   LIST TO COPY FROM                     *)
(*        KEYOUT         O   NEW LSIT WITH ENTITY'S REVERSED       *)
(*        RR             O   EXTERNAL RETURN CODE-                 *)
(*                           = 0  OK RETURN CODE                   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL MODEL ACCESS SOFTWARE ROUTINE                     *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        IF THE INPUT LIST IS NOT EMPTY, A NEW OUTPUT LIST IS       *)
(*        CREATED. THEN THE ENTITIES ARE MOVED INTO THE NEW LIST     *)
(*        IN REVERSE ORDER.                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 04/11/86 MAS2  E. D. SHREVE              W315     *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*END----------------------------------------------------------------*)
(* END %INCLUDE RVRLSM. *)
```

```
%PAGE
(* %INCLUDE SETRULS. *)
(**)
  PROCEDURE SETRULS(CONST USER:ENTKEY; CONST CNST:ENTKEY; CONST
    DEL_LIST:LISTPNTR; VAR RULE:T_RULE; VAR MIN_CNST:LISTPSTN;
    VAR RR:RET_REC);EXTERNAL;
%PAGE
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*       SET DELETE FLAGS ACCORDING TO USER'S DEPENDENCE & STRENGTH   *)
(*       RULES.                                                       *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        USER           I   USER WHOSE RULES ARE TO BE FOUND BASED ON*)
(*                           THE RELATIONSHIP WITH CNST               *)
(*        CNST           I   CNST WHOSE RULES ARE TO BE FOUND BASED ON*)
(*                           THE RELATIONSHIP WITH USER               *)
(*        DEL_LST        I   LIST OF KEYS THAT ARE ELIGIBLE FOR       *)
(*                           DELETION                                 *)
(*        RULE           O   INDICATES WHICH DELETE AND COMPRESS RULES*)
(*                           ARE VALID FOR THIS RELATIONSHIP          *)
(*        MIN_CNST       O   MINIMUM NUMBER OF CONSTITIUENTS FOR USER *)
(*        RR             O   THE FUNCTION RETURN CODE.                *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*       IF THE USER IS IN THE DEL_LST THEN EXIT                      *)
(*       ELSE                                                         *)
(*        THE RULES OF THE CONNECTION ARE FOUND AND THE RULE SET IS   *)
(*        FILLED APPROPRIATELY                                        *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: 06/17/86          B. A. ULMER                  FRMI   *)
(*     ADD NEW PARAMETERS TO SETRULS AND CHANGE PROCESSING TO HANDLE  *)
(*     THE NEW DELETE RULES - MAJOR REWRITE                           *)
(*                                                                    *)
```

```
(*     REVISED: 09/  /85          B. A. ULMER                    FRMI   *)
(*      ADD NEW PARAMETERS TO FNDURUL TO HANDLE TWO NEW DELETE RULES   *)
(*                                                                     *)
```

```
%PAGE
(* %INCLUDE SETSWCI. *)
(**)
  PROCEDURE SETSWCI(CONST LISTIN:LISTPNTR;CONST ISWT:BOOLEAN;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    FUNCTION                                                       *)
(*       SET A SWITCH IN THE USER DATA BLOCK FOR EACH ENTITY         *)
(*       AND ALL CONSTITUENTS INCLUSIVE.                             *)
(*                                                                   *)
(*    LANGUAGE                                                       *)
(*       PASCAL                                                      *)
(*                                                                   *)
(*    PACKAGE                                                        *)
(*       ENTITY  PACKAGE.                                            *)
(*                                                                   *)
(*    ARGUMENTS                                                      *)
(*       INPUT                                                       *)
(*          LISTIN    - THE LIST CONTAINING THE ENTITIES FOR WHICH   *)
(*                      SWITCH WILL BE SET.                          *)
(*          ISWT      - THE BOOLEAN SETTING FOR THE SWITCH.          *)
(*                      VARIABLE DISCRIPTION AS BEFORE.              *)
(*       OUTPUT                                                      *)
(*          RR        - THE FUNCTION RETURN CODE.                    *)
(*                                                                   *)
(*    METHOD                                                         *)
(*       THIS ROUTINE IS CALLED RECURSIVELY TO PROCESS THE ENTITIES  *)
(*       THAT ARE CONSTITUENTS OF THE ENTITIES ON THE INPUT LIST.    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE SETSWCI. *)
```

```
%PAGE
(* %INCLUDE SORTDLST. *)
(**)
  PROCEDURE SORTDLST(CONST DEL_LST:LISTKEY;VAR SRT_LST:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       GIVEN AN APPLICATION LIST OF ENTITIES TO BE DELETED,        *)
(*       DEL_LST RETURNS A SYSTEM LIST SORTED IN USER-CONSTITUENT    *)
(*       ORDER IN SRT_LST.                                           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        DEL_LST        I   APPLICATION LIST CONTAINING THE LISTKEY *)
(*                           OF TH E ENTITIES TO BE DELETED         *)
(*        SRT_LST        O   POINTER TO A SYSTEM LIST CONTAINING THE *)
(*                           ENTITIES OF THE DEL_LST SORTED IN USER- *)
(*                           CONSTITUENT ORDER                       *)
(*        RC             O   EXTERNAL RETURN CODE                   *)
(*                           = 0  OK                                 *)
(*                           > 0  CRITICAL ERROR                     *)
(*                           < 0  WARNING                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*       SET MAPROB FLAG ON FOR ALL ENTITIES IN THE DEL-LST          *)
(*           THAT ARE NOT 'MARKED FOR DELETE'                        *)
(*       REPEAT FOR EACH ENTITY IN DEL_LST                          *)
(*           IF NOT PROCESSED (MAPROB = FALSE)                      *)
(*                CALL SRTBYUSR TO PUT ALL USER ENTITIES ON THE     *)
(*                SRT_LST BEFORE ADDING THE ENTITY.                  *)
(*       RESET MAPROB AND MAPROB2                                    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*       USES NDS PROCEDURES RSTLSM, RDLSM, NEWLSM AND SRTBYUSR      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 12/17/85          B. A. ULMER                 FRMI    *)
(*     CHANGE SRTBYUSR TO SRTBYCNT - SORT LIST NOW IN CONSTITUENT TO *)
(*     USER ORDER                                                    *)
```

```
(*                                                                    *)
(*    REVISED: 12/03/85          E. D. SHREVE                 FRMI    *)
(*     REWRITTEN TO REPLACE THE COMPARE SORT WITH A SUBROUTINE THAT   *)
(*     USES THE SYSTEM FLAGS (MAPROB AND MAPROB2) FOR SORTING         *)
(*                                                                    *)
(*    REVISED: 07/01/85          B. A. ULMER                  FRMI    *)
(*     ELIMINATE THE LEAVE FUNCTION TO IMPROVE COMPATABILITY WITH VAX *)
(*                                                                    *)
(*    REVISED: 04/10/85          B. A. ULMER                  FRMI    *)
(*     DO NOT PROCESS THE ALREADY "MARKED FOR DELETE" ENTITIES        *)
(*                                                                    *)
(*    ORIGINATED: 06/19/84       R. A. MCCLUSKEY              FRMI    *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                 *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(**)
(* END %INCLUDE SORTDLST. *)
(**)
```

```
%PAGE
(* %INCLUDE SORTLSM. *)
(**)
  PROCEDURE SORTLSM(VAR LISTREF:LISTPNTR; CONST PROCNAME:ROUTINE;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       SORT A SYSTEM LIST.                                          *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       LIST PACKAGE.                                                *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          LISTREF   - LIST TO BE SORTED.                            *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*    METHOD                                                          *)
(*       THE SYSTEM LIST LISTREF IS SORTED IN APPLICATION DEFINED     *)
(*       ORDER. THE ORDER IS DETERMINED BY A USER DEFINED FUNCTION;   *)
(*       ORDER. ORDER RETURNS FALSE IF TWO ENTITIES ARE IN ORDER ELSE*)
(*       IT RETURNS TRUE. IF LISTREF HAS LESS THAN TWELVE ENTITIES,   *)
(*       THE BUBBLE SORT ALGORITHM IS USED. IF LISTREF CONTAINS MORE  *)
(*       THAN ELEVEN ENTITIES A SLIGHT VARIATION OF QUICK SORT IS     *)
(*       USED WHEN THE SUBLISTS CREATED BY STANDARD QUICK SORT        *)
(*       CONTAIN LESS THAN TWELVE ENTITIES, SORTLSM REVERTS BACK TO   *)
(*       BUBBLE SORT.  IN GENERAL SORTLSM IS FASTER THAN EITHER       *)
(*       BUBBLE SORT OR QUICK SORT.                                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE SORTLSM. *)
```

```
%PAGE
(* %INCLUDE SRTBYCNT. *)
(**)
  PROCEDURE SRTBYCNT(VAR KEY1:ENTKEY;VAR SRT_LST:LISTPNTR;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*       THIS IS A RECURSIVE ROUTINE THAT PLACES THE CNST ENTITIES    *)
(*       OF KEY1, THAT ARE ON THE DELETE LIST, INTO THE SRT_LST       *)
(*       BEFORE KEY1 IS ADDED.                                        *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        KEY1           I   THE ENTITY THAT WILL BE PLACED ON THE   *)
(*                           OUTPUT LIST ALONG WITH ITS CONSTITUENTS  *)
(*        SRT_LST        O   POINTER TO A SYSTEM LIST CONTAINING THE  *)
(*                           ENTITIES OF THE DEL_LST SORTED IN        *)
(*                           CONSTITUENT-USER ORDER                   *)
(*        RC             O   EXTERNAL RETURN CODE                    *)
(*                           = 0  OK                                  *)
(*                           > 0  CRITICAL ERROR                      *)
(*                           < 0  WARNING                             *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*      (ALL ENTITIES THAT WERE ON THE ORIGINAL DELETE LIST HAVE BEEN *)
(*          FLAGGED IN THE ENTITY BLOCK (MAPROB = TRUE))              *)
(*       EACH CNST OF KEY1 IS PROCESSED:                              *)
(*           IF NOT PROCESSED (MAPROB2 = FALSE) AND IN THE DELETE     *)
(*              LIST (MAPROB = TRUE) THEN                             *)
(*                CALL SRTBYCNT TO PUT THE CNST ENTITY ON SRT_LST     *)
(*       ADD KEY1 TO THE SRT_LST.                                     *)
(*       SET KEY1 (MAPROB2 = TRUE)                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*       USES NDS PROCEDURES RSTLSM, RDLSM,  AND SRTBYCNT             *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(*    CREATED: 12/17/85          B  A. ULMER                    FRMI     *)
(*       THIS ROUTINE IS USED BY SORTDLST FOR SORTING.  IT REPLACES   *)
(*       THE COMPARE SORT IN THE OLD SORTDLST WHICH WAS INEFFICIENT.  *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
%PAGE                                                                  *)
(*--------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                *)
(*       THESE ARE DESCRIBED IN THE NDSDCL INCLUDE MEMBER.            *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(**)
(* END %INCLUDE SRTBYCNT. *)
(**)
```

```
%PAGE
(* %INCLUDE UPDCRBE *)
(**)
  PROCEDURE UPDCRBE(CONST CRB:CRBPNTR; CONST EKEY:ENTKEY;
      VAR POS:LISTPSTN; VAR DIR:LISTDIR; VAR RR:RET_REC);EXTERNAL;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR:  B. A. ULMER            FRMI    CREATED: 85/02/08  CC??*)
(*    VERSION: XXXX                           REVISED: YY/MM/DD  CC  *)
(*                                                                 *)
(*    FUNCTION:                                                    *)
(*       UPDATE AN ENTRY IN THE CRB                                *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*       IBM PASCAL LANGUAGE                                       *)
(*       IBM 30XX, 43XX DEPENDENT CODE, OR OTHER APPROPRIATE H/W.  *)
(*                                                                 *)
(*    EXECUTION PROCEDURE:                                         *)
(*       HOW IS THIS ROUTINE/MODULE TO BE EXECUTED.                *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*       NAME    I/O  DESCRIPTION                                  *)
(*       CRB     I/O  CONSTITUENT READ BLOCK ADDRESS               *)
(*       EKEY     I   ENTITY KEY OF ENTRY TO UPDATE                *)
(*       POS      I   NEW LIST POSITION SETTING                    *)
(*       DIR      I   NEW DIRECTION OF LIST (FORWARD OR REVERSE)   *)
(*       RR       O   ERROR CONDITION RETURN CODE                  *)
(*                    = 0  OK RETURN CODE                          *)
(*                    = 1  YOU BLEW IT                             *)
(*                    = 2  THE ROUTINE BLEW IT                     *)
(*                                                                 *)
(*    COMMONS:                                                     *)
(*      COM1                                                       *)
(*        VAR1      I   VAR1 NAME MUST BE FILLED, CHARACTER DATA   *)
(*                      MUST BE PROVIDED                           *)
(*        VAR2      I   VAR2 MUST BE SPECIFIED                     *)
(*      COM2                                                       *)
(*        VAR3      I   CHARACTER DATA MUST BE SPECIFIED           *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*        DETAILED DESCRIPTION OF HOW THIS ROUTINE WORKS, WHICH    *)
(*        FILES NEED TO BE OPENED/CLOSED, FILES USED, ETC.         *)
(*                                                                 *)
(*    COMMENTS:                                                    *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND*)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                  *)
(*                                                                 *)
```

```
(*    CHANGE CONTROL:                                                  *)
(*        YY/MM/DD  CCZZ  I. M. THECHANGER                             *)
(*           DESCRIPTION OF LATEST CHANGE MADE.                        *)
(*        YY/MM/DD  CCYY  I. M. THEPROGRAMMER                          *)
(*           DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE     *)
(*           NARRATION ON THE NEXT LINE.                               *)
(*        YY/MM/DD  CCXX  I. M. APERSON                                *)
(*           DESCRIPTION OF FIRST CHANGE MADE.                         *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(**)
(* END %INCLUDE UPDCRBE *)
```

```
%PAGE
(* %INCLUDE VERAPN. *)
(**)
  PROCEDURE VERAPN(CONST KEY1:ANYKEY;CONST KEY2:ANYKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       VERIFY LEGALITY OF APPENDING AN ENTITY OR LIST OF ENTITIES   *)
(*       (KEY2) TO AN ENTITY OR LIST OF ENTITIES (KEY1).              *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       VERIFY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEY1       - KEY OF APPLICATION LIST TO WHICH ENTITIES   *)
(*                       ARE TO BE APPENDED.  IF ENTITY KEY, THEN     *)
(*                       ADD TO CONSTITUENT LIST.                     *)
(*          KEY2         KEY OF APPLICATION LIST OF ENTITIES TO      *)
(*                       APPEND.  IF ENTITY KEY, THEN ADD ENTITY      *)
(*                       TO LIST.                                     *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR         - THE FUNCTION RETURN RECORD.                 *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE VERAPN. *)
```

```
%PAGE
(* %INCLUDE VERCN. *)
(**)
  PROCEDURE VERCN(CONST KEYLU:LISTKEY;CONST KEYLC:LISTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       VERIFY LEGALITY OF CONNECTING EACH ENTITY ON A LIST OF       *)
(*       USERS TO EACH ENTITY ON A LIST OF CONSTITUENTS.              *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       VERIFY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYLU     - KEY OF LIST OF USERS.                       *)
(*          KEYLC     - KEY OF LIST OF CONSTITUENTS.                *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                 *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE VERCN. *)
```

```
%PAGE
(* %INCLUDE VERCR *)
(**)
  PROCEDURE VERCR(VAR ENTDEF:ENTBLOCK;CONST KEYE:ANYKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-----------------------------------------------------------------*)
(*                                                                 *)
(*    AUTHOR:  UNKNOWN                CADD    CREATED: YY/MM/DD CC *)
(*    VERSION: MAS VER 2                      REVISED: 84/10/11 CC *)
(*                                                                 *)
(*    FUNCTION:                                                    *)
(*       VERIFY LEGALITY OF CREATING AN ENTITY WITH THE USER       *)
(*       SUPPLIED ENTITY DATA BLOCK AND LIST OF CONSTITUENTS.      *)
(*                                                                 *)
(*    ENVIRONMENT:                                                 *)
(*        IBM PASCAL LANGUAGE                                      *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                           *)
(*                                                                 *)
(*    DESCRIPTION OF ARGUMENTS:                                    *)
(*        NAME    I/O  DESCRIPTION                                 *)
(*        ENTDEF   I   USER SUPPLIED DATA FOR ENTITY BLOCK.        *)
(*        KEYE     I   KEY OF ENTITY OR APPLICATIONS LIST OF       *)
(*                     ENTITIES TO BE CONSTITUENTS OF THIS ENTITY. *)
(*        RR       O   ERROR CONDITION RETURN CODE.                *)
(*                     = 0  NORMAL RETURN CODE.                    *)
(*                                                                 *)
(*    COMMONS:                                                     *)
(*                                                                 *)
(*    PROCESSING DESCRIPTION:                                      *)
(*                                                                 *)
(*    COMMENTS:                                                    *)
(*                                                                 *)
(*    CHANGE CONTROL:                                              *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                      *)
(*           UPDATED DOCUMENTATION.                                *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                        *)
(*           CHANGED ENTDEF FROM CONST TO VAR.                     *)
(*                                                                 *)
(*-----------------------------------------------------------------*)
(**)
(* END %INCLUDE VERCR *)
```

```
%PAGE
(* %INCLUDE VERDEL. *)
(**)
  PROCEDURE VERDEL(CONST KEYE:ANYKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       VERIFY LEGALITY OF DELETING AN ENTITY.                       *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       VERIFY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE      - KEY OF ENTITY TO BE DELETED FROM NETWORK.     *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE VERDEL. *)
```

```
%PAGE
(* %INCLUDE VERGT. *)
(**)
  PROCEDURE VERGT(CONST KEYE:ENTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*    FUNCTION                                                        *)
(*       VERIFY LEGALITY OF RETRIEVING AN ENTITY WITH THE USER        *)
(*       SUPPLIED ENTITY KEY.                                         *)
(*                                                                    *)
(*    LANGUAGE                                                        *)
(*       PASCAL.                                                      *)
(*                                                                    *)
(*    PACKAGE                                                         *)
(*       VERIFY PACKAGE.                                              *)
(*                                                                    *)
(*    ARGUMENTS                                                       *)
(*       INPUT                                                        *)
(*          KEYE      - KEY OF ENTITY TO BE RETRIEVED FROM NETWORK.   *)
(*                                                                    *)
(*       OUTPUT                                                       *)
(*          RR        - THE FUNCTION RETURN RECORD.                   *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(**)
(* END %INCLUDE VERGT. *)
```

```
%PAGE
(* %INCLUDE VERUD *)
(* (VERFORM) VERIFY ROUTINE FORMALS. *)
(**)
  PROCEDURE VERUD(CONST KEYE:ENTKEY;VAR ENTDEF:ENTBLOCK;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    AUTHOR:  UNKNOWN                CADD    CREATED: YY/MM/DD CC   *)
(*    VERSION: MAS VER 2                      REVISED: 84/10/11 CC   *)
(*                                                                   *)
(*    FUNCTION:                                                      *)
(*       VERIFY LEGALITY OF UPDATING AN ENTITY WITH THE USER         *)
(*       SUPPLIED ENTITY KEY USING THE USER SUPPLIED ENTITY DATA     *)
(*       BLOCK AND LIST OF CONSTITUENTS.                             *)
(*                                                                   *)
(*    ENVIRONMENT:                                                   *)
(*        IBM PASCAL LANGUAGE                                        *)
(*        IBM 30XX, 43XX, DEC VAX 11/780                             *)
(*                                                                   *)
(*    DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME    I/O  DESCRIPTION                                   *)
(*        KEYE     I   KEY OF EXISTING ENTITY.                       *)
(*        ENTDEF   I   USER SUPPLIED DATA FOR NEW ENTITY BLOCK.      *)
(*        KEYL     I   KEY OF LIST OF CONSTITUENTS TO BE CONNECTED   *)
(*                     TO THIS ENTITY.                               *)
(*        RR       O   ERROR CONDITION RETURN CODE.                  *)
(*                     = 0  NORMAL RETURN CODE.                      *)
(*                                                                   *)
(*    COMMONS:                                                       *)
(*                                                                   *)
(*    PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    COMMENTS:                                                      *)
(*                                                                   *)
(*    CHANGE CONTROL:                                                *)
(*        84/10/11  MAS VER 2  D. J. KERCHNER                        *)
(*           UPDATED DOCUMENTATION.                                  *)
(*        84/10/04  MAS VER 2  E. D. SHREVE                          *)
(*           CHANGED ENTDEF FROM CONST TO VAR.                       *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(**)
(* END %INCLUDE VERUD *)
```

```
%PAGE
(* %INCLUDE XIEMM. *)
(**)
  PROCEDURE XIEMM(VAR KEYE:ENTKEY;VAR RR:RET_REC);EXTERNAL;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        TO DELETE AN ENTITY.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        KEYE          I/O  KEY OF ENTITY TO BE DELETED, WILL BE    *)
(*                           SET TO NIL.                             *)
(*                                                                   *)
(*          RR           O   THE FUNCTION RETURN CODE.               *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: 09/13/85          L. J. BEHAN                  FRMI  *)
(*     CHANGED TO ENSURE THE DECREMENTING OF THE READ POSITION OF A  *)
(*     USER ENTITY CONSTITUENT LIST                                  *)
(*                                                                   *)
(*    REVISED: 02/18/85          B. A. ULMER                  FRMI  *)
(*     CHANGED THE STRUCTURE OF THE INTERNAL ITEM FOR THE            *)
(*     IMPLEMENTATION OF THE CRB                                     *)
(*                                                                   *)
(*    REVISED: 10/05/84          E. D. SHREVE                 FRMI  *)
(*     CHANGED THE KEYE PARMS FOR XULST AND XCLST TO VAR             *)
(*                                                                   *)
```

```
%PAGE
(* %INCLUDE XREMM. *)
(**)
  PROCEDURE XREMM(CONST KEYEU:ENTKEY;CONST KEYEC:ENTKEY;
      VAR RR:RET_REC);EXTERNAL;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*       DELETE THE FIRST RELATION BETWEEN THE USER AND CONSTITUENT.      *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL.                                                          *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       ENTITY PACKAGE.                                                  *)
(*                                                                        *)
(*    ARGUMENTS                                                           *)
(*       INPUT                                                            *)
(*          KEYEU     - THE INTERNAL ITEM ON THE USER SIDE OF THE        *)
(*                      RELATION.                                         *)
(*          KEYEC     - THE INTERNAL ITEM ON THE CONSTITUENT SIDE        *)
(*                      OF THE RELATION.                                  *)
(*                                                                        *)
(*       OUTPUT                                                           *)
(*          RR        - THE FUNCTION RETURN RECORD.                      *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)
(* END %INCLUDE XREMM. *)
```

APPENDIX F

ACCESS SOFTWARE DATA DICTIONARY

The following members of the Access Software constitute a data dictionary. These listings are provided on the following pages.

APLTYP	- Application type declarations for MAS interface routines	F-2
ENTBLK	- Definition of an application data block for an entity	F-3
MASKIND	Constants for internal routines	F-4
MASTYP	- Constants and types for MAS interface routines	F-6
NDSDCL	- Constants and types for MAS internal routines	F-10
$PCMGT	- MAS memory management type declaration	F-21
PCMGT	MAS memoty management type declaration	F-22
SCHDCL	- Constants and types for MAS internal routines	F-23
SCHTYP	- Constants and types for MAS internal routines	F-24
TVERIFY	Verifies common types	F-28

```
(* %INCLUDE APLTYP *)
(*-------------------------------------------------------------------*)
(*    APPLICATION TYPE DECLARATIONS FOR USING THE MAS PACKAGE.       *)
(*-------------------------------------------------------------------*)
TYPE
  ANYKEY = INTEGER;
  LISTKEY = ANYKEY;
  ENTKEY = ANYKEY;
  ORD_KIND = INTEGER;
  EXT_RET_CODE = INTEGER;
  LISTPSTN = INTEGER;
  LISTINDX = INTEGER;
  LISTSIZE = INTEGER;
  ROUTINE = ARRAY(.1..8.) OF CHAR;
  NAMTYP = PACKED ARRAY(.1..6.) OF CHAR;
(* END %INCLUDE APLTYP *)
```

```
%PAGE
(*--------------------------------------------------------------------*)
(* (ENTBLK) DEFINITION OF ENTITY BLOCK TYPE.                          *)
(*--------------------------------------------------------------------*)
%INCLUDE PRINTOFF
(**)
(*--------------------------------------------------------------------*)
(*                          MAS VERSION 1                             *)
(*                    PACKAGE:  ENTITY PACKAGE.                       *)
(*--------------------------------------------------------------------*)
(**)
(* THE ENTITY BLOCK IS DEFINED SEPARATELY TO ALLOW THE UNLIKELY       *)
(* APPLICATION STRUCTURE OF AN ENTDATA RECORD CONTAINING FIELDS OF    *)
(* TYPES DEFINED IN NDSDCL.  IN SUCH A CASE THE APPLICATION SHOULD    *)
(* INCLUDE NDSDCL, THEN DEFINE ENTDATA, THEN INCLUDE ENTBLOCK.        *)
(**)
  CONST
    SYS_MRDFLG = 1;      (*    MARK FOR DELETE FLAG            BAU  *)
    SYS_PRCFLG = 2;      (*    PROCESS FLAG                         *)
    SYS_ABSFLG = 3;      (*    ABSENT/PRESENT FLAG                  *)
    SYS_APLFLG = 4;      (*    APPLICATION FLAG                     *)
  TYPE
    SYSINDX=1..4;
    ENTBLOCK=RECORD
      KIND:ORD_KIND;
      SIZE:ENTSIZE;
      SYSUSE:ARRAY(.SYSINDX.) OF BOOLEAN;
      DATA:ENTDATA;
    END;
%PRINT ON
(* END %INCLUDE ENTBLK *) 1
```

```
(* %INCLUDE MASKIND. *)
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    FUNCTION                                                            *)
(*        CONSTANT DECLARATIONS FOR MAS ENTITY TYPES                      *)
(*                                                                        *)
(*    LANGUAGE                                                            *)
(*       PASCAL                                                           *)
(*                                                                        *)
(*    PACKAGE                                                             *)
(*       CADD EMULATION PACKAGE                                           *)
(*                                                                        *)
(*    ARGUMENTS - NONE.                                                   *)
(*                                                                        *)
(*    COMMENTS                                                            *)
(*        1001 - 1480 ARE CADD ENTITY TYPES                               *)
(*        1501 - 1508 ARE PDDI ENTITY TYPES (TOPOGRAPHICAL)               *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(**)

CONST
    HEADER                = 1000;
    MAS_POINT             = 1001;
    MAS_LINE              = 1002;
    MAS_PLANE             = 1003;
    MAS_ARC               = 1004;
    MAS_CIRCLE            = 1014;
    MAS_PC_CURVE          = 1005;
    MAS_SPHERE            = 1006;
    MAS_CROSSHAIR         = 1007;
    MAS_CONIC             = 1010;
    MAS_ELLIPSE           = 1020;

    MAS_LOFT_MAT          = 1051;
    MAS_LOFT_FUN          = 1052;
    MAS_LOFT_SPC          = 1053;
    MAS_LOFT_LPA          = 1054;
    MAS_LOFT_FAC          = 1055;
    MAS_LOFT_COF          = 1056;

    MAS_PCPATCH           = 1086;
    MAS_BOUNDED_PLANE     = 1087;
    MAS_SURFACE           = 1088;
    MAS_GROUP             = 1089;
    MAS_TEXT              = 1090;
    MAS_COORDINATE_DIM    = 1180;
    MAS_CONVENTIONAL_DIM  = 1280;
    MAS_ANGULAR_DIM       = 1380;
    MAS_RADIUS_DIM        = 1480;
```

```
    MAS_VERTEX              = 1501;
    MAS_EDGE                = 1502;
    MAS_LOOP                = 1504;
    MAS_FACE                = 1505;
    MAS_SHELL               = 1507;
    MAS_OBJECT              = 1508;

    DELETE_OP               = 2010;
    REPLACE_OP              = 2020;
    CONNECT_OP              = 2030;
    DISCONNECT_OP           = 2040;
    REPLACE_ATTRIBUTE_OP    = 2050;

    PICK_ENTITY             = 2060;

(* END %INCLUDE MASKIND. *)
```

```
%PAGE
(*-------------------------------------------------------------------*)
(*  (MASTYP) CONSTANTS AND TYPES USED IN THE MAS INTERFACE.          *)
(*-------------------------------------------------------------------*)
%INCLUDE PRINTOFF
(*-------------------------------------------------------------------*)
(*                        MAS VERSION 1                              *)
(*                   PACKAGE:  NETWORK PACKAGE.                      *)
(*-------------------------------------------------------------------*)
(**)
CONST
(**)
(*-------------------------------------------------------------------*)
(* THE NUMBER OF ROUTINES IN MAS/SCHEMA/NDS.                        *)
(*-------------------------------------------------------------------*)
(**)
    NUM_MAS_ROUTINES=71;
(**)
(*-------------------------------------------------------------------*)
(* EACH INTERFACE ROUTINE MUST BE ASSOCIATED WITH A NUMBER FOR      *)
(* STATISTICS GENERATION.                                           *)
(*-------------------------------------------------------------------*)
(**)
    MAEUD_ID=1;
    MAEGTK_ID=2;
    MAECR_ID=3;
    MADMP_ID=4;
    MAINIT_ID=5;
    MDMP_ID=6;
    MAL_ID=7;
    MALATC_ID=8;
    MALNO_ID=9;
    MALGTK_ID=10;
    MAEXEQ_ID=11;
    MALXEQ_ID=12;
    MALK_ID=13;
    MALFND_ID=14;
    MAEU_ID=15;
    MALD_ID=16;
    MALRDE_ID=17;
    MALCPY_ID=18;
    MAEC_ID=19;
    MAECI_ID=20;
    MAEUI_ID=21;
    MAESWT_ID=22;
    MAESVL_ID=23;
    MALKL_ID=24;
    MAED_ID=25;
    MAEDT_ID=26;
    MALRMV_ID=27;
    MALOR_ID=28;
```

```
        MALRPL_ID=29;
        MALAND_ID=30;
        MALNOT_ID=31;
        MAEA_ID=32;
        MAEAI_ID=33;
        MALSTF_ID = 34;
        MALSTR_ID = 35;
        MALRD_ID  = 36;
        MALDA_ID  = 37;
        MALDI_ID  = 38;
        MAEAV_ID  = 39;
        MAKILL_ID = 40;
        MAEUIK_ID = 41;
        MAECIK_ID = 42;
        MALINS_ID = 43;
        MALREP_ID = 44;
        MAEDI_ID = 45;
        MAEDTI_ID = 46;
        MAECTK_ID = 47;
        MAEKND_ID = 48;
        MAKXEQ_ID = 49;
        MALN_ID   = 50;
        MAESWA_ID = 51;
        MAEUSR_ID = 52;
        MAEGKN_ID = 53;
        MASMSZ_ID = 54;
        MALOCK_ID = 55;
        MAKCNT_ID = 56;
        MAQURY_ID = 57;
        MAUPDT_ID = 58;
        MIDBD_ID = 59;
        MIDBRV_ID = 60;
        MAERST_ID = 61;
        MARSGT_ID = 62;
        MARSCR_ID = 63;
        MALRVS_ID = 64;
        MAEDTS_ID = 65;
        MALSRT_ID = 66;
        MALROR_ID = 67;
        MAECXQ_ID = 68;
        MAEUXQ_ID = 69;
        MAECMP_ID = 70;
        MAECQY_ID = 71;
    (**)
    (*------------------------------------------------------------------------*)
    (* THE EXTERNAL RETURN CODE INITIALIZATION AND OK VALUE.                  *)
    (*------------------------------------------------------------------------*)
    (**)
        NORMAL_RC=0;
    (**)
```

```
(*-------------------------------------------------------------------*)
(* THE RANGE ON AN ENTITY.                                           *)
(*-------------------------------------------------------------------*)
(**)
    MIN_ENT=0;
    MAX_ENT=65535;
(**)
(*-------------------------------------------------------------------*)
(* THE EXTERNAL NIL POINTER VALUE.                                   *)
(*-------------------------------------------------------------------*)
(**)
    NIL_PNTR=0;
TYPE
    LINE=TEXT;
    PGMNAME=PACKED ARRAY(.1..8.) OF CHAR;
    ERRMSG=PACKED ARRAY(.1..25.) OF CHAR;
(**)
(*-------------------------------------------------------------------*)
(* EXTERNAL RETURN CODE.                                             *)
(*-------------------------------------------------------------------*)
(**)
    EXT_RET_CODE=INTEGER;
(**)
CONST
(**)
(*-------------------------------------------------------------------*)
(* NON-ERROR EXTERNAL RETURN CODE VALUE.                             *)
(*-------------------------------------------------------------------*)
(**)
    OKRC=0;
(**)
(*-------------------------------------------------------------------*)
(* MAXIMUM VALUE TO BE IN THE OVERFLOW REQUESTED SIZE ARRAY          *)
(*-------------------------------------------------------------------*)
(**)
    MAX_OVRFLW=10;
(**)
(*-------------------------------------------------------------------*)
(* COMMON AREA FOR FSTART AND FSTOP ID AND ON/OFF FLAG.              *)
(*-------------------------------------------------------------------*)
(**)
TYPE
    COMMON = RECORD
        ID_FIELD:INTEGER;
        ID_FLAG:INTEGER;
        ID_ERR_PGM:INTEGER;
        ID_ERR_PGMNAME:PGMNAME;
        ID_ERR_CODE:INTEGER;
        ID_ERR_MESSAGE:ERRMSG;
```

```
        OVRFLW_COUNT:INTEGER;
        OVRFLW_ENTRY:ARRAY (.1..MAX_OVRFLW.) OF INTEGER;
    END;
(**)
%PRINT ON
(* END %INCLUDE MASTYP *)
```

```
(*-------------------------------------------------------------------*)
(*  (NDSDCL) NETWORK DATA STRUCTURE TYPES AND CONSTANTS USED BOTH    *)
(*  BY BOTH INTERNAL NDS ROUTINES AND EXTERNAL APPICATIONS.          *)
(*-------------------------------------------------------------------*)
%INCLUDE PRINTOFF
(*-------------------------------------------------------------------*)
(*                        MAS VERSION 2                              *)
(*                    PACKAGE:  NETWORK PACKAGE.                     *)
(*    CHANGE CONTROL:                                                *)
(*        02/11/85  MAS VER 2  B. A. ULMER                           *)
(*           ADD CRBPNTR TO T_INT_ROOT                               *)
(*           ADD MAX_RDB_SIZE                                        *)
(*           ADD INCREMENT VALUES FOR CONSTITUENT READ BLOCK         *)
(*           ADD T_CNST_RDBLK AND CNST_RDB_ENTRY                     *)
(*           ADD 2 NEW ERROR MESSAGES                                *)
(*           ADD RDBEXIST TO T_INT_ITEM                              *)
(*           DELETE ROOT FROM T_INT_ITEM                             *)
(*           DELETE DIRLIST FROM T_INT_ITEM                          *)
(*        04/23/85   MAS VER 2   E. D. SHREVE                        *)
(*           ADD DELTFLG TO T_APPL_LIST TO MARK LISTS FOR NONDELETE. *)
(*           ADD PROCESS FLAG (MAPROB) FOR INTERNAL MAS USAGE        *)
(*        05/21/85   MAS VER 2   B. A. ULMER                         *)
(*           ADD NO_LIST_CREATED WARNING TO LIST OF WARNINGS         *)
(*           ADD INVALID_FLAG_NAME TO LIST OF ERRORS                 *)
(*        06/13/85   MAS VER 2   B. A. ULMER                         *)
(*           ADD NAMTYP FOR PARAMETER FOR MAQURY ANS MAUPDT          *)
(*        07/11/85   MAS VER 2   B. A. ULMER                         *)
(*           ADD NUM_WARN AND NUM_ERROR FOR CHANGE TO CNVRR AND      *)
(*               MSTATUS COMMON                                      *)
(*        10/24/85   MAS VER 2   B. A. ULMER                         *)
(*           ADD RTS_NOT_IN_WORKING_FORM AND SIZE_NOT_LARGE_ENOUGH   *)
(*               TO LIST OF ERRORS                                   *)
(*        03/12/86   MAS VER 2   E. D. SHREVE                        *)
(*           ADD MAPROB2 TO THE IIT FOR SORT DELETE ROUTINES         *)
(*        03/21/86   MAS VER 2   B. A. ULMER                         *)
(*           ADD CORE_NOT_AVAILABLE, NOT_ENOUGH_CORE_FOR_INIT, AND   *)
(*            ABSOLUTELY_NO_MORE_CORE TO ERRORS FOR PROCESSING OF THE*)
(*            "OUT OF SPACE" CONDITION                               *)
(*        04/22/86   MAS VER 2   B. A. ULMER                         *)
(*           ADD NEW WARNING MESSAGE FOR MAEDTS - EMPTY_MARK_LIST    *)
(*            AND CHANGED THE NUM_WARN TO 7                          *)
(*        05/05/86   MAS VER 2   B. A. ULMER                         *)
(*           ADD NEW WARNING CODES FOR HANDLING EMPTY OUTPUT LISTS   *)
(*            FOR MAEDTS AND ERROR CODES FOR MAINIT                  *)
(*                        MAS VERSION 3                              *)
(*                    PACKAGE:  NETWORK PACKAGE.                     *)
(*        06/17/86   MAS VER 3   B. A. ULMER                         *)
(*           CHANGE STRUCTURE OF IIT FOR NEW DELETE RULES - ADD TYPE *)
(*            DEFINITIONS FOR THE SCHEMA INSTANCE COLLECTOR          *)
(*        06/18/86   MAS VER 3   B. A. ULMER                         *)
(*           ADD NEW ERROR CODES FOR SETRULS ROUTINE - ADD MAX_GROUP *)
(*           CONSTANT FOR THE INSTANCE COLLECTOR DEFINITION          *)
```

```
(*        07/01/86   MAS VER 3   B. A. ULMER                          *)
(*           CHANGE LSTNG - T_NDSGVM DUE TO NDS NAME CHANGE CONFLICTS*)
(*        07/22/86   MAS VER 3   B. A. ULMER                          *)
(*           CHANGE T_NDS_ERROR - ERROR CODE                          *)
(*                  CANT_MARK_FOR_DELETE TO SCHEMA_ROOT_NIL - BUG FIX*)
(*                  CANT_MARK_FOR_DELETE TO SCHEMA_ROOT_NIL - BUG FIX*)
(*           THAT DEALS WITH MAERST                                   *)
(*-------------------------------------------------------------------*)
(**)
CONST
(**)
(*-------------------------------------------------------------------*)
(* THE NUMBER OF CHARACTERS PER WORD IS MACHINE DEPENDENT. THE       *)
(* ASSUMPTION HAS BEEN MADE THAT AN INTEGRAL NUMBER OF CHARACTERS    *)
(* WILL OVERLAY AN INTEGER.                                          *)
(*                 IBM S/370   DEC VAX 11/780   CDC CYBER            *)
(*  CHARS_PER_WORD     4             4           10                  *)
(*-------------------------------------------------------------------*)
(**)
    CHARS_PER_WORD = 4;
(**)
(*-------------------------------------------------------------------*)
(* THE MAXIMUM SIZE OF KIND IS THE MAXIMUM INTEGER.                  *)
(*-------------------------------------------------------------------*)
(**)
    MAX_ENT_KIND = MAXINT;
(**)
(*-------------------------------------------------------------------*)
(* LIST INCREMENTS ARE USED TO CONTROL THE MEMORY VS SPEED           *)
(* TRADEOFF FOR THE MANAGEMENT OF THEIR CORRESPONDING LIST TYPES.    *)
(* THEIR VALUE CORRESPONDS TO THE NUMBER OF ENTITIES ADDED TO THE    *)
(* LIST EACH TIME GROWTH IS REQUIRED AND THE NUMBER OF ENTITIES      *)
(* REMOVED WHEN LIST COMPRESSION IS APPLICABLE.                      *)
(*-------------------------------------------------------------------*)
(**)
    APPL_LIST_INCR = 10;
    ITEM_LIST_INCR = 24;
    USER_LIST_INCR = 2;
    CNST_LIST_INCR = 2;
    LIST_OF_ROOTS_INCR  = 10;
    LIST_OF_LISTS_INCR  = 20;
    STACK_OF_LISTS_INCR = 20;
    INST_COL_CNST_INCR = 200;
    LRG_LIST_INCR_1 = 20;
    LRG_LIST_INCR_2 = 100;
    LRG_LIST_INCR_3 = 200;
                                    (* IF LIST PROCESSING CHANGED.     *)
(**)
```

```
(*-------------------------------------------------------------------*)
(* MAXIMUM NUMBER OF DIFFERENT KIND OF CONNECTIVE RELATIONSHIP AN    *)
(* ENTITY CAN HAVE WITH ITS CONSTITUENTS                   6/18/86*)
(*-------------------------------------------------------------------*)
(**)
    MAX_GROUP = 8;
(**)
(*-------------------------------------------------------------------*)
(* RECORD LENGTH FOR AN ENTRY IN THE CRB                        BAU*)
(* CONSTITUENT READ BLOCK INCREMENT FOR EXPANSION/COMPRESSION   BAU*)
(*-------------------------------------------------------------------*)
(**)
    CRB_REC_LEN = 12;
    CNST_RDBLK_INCR = 4;
(**)
(*-------------------------------------------------------------------*)
(* DUE TO THE IBM/370 24 BIT ADDRESSING LIMIT, THE MAXIMUM SIZE      *)
(* OF A ENTBLOCK IS 16777215 BYTES.                                  *)
(*-------------------------------------------------------------------*)
(**)
    MAX_ENT_SIZE  = 800000;         (*16777215          40*)
    MAX_ENT_WORDS = 200000;         (* 4194301          10*)
(**)
(*-------------------------------------------------------------------*)
(* TO FORCE VARIABLES TO A HALFWORD OF STORAGE                  BAU*)
(*-------------------------------------------------------------------*)
(**)
    MAX_RDB_SIZE = 65535;
(**)
(*-------------------------------------------------------------------*)
(* EACH ENTRY IN A LIST OCCUPIES ONE WORD. THE LIST ITSELF CONTAINS  *)
(* 1 WORD OF OVERHEAD. THE NUMBER OF ENTITIES IN A LIST ARE LIMITED  *)
(* TO THE NUMBER OF WORDS ADDRESSABLE - 1 OR (2**22)-1.              *)
(*-------------------------------------------------------------------*)
(**)
    MAX_LIST_SIZE = 4194302;
(**)
%PAGE
TYPE
    SIGNED_INT2   = PACKED -32768..32767;
    UNSIGNED_INT2 = PACKED 0..65535;
    NATURAL2      = PACKED 1..65535;
(**)
    RDBSIZE       = PACKED 0..MAX_RDB_SIZE;                  (* BAU *)
(**)
```

```
(*-------------------------------------------------------------------*)
(* THE ORDER TYPE IS USED FOR THE RESULT OF THE APPLICATION DEFINED  *)
(* ORDER FUNCTION.  THIS FUNCTION IS USED AS INPUT BY THE PROCEDURES *)
(* THAT SORT LISTS.                                                  *)
(*-------------------------------------------------------------------*)
(**)
    T_ORDER  = (N_LESS,N_EQUAL,N_GREATER);
    ORD_KIND = 0..MAX_ENT_KIND;
(**)
CONST
(**)
(*-------------------------------------------------------------------*)
(* KIND OF NIL_KIND INDICATES ENTBLOCK HAS BEEN DELETED.             *)
(*-------------------------------------------------------------------*)
(**)
    NIL_KIND = 0;
(**)
(*-------------------------------------------------------------------*)
(* NUMBER OF WARNINGS AND ERRORS                          BAU 7/11/85*)
(*-------------------------------------------------------------------*)
(**)
    NUM_WARN = -10;
    NUM_ERROR = 43;
(**)
%PAGE
TYPE
(**)
(*-------------------------------------------------------------------*)
(* INTERNAL RETURN CODES.                                            *)
(*-------------------------------------------------------------------*)
(**)
    T_NDS_ERROR =
        (OK,                        (*  0 *)
         BAD_ENT_KIND,              (*  1 *)
         INVALID_CREATE,            (*  2 *)
         CANT_CREATE_LIST,          (*  3 *)
         MAS_INIT_FAILED,           (*  4 *)
         INVALID_UPDATE,            (*  5 *)
         CANT_UPDATE_ENT,           (*  6 *)
         CANT_CREATE_ENT,           (*  7 *)
         CANT_VERIFY_CONNECT,       (*  8 *)
         INVALID_CONNECTION,        (*  9 *)
         CANT_CONNECT,              (* 10 *)
         ABSENT_INPUT,              (* 11 *)
         INVALID_GET,               (* 12 *)
         NDS_OP_COMPLETE,           (* 13 *)
         BAD_LIST_POSITION,         (* 14 *)
         MAXIMUM_LIST_SIZE,         (* 15 *)
         BAD_LIST_MOVE_COUNT,       (* 16 *)
         BAD_LIST_REFERENCE,        (* 17 *)
         BAD_ENT_KEY,               (* 18 *)
```

```
            DUPLICATE_SCH,              (* 19 *)
            DUMP_ERROR,                 (* 20 *)
            BAD_ENT_SIZE,               (* 21 *)
            BAD_SCH_KIND,               (* 22 *)
            PROC_CODE_ERROR,            (* 23 *)
            PROC_OUT_OF_RANGE,          (* 24 *)
            NO_MATCH_FOUND,             (* 25 *)
            DUPS_NOT_REMOVED,           (* 26 *)
            INVALID_DELETE,             (* 27 *)
            BAD_ENTITY_ON_USER_LIST,    (* 28 *)
            BAD_DELETE_KEY,             (* 29 *)
            EMPTY_MODEL,                (* 30 *)
            ARG_OUT_OF_RANGE,           (* 31 *)
            INVALID_CRB_POSITION,       (* 32 *)
            CRB_ENTRY_NOT_FOUND,        (* 33 *)
            INVALID_FLAG_NAME,          (* 34 *)
            SCHEMA_ROOT_NIL,            (* 35 *)
            SIZE_NOT_LARGE_ENOUGH,      (* 36 *)
            RTS_NOT_IN_WORKING_FORM,    (* 37 *)
            CORE_NOT_AVAILABLE,         (* 38 *)
            NOT_ENOUGH_CORE_FOR_INIT,   (* 39 *)
            ABSOLUTELY_NO_MORE_CORE,    (* 40 *)
            MAINIT_ALREADY_DONE,        (* 41 *)
            RULE_DOES_NOT_MATCH,        (* 42 *)
            ENTITY_NOT_FOUND_LIST    );(* 43 *)
(**)
    T_NDS_WARNING =
        (OKW,                           (*  0 *)
         NO_SUCH_SCH,                   (*  1 *)
         PROC_WARNING_CODE,             (*  2 *)
         EMPTY_DELETE_LIST,             (*  3 *)
         EMPTY_EXCEPTION_LIST,          (*  4 *)
         END_OF_LIST,                   (*  5 *)
         NO_LIST_CREATED,               (*  6 *)
         EMPTY_MARK_LIST,               (*  7 *)
         EMPTY_MARK_N_EXCEPTION,        (*  8 *)
         EMPTY_DELETE_N_EXCEPTION,(*  9 *)
         EMPTY_MARK_N_DELETE);          (* 10 *)
(**)
    RETURN_CODE = RECORD
        ERROR   : T_NDS_ERROR;
        WARNING : T_NDS_WARNING;
    END;
(**)
    RET_REC       = RECORD
        RC        : RETURN_CODE; ROUT_NAME : STRING(8);
    END;
(*-------------------------------------------------------------------*)
(* NEW DELETE AND COMPRESS RULE STRUCTURES                  6/17/86 *)
(*-------------------------------------------------------------------*)
(**)
    T_RULE_ELMNTS = (COMPRESS, DELETE, USER_DELETE, CNST_DELETE);
```

```
(**)
    T_RULE = SET OF T_RULE_ELMNTS;
(**)
    T_GROUP = RECORD
      LAST_CNST : RDBSIZE;
      RULE : T_RULE;
    END;
(**)
    T_GROUP_ARRAY = ARRAY (.1..MAX_GROUP.) OF T_GROUP;
(**)
(*-------------------------------------------------------------------*)
(* ENTITIES ARE DYNAMIC OBJECTS REFERENCED BY APPLICATION KEYS       *)
(* AND WHICH CAN BE REFERENCED BY LISTS.                             *)
(*-------------------------------------------------------------------*)
(**)
    ENTITIES = (NIL_ENT, INT_ROOT, INT_ITEM, APPL_LIST);
(**)
(*-------------------------------------------------------------------*)
(* LIST POINTERS ARE USED TO CONNECT ENTITIES TO SYSTEM LISTS.       *)
(* A SYSTEM LIST CAN NEVER BE CONNECTED TO MORE THAN ONE ENTITY.     *)
(*-------------------------------------------------------------------*)
(**)
    LISTPNTR = @T_SYS_LIST;
(**)
(*-------------------------------------------------------------------*)
(* THE SYSTEM LIST POSITION INDICATES RELATIVE POSITION WITHIN A     *)
(* SYSTEMS LIST.  IT IS NOT A COMPONENT WITH THE SYSTEM LIST TYPE    *)
(* BECAUSE ONLY SYSTEM LISTS CONNECTED TO APPLICATION LIST           *)
(* ENTITIES HAVE AN ASSOCIATED POSITION.                             *)
(*-------------------------------------------------------------------*)
(**)
    LISTPSTN = 0..MAX_LIST_SIZE;
(**)
(*-------------------------------------------------------------------*)
(* ALLOWABLE SIZE OF SYSTEMS LISTS                                   *)
(*-------------------------------------------------------------------*)
(**)
    LISTSIZE = 0..MAX_LIST_SIZE;
(**)
(*-------------------------------------------------------------------*)
(*  LIST INDEX FOR INDEXING INTO SYSTEM LISTS.                       *)
(*-------------------------------------------------------------------*)
(**)
    LISTINDX = 1..MAX_LIST_SIZE;
(**)
```

```
(*-------------------------------------------------------------------*)
(* POINTER TO A T_ENTITY IN THE NETWORK DATA STRUCTURE.              *)
(*-------------------------------------------------------------------*)
(**)
    T_KEY = @T_ENTITY;
    ENTKEY = RECORD
        P  : T_KEY;
    END;
(**)
(*-------------------------------------------------------------------*)
(* SIZE OF USER ENTITY DATA IN WORDS.                                *)
(*-------------------------------------------------------------------*)
(**)
    ENTSIZE = 0..MAX_ENT_SIZE;
(**)
(*-------------------------------------------------------------------*)
(* SIZE OF APPLICATION DATA IN CHARACTERS.                           *)
(*-------------------------------------------------------------------*)
(**)
    ENTCHARSIZE = 0..MAX_ENT_SIZE;
(**)
(*-------------------------------------------------------------------*)
(* MACHINE DEPENDENT INDEX TYPE FOR MOVING USER DATA AS CHARACTER    *)
(* ARRAYS.                                                           *)
(*-------------------------------------------------------------------*)
(**)
    ENTCHARINDX = 1..MAX_ENT_SIZE;
(**)
(*-------------------------------------------------------------------*)
(* MACHINE DEPENDENT INDEX TYPE FOR MOVING USER DATA AS WORDS IN     *)
(* SYSTEMS FORMAT.                                                   *)
(*-------------------------------------------------------------------*)
(**)
    ENTINDX = 1..MAX_ENT_WORDS;
(**)
(*-------------------------------------------------------------------*)
(* USER DATA CAN BE VIEWED AS ARRAY OF CHARACTERS OR INTEGERS.       *)
(* POINTER TO DYNAMICALLY ALLOCATED SYSTEM DATA STORAGE AREAS.       *)
(*-------------------------------------------------------------------*)
(**)
    ENTPNTR = @ENTBLOCK;
%PAGE
(**)
(*-------------------------------------------------------------------*)
(* NDS DYNAMIC OBJECTS.                                              *)
(*-------------------------------------------------------------------*)
(**)
TYPE
(**)
```

```
(*-------------------------------------------------------------------*)
(* POINTER TO CONSTITUENT READ BLOCK                            BAU*)
(*-------------------------------------------------------------------*)
(**)
    CRBPNTR = @T_CNST_RDBLK;
(**)
(*-------------------------------------------------------------------*)
(* EACH NDS HAS ONE AND ONLY ONE INTERNAL ROOT.                      *)
(*-------------------------------------------------------------------*)
(**)
    T_INT_ROOT   = RECORD
        ROOT       : ENTKEY;
        SCH_ROOT   : ENTKEY;
        CNST_RDBLK : CRBPNTR;
    END;
(**)
(*-------------------------------------------------------------------*)
(* DIRECTED LIST TYPE FOR MAINTAINING DIRECTION AND POSITION IN LISTS*)
(*-------------------------------------------------------------------*)
(**)
    LISTDIR = (FORWARD, REVERSE);
(**)
    DIRLST        = RECORD
        LIST      : LISTPNTR;
        POSITION  : LISTPSTN;
        DIRECTION : LISTDIR;
    END;
(**)
(*-------------------------------------------------------------------*)
(* CONSTITUENT READ BLOCK FOR MAINTAINING "POSITION" AND "DIRECTION" *)
(* IN CONSTITUENT LISTS                                         BAU*)
(*-------------------------------------------------------------------*)
(**)
    CNST_RDB_ENTRY = RECORD
        ENT       : ENTKEY;
        POSITION  : LISTPSTN;
        DIRECTION : LISTDIR;
    END;
(**)
    T_CNST_RDBLK = RECORD
        SIZE           : RDBSIZE;
        CRBLNG         : RDBSIZE;
        CNST_RDBLK_ARY : ARRAY(.RDBSIZE.) OF CNST_RDB_ENTRY;
    END;
(**)
```

```
(*-------------------------------------------------------------------*)
(* EACH NDS HAS ONE INTERNAL ITEM FOR EACH ENTBLOCK.                 *)
(*-------------------------------------------------------------------*)
(*  T_INT_ITEM = RECORD
        RDBEXIST : BOOLEAN;
        MAPROB   : BOOLEAN;     ADDED 04/23/85 - MAS PROCESS FLAG
        MAPROB2  : BOOLEAN;     ADDED 12/03/85 - FOR DELETE     EDS
        USERS    : LISTPNTR;
        CNSTS    : LISTPNTR;
        ENPTR    : ENTPNTR;
    END;                          *)
(**)
    T_INT_ITEM = RECORD
        RDBEXIST : BOOLEAN;
        MAPROB   : BOOLEAN;  (* ADDED 04/23/85 - MAS PROCESS FLAG   *)
        MAPROB2  : BOOLEAN;  (* ADDED 12/03/85 - FOR DELETE     EDS *)
        COLL_ADB : ENTPNTR;  (* ADDED 06/17/86 - FOR DELETE AND     *)
                             (*                  AND COMPRESS   BAU *)
        USERS    : LISTPNTR;
        CNSTS    : LISTPNTR;
        ENPTR    : ENTPNTR;
    END;
(**)
(*-------------------------------------------------------------------*)
(* THE APPLICATION LIST IS NOT PART OF THE NDS.  THERE IS ONE        *)
(* APPLICATION LIST FOR EACH LIST CREATED BY AN APPLICATION.         *)
(* SOME NDS UTILITIES ALSO CREATE APPLICATION LISTS.                 *)
(*-------------------------------------------------------------------*)
(**)
    DELFLG = (DEL, NODEL);      (* ADDED 4/23/85 *)
(**)
    T_APPL_LIST   = RECORD
        LIST      : LISTPNTR;
        POSITION  : LISTPSTN;
        DIRECTION : LISTDIR;
        DELTFLG   : DELFLG;      (* ADDED 4/23/85 *)
    END;
(**)
    T_ENTITY = RECORD
        FORM : ENTITIES;
        CASE ENTITIES OF
            NIL_ENT   : ();
            INT_ROOT  : (IRT : T_INT_ROOT);
            INT_ITEM  : (IIT : T_INT_ITEM);
            APPL_LIST : (APL : T_APPL_LIST);
    END;
(**)
```

```
(*-------------------------------------------------------------------*)
(* THE SYSTEM LIST IS A VARIABLE LENGTH DYNAMICALLY ALLOCATED        *)
(* STRUCTURE CONTAINING A LIST OF ENTITY REFERENCES                  *)
(*-------------------------------------------------------------------*)
(**)
    T_SYS_LIST = RECORD
        SIZE   : LISTSIZE;
        LSTLNM : LISTSIZE;
        LIST   : ARRAY(.LISTINDX.) OF ENTKEY;
    END;
%PAGE
(**)
(*-------------------------------------------------------------------*)
(* USER OBJECTS                                                      *)
(*-------------------------------------------------------------------*)
(**)
TYPE
    ENTRANGE = RECORD
        LOW  : ORD_KIND;
        HIGH : ORD_KIND;
    END;
(**)
    NDS     = RECORD
        KEY : ENTKEY;
    END;
(**)
    LISTKEY = RECORD
        P   : T_KEY;
    END;
(**)
    ANYKEY = RECORD
        CASE INTEGER OF
            0 : (ENTK : ENTKEY);
            1 : (LSTK : LISTKEY);
    END;

%PAGE
TYPE
(**)
(*-------------------------------------------------------------------*)
(* GLOBAL VARIABLE RECORD                                            *)
(*-------------------------------------------------------------------*)
(**)
    T_NDSGVM           = RECORD
        LIST_OF_ROOTS  : LISTPNTR;
        STACK_OF_LISTS : LISTPNTR; (* MAY NOT BE NEEDED IN FUTURE     *)
                                   (* IF LIST PROCESSING CHANGED.     *)
        MODEL_HEAP : T_KEY;
    END;
```

```
(*------------BAU 6/13-------------------------------------------------*)
(**)
    NAMTYP = PACKED ARRAY (.1..6.) OF CHAR;
%PRINT ON
(* END %INCLUDE NDSDCL *)
```

```
(*************************************************************************)
(* $PCMGT - INCLUDE FILE TO DEFINE TYPES FOR SPACE MANAGEMENT PROCS   *)
(*************************************************************************)
%PRINT OFF
CONST
   XCB_SIZE =  8;
   $CB_SIZE = 16;

TYPE
   XCBP       = @XCB;
   $CBP       = @$CB;

   XCB        = RECORD
                  FREE:     XCBP;
                  SIZE:     INTEGER;
                END;

   $CB        = RECORD
                  NEXT:     $CBP;
                  BIGGEST:  INTEGER;
                  FIRST:    XCB;
                END;

   MEMCONV$   = RECORD
                CASE INTEGER OF
                0:  (I:  INTEGER);
                1:  ($:  $CBP);
                2:  (X:  XCBP);
                END;

   T_$PCMGR   = RECORD
                  SIZE:  INTEGER;
                  PTR:   $CBP;
                END;

%PRINT ON
```

```
(*************************************************************************)
(* PCMGT - INCLUDE FILE TO DEFINE TYPES FOR SPACE MANAGEMENT PROCS   *)
(*************************************************************************)
%PRINT OFF
CONST
   XCB_SIZE =  8;
   $CB_SIZE = 16;
   OSIZE    =  32368;  (* 16184   32368   8092 *)

TYPE
   XCBP    = @XCB;
   $CBP    = @$CB;

   XCB     = RECORD
               FREE:     XCBP;
               SIZE:     INTEGER;
             END;

   $CB     = RECORD
               NEXT:     $CBP;
               BIGGEST:  INTEGER;
               FIRST:    XCB;
             END;

   MEMCONV$  = RECORD
             CASE INTEGER OF
             0:  (I:  INTEGER);
             1:  ($:  $CBP);
             2:  (X:  XCBP);
             END;

   T_$PCMGR  = RECORD
               SIZE:     INTEGER;
               PTR:      $CBP;
               OVERFLOW: $CBP;
               OFLAG:    BOOLEAN;
             END;

%PRINT ON
```

```
(*-------------------------------------------------------------------*)
(*  (SCHDCL) NETWORK DATA STRUCTURE SCHEMA TYPES AND CONSTANTS       *)
(*-------------------------------------------------------------------*)
%INCLUDE PRINTOFF
(*-------------------------------------------------------------------*)
(*                      MAS VERSION 1                                *)
(*                   NDS SCHEMA OBJECTS                              *)
(*                PACKAGE:  SCHEMA PACKAGE.                          *)
(*-------------------------------------------------------------------*)
(**)
%INCLUDE NDSDCL
%INCLUDE SCHTYP
TYPE
(*-------------------------------------------------------------------*)
(* USER DATA CAN BE VIEWED AS ARRAY OF CHARACTERS OR INTEGERS.       *)
(*-------------------------------------------------------------------*)
    ENTDATA=RECORD
      CASE INTEGER OF
        0:(CHARS:PACKED ARRAY(.ENTCHARINDX.) OF CHAR);
        1:(INTS:ARRAY(.ENTINDX.) OF INTEGER); 2:(SCH_ROOTS:T_SCH_ROOT_ENT);
        3:(SCH_INSTS:T_SCH_INST_ENT);
        4:(SCH_CLS:T_SCH_CLS_ENT);
      END (* ENTDATA *);
%INCLUDE ENTBLK;
(**)
%PRINT ON
(* END %INCLUDE SCHDCL *)
```

```
(*-------------------------------------------------------------------*)
(*  (SCHTYP) NETWORK DATA STRUCTURE SCHEMA TYPES AND CONSTANTS       *)
(*-------------------------------------------------------------------*)
%INCLUDE PRINTOFF
(*-------------------------------------------------------------------*)
(*                      MAS VERSION 1                                *)
(*                   NDS SCHEMA OBJECTS                              *)
(*                PACKAGE:  SCHEMA PACKAGE.                          *)
(*                      MAS VERSION 2                                *)
(*                   NDS SCHEMA OBJECTS                              *)
(*                PACKAGE:  SCHEMA PACKAGE.                          *)
(*  CHANGED: B. A. ULMER (LJB)                DATE: 09/04/85         *)
(*  REASON:  ADD TWO NEW DELETE RULES (REQUIRES USER(S) TO EXIST-    *)
(*       REQUIRES CONSTITUENT(S) TO EXIST)                           *)
(*                      MAS VERSION 3                                *)
(*                   NDS SCHEMA OBJECTS                              *)
(*                PACKAGE:  SCHEMA PACKAGE.                          *)
(*  CHANGED: B. A. ULMER                      DATE: 06/18/86         *)
(*  REASON:  CHANGE STRUCTURE OF THE SCHEMA INSTANCE COLLECTOR SO AS *)
(*       HANDLE NEW DELETE AND COMPRESS RULES                        *)
(*-------------------------------------------------------------------*)
(**)
CONST
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA FAST ARRAY MUST HAVE A FIXED NUMBER OF ENTRIES IN      *)
(* PASCAL.                                                           *)
(*-------------------------------------------------------------------*)
(**)
   MAX_SCH_FST_SIZE = 1;
(**)
TYPE
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA KIND IS ASSIGNED THE ORDINAL OF A SCALAR IN SCH_KIND.  *)
(*-------------------------------------------------------------------*)
(**)
   SCH_KIND=(NIL_SCH,SCH_ROOT,SCH_INST,SCH_CLASS);
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA FAST INDEX IS USED FOR INDEXING INTO THE SCHEMA FAST   *)
(* ACCESS LIST.                                                      *)
(*-------------------------------------------------------------------*)
(**)
   SCH_FST_INDX = 1..MAX_SCH_FST_SIZE;
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA ROOT IS AN INTERNAL ITEM IN ENTITY FORMAT CONTAINING   *)
(* 'USER' DATA REQUIRED TO SEARCH THE ENTITIES IN THE MODEL BY KIND. *)
(*-------------------------------------------------------------------*)
(**)
```

```
(*-------------------------------------------------------------------*)
(* T_SCH_PSTN ALLOWS SEQUENTIAL STYLE ACCESS OF ENTIRE SCHEMA IN THE *)
(* SAME MANNER AS SEQUENTIAL READ OF AN APPLICATIONS LIST.  CONTAINS *)
(* POSITION OF CURRENT SCHEMA INSTANCE ENTITY WITHIN SCHEMA AND      *)
(* POSITION OF CURRENT ENTITY WITHIN SCHEMA INSTANCE ENTITY.         *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_PSTN = RECORD
        SCH_PSTN : LISTPSTN;
        ENT_PSTN : LISTPSTN;
    END (* T_SCH_PSTN *);
(**)
(*-------------------------------------------------------------------*)
(* ARRAY_SIZES IN THE SCHEMA_ROOT DEFINE THE AREA OCCUPIED BY THE    *)
(* FAST AND STANDARD ARRAYS AND THE PORTION OF THAT AREA CONTAINING  *)
(* VALID DATA.                                                       *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_ARY_SIZES = RECORD
        FST_ARY_LEN : LISTSIZE;
        FST_ARY_USED_LEN : LISTSIZE;
        STD_ARY_LEN : LISTSIZE;
        STD_ARY_USED_LEN : LISTSIZE;
    END (* T_SCH_ARY_SIZES *);
(**)
(*-------------------------------------------------------------------*)
(* THE FAST ARRAY IS USED FOR A PARTIAL SEARCH FOR THOSE KINDS       *)
(* DETERMINED TO BE ACCESSED MORE OFTEN THAN THE USUAL CASE.         *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_FST_ENTRY = RECORD
        KIND : ORD_KIND;
        STD_INDX : LISTINDX;
    END (* T_SCH_FST_ENTRY *);
    T_SCH_FST_ARY = ARRAY (.SCH_FST_INDX.) OF T_SCH_FST_ENTRY;
(**)
(*-------------------------------------------------------------------*)
(* THE STANDARD ARRAY CONTAINS EACH KIND IN THE SAME SEQUENCE AS ITS *)
(* CORRESPONDING SCHEMA APPEARS IN THE CONSTITUENT LIST OF THE       *)
(* SCHEMA_ROOT.                                                      *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_STD_ENTRY = RECORD
        KIND : ORD_KIND;
    END (* T_SCH_STD_ENTRY *);
    T_SCH_STD_ARY = ARRAY (.LISTINDX.) OF T_SCH_STD_ENTRY;
(**)
```

```
(*-------------------------------------------------------------------*)
(* THE SCHEMA_ROOT_ENT MAY BE HANDLED AS ANY OTHER ENTDATA BY       *)
(* ROUTINES USING THE STANDARD DEFINITION OF ENTDATA.               *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_ROOT_ENT = RECORD
        POSITION : T_SCH_PSTN;
        ARY_SIZES : T_SCH_ARY_SIZES;
        FST_ARY : T_SCH_FST_ARY;
        STD_ARY : T_SCH_STD_ARY;
    END (* T_SCH_ROOT_ENT *);
    SCHRPNTR = @T_SCH_ROOT_ENT;
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA_INST_ENT MAY BE HANDLED AS ANY OTHER ENTDATA BY       *)
(* ROUTINES USING THE STANDARD DEFINITION OF ENTDATA.               *)
(*-------------------------------------------------------------------*)
(*
    T_SCH_INST_ENT = RECORD
        KIND : ORD_KIND;
        POSITION : LISTPSTN;
        RULE_DEP : BOOLEAN;
        RULE_STRNGTH : BOOLEAN;
        RULE_REQ_USER : BOOLEAN;                          LJB BAU
        RULE_REQ_CNST : BOOLEAN;                          LJB BAU
    END;   T_SCH_INST_ENT
    SCHIPNTR = @T_SCH_INST_ENT;          *)
(**)
    T_SCH_INST_ENT = RECORD
        KIND : ORD_KIND;
        POSITION : LISTPSTN;
        NUM_GROUP : LISTPSTN;
        MIN_CNST : LISTPSTN;
        GROUP : T_GROUP_ARRAY;
    END (* T_SCH_INST_ENT *);
    SCHIPNTR = @T_SCH_INST_ENT;
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA_CLS_ENT MAY BE HANDLED AS ANY OTHER ENTDATA BY        *)
(* ROUTINES USING THE STANDARD DEFINITION OF ENTDATA.               *)
(*-------------------------------------------------------------------*)
(**)
    T_SCH_CLS_ENT = RECORD
        POSITION : LISTPSTN;
        KIND : ORD_KIND;
        END (* T_SCH_CLS_ENT *);
        SCHCPNTR = @T_SCH_CLS_ENT;
(**)
```

```
CONST
(**)
    SCH_IN_MIN_SIZE = SIZEOF (T_SCH_INST_ENT);
    SCH_CL_MIN_SIZE = SIZEOF (T_SCH_CLS_ENT);
(**)
(*-------------------------------------------------------------------*)
(* THE SCHEMA ROOT CONTAINS A USER DATA BLOCK WHOSE SIZE DEPENDS     *)
(* UPON THE NUMBER OF KINDS MODELED IN THE CURRENT SCHEMA. ITS       *)
(* MINIMUM SIZE IS THE SIZE OF THE STATIC FIELDS PLUS THE MINUMUM    *)
(* SIZE OF EACH DYNAMIC FIELD. ITS MAXIMUM SIZE IS THE SIZE OF THE   *)
(* STATIC FIELDS PLUS THE MAXIMUM SIZE OF EACH DYNMAIC FIELD         *)
(*-------------------------------------------------------------------*)
(**)
    SCH_RT_MIN_SIZE=SIZEOF(T_SCH_ROOT_ENT)-SIZEOF(T_SCH_STD_ARY);
    SCH_RT_MAX_SIZE=SIZEOF(T_SCH_ROOT_ENT);
(**)
(*-------------------------------------------------------------------*)
(* MAX NUMBER OF T_SCH_INST_ENT THAT A T_SCH_CLASS_ENT CAN COLLECT   *)
(*-------------------------------------------------------------------*)
CONST
   MAX_NUM_KIND=50;
TYPE
   KIND_ARRAY=ARRAY(.1..MAX_NUM_KIND.) OF ORD_KIND;
%PRINT ON
(* END %INCLUDE SCHTYP *)
```

```
%PAGE
(* (TVERIFY) VERIFY COMMON TYPE. *)
%INCLUDE PRINTOFF
(*--------------------------------------------------------------------*)
(*                          MAS VERSION 1                             *)
(*--------------------------------------------------------------------*)
(**)
TYPE
  VERIFY_COMMON=RECORD
    UPDATE:BOOLEAN;
    CREATE:BOOLEAN;
    GET:BOOLEAN;
    CONNECT:BOOLEAN;
    DELETE:BOOLEAN;
  END;
%PRINT ON
(* END %INCLUDE TVERIFY *)
```

APPENDIX G

PDDI DATA DICTIONARY (SCHEMA)

This appendix provides a data dictionary for the PDDI software. This data dictionary is a listing of the entities and attributes in the PDDI Schema. The schema provides the complete and logical view of all the data in PDDI.

An entity is a collection of related attributes treated as a unit. The tables on the following pages are listed by entity. The entity is the first line in the table. All other entries in the table are attributes of the entity.

Mapping into the Working Form using the PDDI Data Dictionary G-2
Entity Type Codes G-26
Data Dictionary G-31

UNDERSTANDING AND MAPPING INTO THE WORKING FORM MODEL

Page

Overview G-3

Mapping Conceptual Schema Entities To Physical Entities. G-4
CS to ADB G-4
CS to CL. G-6
CS Structures to Subentities. G-8

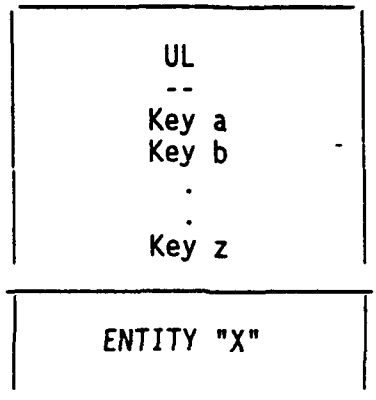
Mapping Into The Working Form Using the Pascal Include FILES G-10
Kind Constants. G-11
Entity Declarations (ADB) G-12
Entity Declaration (ADB/CL) G-13
Key Declarations (CL) G-14
Class Declarations. G-15
Category Declaration. G-16

Mapping Into the Working Form Using the PDDI Data Dictionary G-17
Attribute Records G-17
First Record G-18
2nd Through n Records G-18

UNDERSTANDING AND MAPPING INTO THE WORKING FORM MODEL

OVERVIEW

This document explains the conversion of PDDI Conceptual Schema entities into Work Form entities and two methods for mapping into these Working Form entities. The Working Form is a computer memory resident network created by the PDDI access software. An entity is comprised of a User List (UL), a Constituent List (CL) and a Attribute Data Block (ADB) as illustrated below.



CL		ADB
--		---
Key e		bytebytebytebytebytebytebytebyte
Key f		bytebytebytebytebytebytebytebyte
.		bytebytebytebytebytebytebytebyte
.		bytebytebytebytebytebytebytebyte
Key m		bytebytebytebytebytebytebytebyte
		byte.....byte

The UL is a list of entities keys that reference Entity "X". This list is automatically generated by the access software. The CL is a list of entities keys that Entity "X" references. The ADB is a sequence of data bytes that describe the entity.

MAPPING CONCEPTUAL SCHEMA ENTITIES TO PHYSICAL ENTITIES

This section presents the details on mapping the data documented in the "PDDI Conceptual Schema" into physical entities residing in the PDDI working form.

Generally Conceptual Schema (CS) attributes that are defined as INTEGER, REAL, STRING, LOGICAL, or, ENUMERATION will reside in the ADB of the working form entity. Attributes that are defined as POINTER will reside as references in the Constituent List (CL) of the working form entity.

Mapping rules describe how CS attributes are mapped into the ADB of an entity and into the CL of an entity. These specific rules are as follows:

CS TO ADB

INITIAL ADB INFORMATION - The first 24 bytes of the ADB must contain the same type information. This is:

1) KIND - A four byte integer specifying the entity type.
   For example KIND = 4001 implies this entity is a "POINT",
   KIND = 8003 implies this entity is a "LOOP"
2) LENGTH - A four byte integer specifying the length in bytes of the ADB.
3) SYSUSE - A four byte integer used by the access software.
4) VERSION - A four byte integer used to indicate the version of the entity.
5) SYSIDENT - A four byte integer used as an Access Software identification number
6) IDENT - A four byte integer used to differentiate this instance from all other instances of the entity KIND.

Only the IDENT attribute is listed in the CS. If the CS entity is defined as containing "DISPLAY" data then that DISPLAY data will follow the IDENT attribute in the ADB.

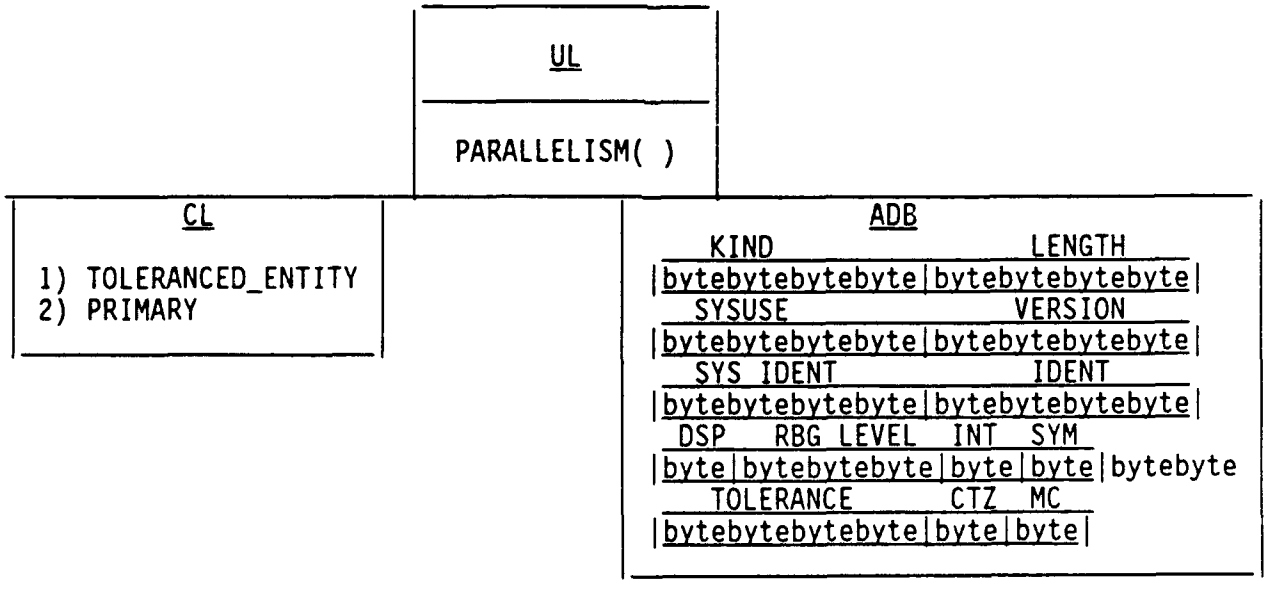
SUBSEQUENT ADB INFORMATION - The mapping of attributes from the CS to the ADB from this point on is not a direct mapping. The remaining CS attributes to be put in the ADB, reside at a position in the ADB that provides optimal use of space. The mapping rules to determine a CS's attribute location in the ADB are as follows:

- Attributes that must reside on double word boundaries (8 byte REALS) are first.
- Attributes that must reside on single word boundaries (4 byte REALS, and 4 byte INTEGERS) are second.
- Attributes that must reside on half word boundaries (2 byte INTEGERS) are third.
- Attributes that must reside on one byte boundaries (1 byte INTEGERS, LOGICALS, ENUMERATIONS, and STRINGS) are fourth.

An example of the mapping of CS attributes to ADB is illustrated below.

```
PARALLELISM = ENTITY(9011);
  IDENT              : KEY T_IDENT;
  DISPLAY            : T_DISPLAY;
  TOLERANCED_ENTITY  : ARRAY (i to *) OF
                       FEATURE, FACE,
                       EDGE, GEOMETRY;
  CYLIN_TOL_ZONE     : LOGICAL;
  TOLERANCE          : REAL PRECISION (6);
  MAIL_COND          : (N, M, L, S);
  PRIMARY            : DATUM_FRAME;
END_ENTITY;
```

```
T_DISPLAY =
 STRUCTURE;
  DISPLAYED : LOGICAL;
  RBG_LEVEL : ARRAY(3) OF
              INTEGER VALUES
              (0 TO 255);
  INTENSITY : INTEGER VALUE
              (0 to 255);
  SYMBOL    : INTEGER VALUE
              (0 to 255);
 END_STRUCTURE;
```



CS TO CL

As previously mentioned CS attributes that are defined as POINTER reside as references in the CL. Attributes that are defined as "ARRAY OF POINTERS" may not reside in the subject entities CL as references. They are defined in the working form as follows:

ARRAY OF POINTERS - If an attribute in the CS is defined as an ARRAY OF POINTERS then all of the entities to be referenced are referenced in a "ARRAY" entity. This ARRAY entity is referenced in the CL of the subject entity. An ARRAY entity contains at least one constituent and inherits the properties of its parent entity. It is identified by the unique kind, 1100. The ARRAY entity is needed to ensure that the CL of all entities is a fixed length. The programming procedures used in creating, querying, and manipulating entities in the working form are standard since all entities have a fixed length.

OPTIONAL POINTERS - If an attribute in the CS is defined as an OPTIONAL POINTER then the entity that is to be referenced in the CL is a "NIL" entity. A NIL entity has no constituents and is identified by the unique kind, 1307. The NIL entity was created to ensure that the CL of all entities is a fixed length.

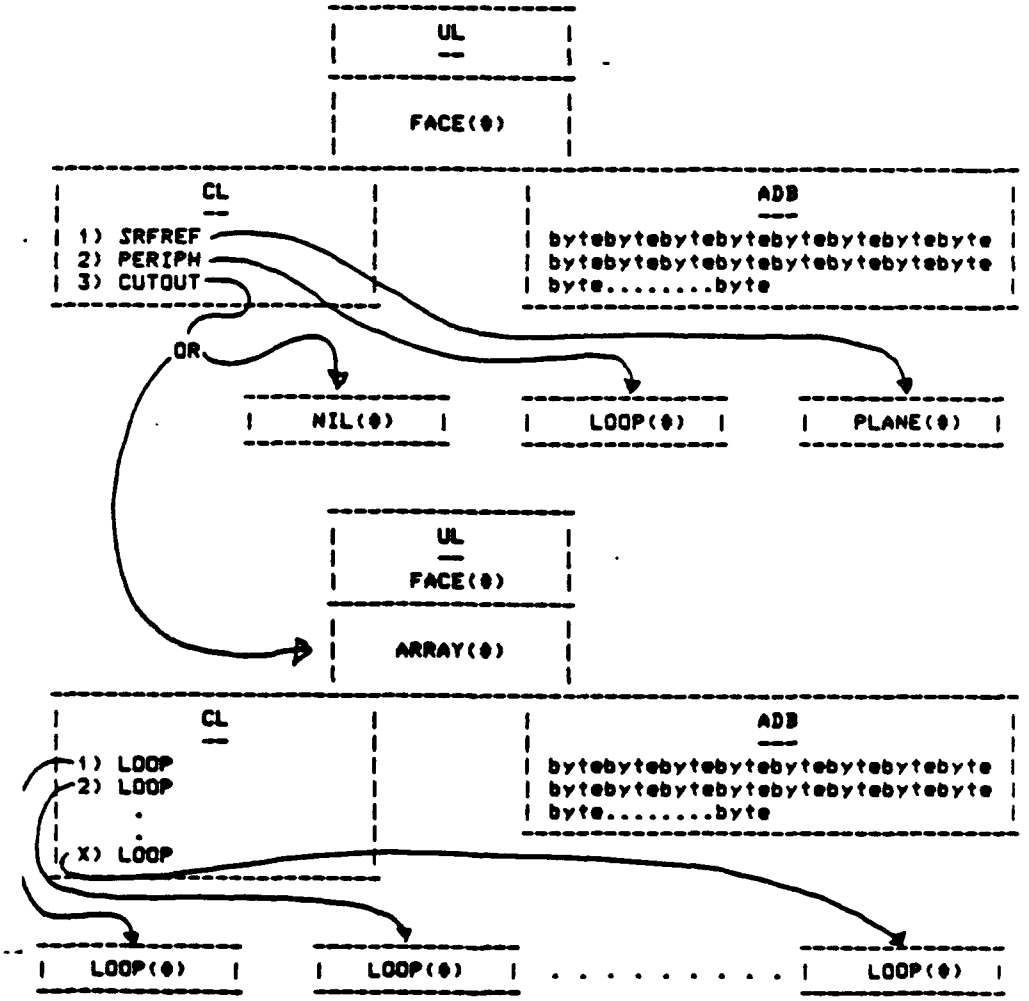
See example of the mapping of CS attributes to CL illustrated on Page G-6.

MAPPING CS ATTRIBUTES TO CL

CS Listing for the Entity Face

```
FACE = ENTITY(8004);
  IDENT     : KEY T_IDENT:
  DISPLAY   : T_DISPLAY:
  SRFREF    : SURFACES;
  REVERSED  : LOGICAL;
  PERIPH    : LOOP;
  CUTOUT    : ARRAY(0 to *) OF LOOP;
END_ENTITY;
```

WF Illustration for the Entity Face



CS STRUCTURES TO SUBENTITIES

A third CS construct is the STRUCTURE. The STRUCTURE is an unordered hetergenous collection of arbitrary attributes. The CS STRUCTURE may be contained in the entity of the working form in one of three ways. If the STRUCTURE in the CS contains:

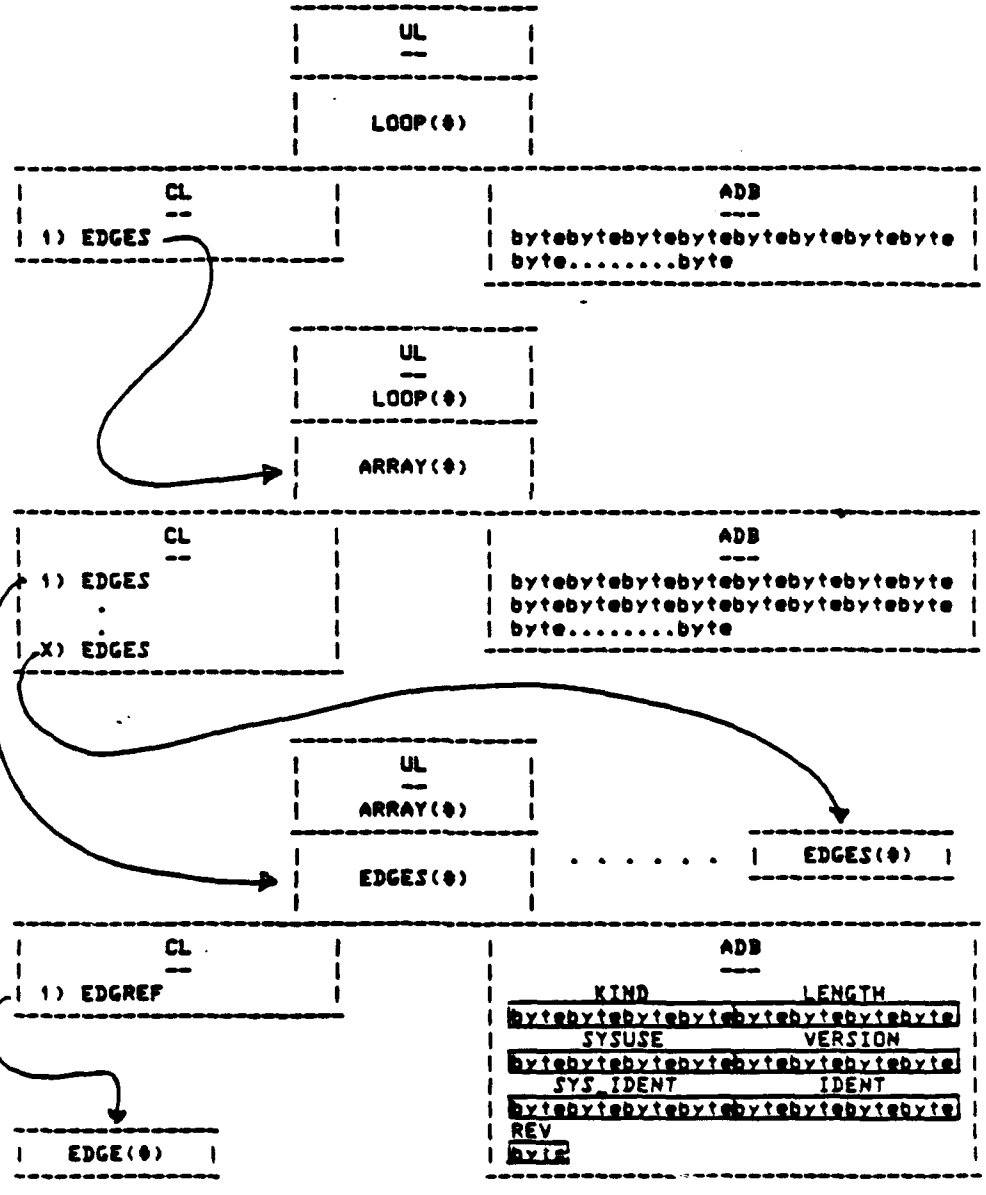
1) Only DATA (INTEGER, REAL, STRING, LOGICAL, ENUMERATION) attributes, then those attributes will reside in the ADB of the entity referencing the structure.
2) Only POINTER attributes, then those attributes will reside in the CL of the entity referencing the structure.
3) DATA and POINTER attributes, then a SUBENTITY is created that contains all the attributes defined in that STRUCTURE. This SUBENTITY is referenced as the last constituent in the CL of the subject entity. STRUCTURE is used in the CS to indicate that a relationship exists between some subset of attributes within the entity. Since (1) the CS DATA attributes are mapped into the ADB of the working form entity and (2) the CS POINTER attributes are mapped into the CL of the working form entity, the subject working form entity has no means of maintaining the relationship defined by the CS STRUCTURE construct. Therefore, in order to maintain this relationship in the working form, a SUBENTITY is generated that contains only the ADB and CL attributes that the CS STRUCTURE defined. This newly created entity becomes a reference in the CL of the subject entity.

An example of this is illustrated below.

CS Listing for the Entity - Loop

```
LOOP = ENTITY(8003);
  IDENT   : KEY T_IDENT:
  DISPLAY : T_DISPLAY:
  EDGES   : ARRAY(2 to *) of
              STRUCTURE
                EDGREF   : EDGE;
                REVERSE  : LOGICAL;
              END_STRUCTURE;
END_ENTITY;
```

WF Illustration for the Entity LOOP



MAPPING THE PDDI PASCAL INCLUDE FILES INTO THE WORKING FORM

This section details the access to the Working Form Entity by the PDDI PASCAL INCLUDE FILES. It consists of five parts.

1) Kind Constants - Identifying integers assigned to each entity type in the schema.

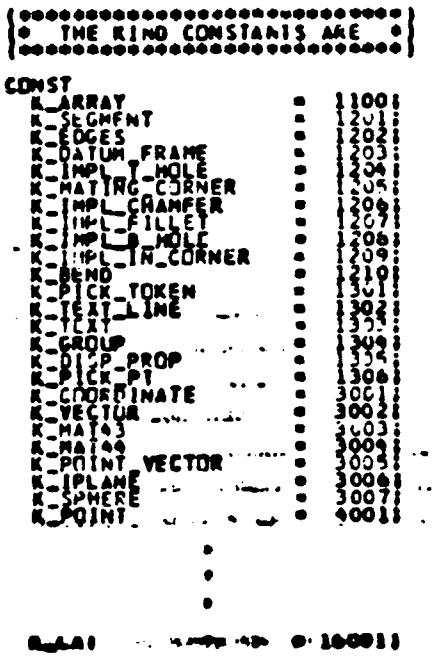
2) Entity Declarations - Mapping of ADB and Constituent List data of the Working Form entity.

3) Key Declarations - A special technique of mapping Constituent List data of the Working Form entity.

4) Class Declarations - An enumeration of entities by Class.

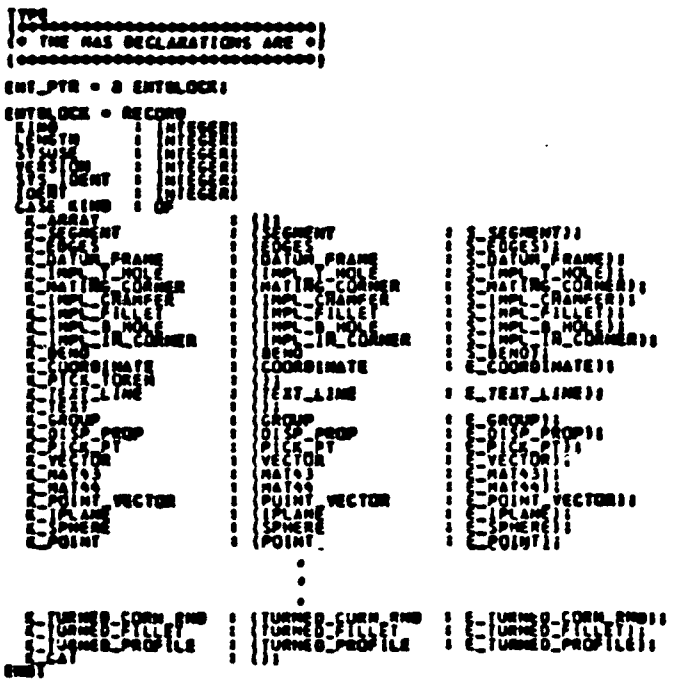
5) Category Constants - An enumeration of all the entities in the schema.

KIND CONSTANTS



- GIVEN AN ADB THE "KIND CONSTANTS" ARE USED TO IDENTIFY THE TYPE OF ENTITY.
- USING CONSTANT NAMES INSTEAD OF KIND NUMBERS WILL PREVENT CODE CHANGES IF KIND NUMBERS CHANGE.

ENTITY DECLARATIONS (ADB)



- THIS SECTION MARKS THE BEGINNING OF THE ATTRIBUTE DATA BLOCK (ADB) DECLARATIONS.
- " " ALLOWS SPACE TO BE DYNAMICALLY ALLOCATED AND RELEASED.
  - EXAMPLE:

```
VAR
  ADB : ENT_PTR;

BEGIN;
  MARK;
  NEW (ADB);
    :
  RELEASE;
 END;
```

- FIXED PART OF ADB
  - KIND, LENGTH, SYSUSE, VERSION, SYS_IDENT, IDENT
  - TO ACCESS THIS DATA

    ADB.KIND = THE ENTITY KIND

ENTITY DECLARATION (ADB/CL)

```
(***** FACE                *****)
TYPE
  E_FACE = RECORD
    REVERSE : BOOLEAN;
  END;

  C_T_FACE = RECORD
    LOOPREF : [NODE];  (* POINTERS *
              SURFACE_CLASS *)
    PERIPH  : [NODE];  (* POINTERS *
              LOOP *)
    CUTOUT  : [NODE];  (* ARRAY(.0..0.) OF POINTERS *
              LOOP *)
  END;

CONST
  C_FACE = C_T_FACE( 1, 2, 3);

TYPE
  KEY_T_FACE = RECORD
    LOOPREF : [ENTKEY];  (* POINTERS *
              SURFACE_CLASS *)
    PERIPH  : [ENTKEY];  (* POINTERS *
              LOOP *)
    CUTOUT  : [ENTKEY];  (* ARRAY(.0..0.) OF POINTERS *
              LOOP *)
  END;
(***** FACE                *****)
```

- O E_FACE IS THE RECORD CORRESPONDING TO THE VARIANT PART OF THE FACE ADB.
  - TO ACCESS THE REVERSE FLAG OF FACE

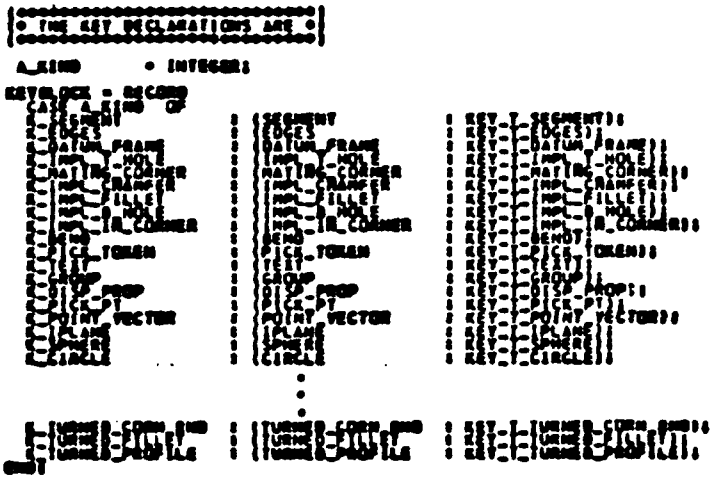
    ADB.FACE.REVERSE

- O C_FACE AND KEY_T_FACE ARE TWO DIFFERENT WAYS TO ACCESS THE CONSTITUENT LIST OF THE FACE ENTITY.
- O C_FACE:
  - TO ACCESS THE SRFREF KEY OF FACE

    MALGTK(FACE_KEY,C_FACE,SRFREF, SRFREF_KEY,IRC)
  - TO ACCESS THE ADB OF :SRFREF

    MAEGTK(SRFREF_KEY,ADB,IRC)

KEY DECLARATIONS (CL)

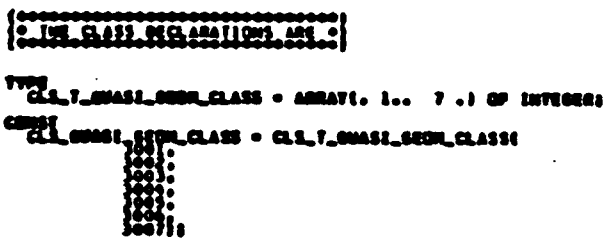


```
0  KEY_T_FACE:

   VAR

      CL_KEY : KEYBLOCK;

   -  TO ACCESS THE SRFREF KEY OF FACE

         MALGET(FACE_KEY,CL_KEY,IRC)

   -  TO ACCESS THE ADB OF SRFREF

         MAEGTK(CL_KEY,FACE,SRFREF,ADB,IRC)
```

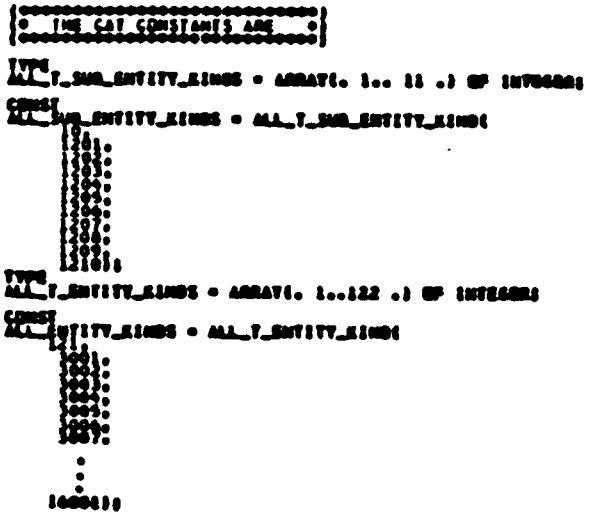
CLASS DECLARATIONS

0 LISTS ALL THE ENTITY KINDS IN A CLASS

- CLS_QUASI_GEOM_CLASS(.2.) = 3002



CATEGORY CONSTANTS DECLARATIONS



- LISTS ALL THE SUBENTITY AND ENTITY KINDS
  - ALL_SUB_ENTITY_KINDS(.1.) =

    THE NUMBER OF SUBENTITIES IN THE ARRAY
  - ALL_SUB_ENTITY_KINDS(.2.) = 1201;

MAPPING INTO THE WORKING FORM
USING THE PDDI DATA DICTIONARY

Pascal Include Files are the primary method for accessing the data in the PDDI entities. The Data Dictionary shows the way Pascal stores this data in the entity. It is intended to be used by FORTRAN and Assembler applications to access the data.

The data dictionary is a set of entity definitions where each entity is defined in a different member of an IBM partitioned dataset. An entity definition, therefore each member of the dataset, is accessed by "dataset.name (#entity_kind_number)". In this way, only the entity kind number is needed to access its corresponding entity definition.

An example of an entity definition is illustrated below.

```
1st rec     IMPL_B_HOLE      , 1204,12,1
2nd rec     KIND             , 1,1, 0,1,  4,   0
            LENGTH           , 2,1, 0,1,  4,   4
            SYSUSE           , 3,1, 0,1,  4,   8
            VERSION          , 4,1, 0,1,  4,  12
            SYS_IDENT        , 5,1, 0,1,  4,  16
      t     IDENT            , 6,1, 0,1,  4,  20
      h     DIAMETER         , 9,1, 0,2,  4,  24
      r     PARM             ,12,1, 0,2,  4,  28
      o     HOLE_TYPE        ,11,1, 0,5,  1,  32
      u   X  3,CONE             ,BULL             ,BALL
      g     LOCATE           , 7,1, 0,7,  4,   1
      h   X  1, 4000
            AXIS             , 8,1, 0,7,  4,   2
          X  1, 3002
            BOTTOM           ,10,1, 0,7,  4,   3
          X  1, 4000
            ENTRY            ,13,1, 1,7,  4,   4
          X  1, 8004
nth rec   A  1,254
```

ATTRIBUTE RECORDS

The definition of an entity is comprised of many attribute records. With the exception of the first record, each record defines an attribute or part of an attribute within the entity.

FIRST RECORD

The first record of the entity definition contains four (4) pieces of information about the entity. Below is an example of this record.

```
   (1)                  (2)   (3)  (4)
IMPL_B_HOLE           , 1204,  13,   1
```

(1) ENTITY NAME field (16 character alpha)

(2) ENTITY KIND field

(3) NUMBER OF ATTRIBUTES FIELD (2 character numeric)

(4) DELETABILITY STATUS field (1 character) 0 or 1

- Zero (0) implies this entity is independent. This entity does not require a parent entity to exist and can be deleted at any time.
- One (1) implies this entity is dependent. This entity requires a parent entity to exist and cannot be deleted if its parent is present.

SECOND - n RECORDS

The second through n records of an entity definition contains the definition of each attribute within the entity. The different definition fields contained in each of these records are as follows:

```
(1)    (2)          (3)(4)(5)(6)(7) (8)
 A, DIAMETER        , 9,1, 0,2,  4,  24
```

(1) CONTINUATION FLAG field (1 character alpha A, B, E, or X)

- A letter signifying additional data about an attribute is contained on this line. There are four possible continuation flags.

  "X" - Marks a continuation line that defines additional data for SCALAR, SUBENTITY, and POINTER data types.

  "B" and "E" - Respectively marks a continuation line that defines the beginning and end of a group of attributes that are defined as a structure.

"A" - Marks a continuation line that defines the array bounds for the previously defined attribute. An attribute can be defined in terms of 1-n dimensional array. An example of the representation of a 3 dimensional array, of ADB information, is as follows.

```
Line 1 => DIAMETER        , 9,1, 3,2  4,  24
Line 2 => A  1,  2,1,  3,1,  4
```

This example defines a 3 dimensional array that is then defined as...

```
1)  REAL*4 DIAMETER(4,3,2) for "FORTRAN" and
2)  DIAMETER : ARRAY(.1..2.) OF
               ARRAY(.1..3.) OF
               ARRAY(.1..4.) OF SHORTREAL for "PASCAL"
```

An attribute can be defined in terms of 1-n dimensional array. An example of the representation of a (3) three dimensional array of ADB information is as follows:

```
     DIAMETER   ,9,  1,  3,  2,  4,  24
A      1,   2, 1,   -3,  1,  4
```

The location of data in storage for an ARRAY attribute is identical to Pascal stores arrays. For multidimensional arrays data is stored so that the Depth index is exhausted first, the Col index is exhausted second, and the Row index is exhausted last.

The REAL type data corresponding to this DIAMETER attribute would reside in the ADB as shown below.

	ADB displacement	Pascal Index	Fortran Index
	24	(1,1,1)	(1,1,1)
	28	(1,1,2)	(2,1,1)
	32	(1,1,3)	(3,1,1)
	36	(1,1,4)	(4,1,1)
	40	(1,2,1)	(1,2,1)
	44	(1,2,2)	(2,2,1)
	48	(1,2,3)	(3,2,1)
	52	(1,2,4)	(4,2,1)
	56	(1,3,1)	(1,3,1)
	.	.	.
	.	.	.
(col,dep)	68	(1,3,4)	(4,3,1)
	72	(2,1,1)	(1,1,2)
	76	(2,1,2)	(2,1,2)
	80	(2,1,3)	(3,1,2)
	84	(2,1,4)	(4,1,2)
	88	(2,2,1)	(1,2,2)

```
                    .              .              .
                    .              .              .
                    .              .              .
                    .              .              .
(row,col,dep)  120              (2,3,4)        (4,3,2)
```

Arrays for attributes that reside in the CL are stored in ARRAY entities. The example below was taken from the definition of the RB_SPLINE entity.

```
Line 1 => CONTROL          ,14,2, 1,7,  4,   1
Line 2 => X 2, 4000, 3001
Line 3 => A  2, 30
```

```
(1)    (2)          (3)(4)(5)(6)(7) (8)
 A, DIAMETER        , 9,1, 0,2,  4,  24
```

---

(2) ATTRIBUTE NAME field (16 character alpha)

The name of the attribute entity.

(3) CONCEPTUAL SCHEMA ORDER field (2 digit numeric)

- This field indicates the order of this attribute in the formal specification of this entity. It is this published conceptual schema order (CS order) that the entity attributes will follow in the Exchange Format file.

(1) (2)	(3)(4)(5)(6)(7) (8)
A, DIAMETER	, 9,1, 0,2, 4, 24

(4) MINIMUM OCCURRENCES field (1 digit numeric)

- An integer number signifying the minimum amount of data that can be stored for this attribute in increments of size.

  A zero (0) for this field implies that this attribute is optional. The type code (1-6) signifies that this attribute is ADB type data, zeros (0) or blank ( ) may be stored. The type code (7 or 8) signifies that this is CL type data, the entity reference in the CL may point to a NIL entity.

(5) ARRAY DIMENSION field (2 digit numeric) -

An integer number signifying the dimension of the array of ADB or CL data. Zero (0) implies only one instance of data.

(6) TYPE CODE field (integer number between 1 and 9)

- The TYPE CODE field signifies the type of data this attribute contains and what part of the entity it resides in, either the ADB or CL. The types are;

  1=> INTEGER
  2=> REAL
  3=> CHARACTER
  4=> LOGICAL
  5=> SCALAR
  6=> SET
  7=> POINTER
  8=> SUBENTITY
  9=> STRUCTURE

- Data types 1 through 6 reside in the ADB of the entity.
  Data types 7 and 8 reside in the CL of the entity.
  Data type 9 is a special type that gives location information about the attributes it encompasses.

(6) TYPE CODE field (Cont.)

- Below is a detailed explanation of each of the data types;

1 - INTEGER
A 1, 2, or 4 byte integer.
Resides on a single byte boundary, double byte boundary or a full word boundary. The example below was taken from the definition of the entity IMPL_B_HOLE.

```
KIND            ,1,  1,  1,  1,1,  4,    0
```

2 - REAL
A 4 or 8 byte real.
Resides on a full word boundary or a double word boundary. The example below was taken from the definition of the entity IMPL_B HOLE.

```
DIAMETER        ,1,  1,  1,  1,2,  4,    24
```

3 - CHARACTER
The character resides in 1 byte of storage and the SIZE field signifies how many characters are present. No boundary alignment. The example below was taken from the definition of the entity DETAIL_MODEL.

```
FSCM_CODE       ,1,  1,  1,  1,3,  5,    55
```

4 - LOGICAL
A 1 byte integer such that 0 = > FALSE and 1 = > TRUE.
No boundary alignment. The example below was taken from the definition of the entity POINT_VECTOR.

```
DISPLAYED       ,1,  1,  1,  1,4,  1,    24
```

5 - SCALAR
A 1 byte integer (1 - 256) that indexes to the scalar stored. The scalar names are enumerated latter in the attribute record.

(6) TYPE CODE field (Cont.)

See example below. No boundary alignment.

```
MATL_COND      ,1,  1,  1,  1,5,  1,  24,  4,N,M,L,S
  n             m   r   c   d t   s    a   n s s s s
  a             i   o   o   e y   i    d   u c c c c
  m             n   w   l   p p   z    b   m l l l l
  e                         t e   e        r r r r
                            h         d   s
                                      i   c n n n n
                                      s   l a a a a
                                      p   r m m m m
                                          s 1 2 3 4
```

If "M" was the scalar to be represented, a 2 would be at the 24th byte in the ADB.

6 - SET
Not incorporated.

7 - POINTER
Attributes of this type signify that a reference resides in the CL. The kinds of entities that this attribute can reference are enumerated latter in the attribute record. The example below was taken from the definition of the entity CIRCULAR_RUNOUT.

```
TOLERANCE_ENTITY,1,254,  1,  1,7,  4,    0, 1, 4,11000, 8004, 8002, 2000
PRIMARY          ,1,  1,  1,  1,7,  4,    0, 2, 1, 9020
CO_DATUM         ,0,  1,  1,  1,7,  4,    0, 3, 1, 9020
```

TOLERANCE_ENTITY is the first reference in the CL. Four different kinds of entities can be referenced.

8 - SUBENTITY
Attributes of this type are the same as attributes of type POINTER. The SUBENTITY code signifies that the type of entity pointed to was created as a result of the physical implementation of the Conceptual Schema. Entities of this type originally were structures in the definition of this entity.

9 - STRUCTURE
If in the Conceptual Schema definition of an entity an array of a structure is defined, and that structure contains only ADB type DATA or only CL type DATA then those attribute fields will be preceded and followed by attribute fields of type STRUCTURE in the data dictionary. The example below was taken from the definition of the entity PRS.

(6) TYPE CODE field (Cont.)

```
BEGIN_STRUCTURE ,0, 32,  1,  1,9,  0,  45
TVALUE          ,1,  1,  1,  1,2,  4,  48
UVALUE          ,1,  1,  1,  1,2,  4,  52
END_STRUCTURE   ,0, 32,  1,  1,9,  8, 304
                                   s  s  e  s
                                   t  i  n  t
                                   r  z  d  r
                                   u  e     u
                                   c  o     c
                                      f
```

The attribute type STRUCTURE supplies information on the location of the array of attributes that are bracketed within. The REAL type data corresponding to the TVALUE, and UVALUE attributes would reside in the ADB as shown below.

ADB Displacement	TVALUE Index	UVALUE Index
48	1	-
52	-	1
56	2	-
60	-	2
65	3	-
68	-	3
.	.	.
.	.	.
.	.	.
296	32	-
304	-	32

```
(1)    (2)          (3)(4)(5)(6)(7) (8)
 A, DIAMETER        , 9,1, 0,2,  4,  24
```

(7) SIZE field

- An integer number signifying the number of bytes that one instance of this attribute takes in storage.

(8) ADB/CL DISPLACEMENT field

- An integer number signifying (1) the starting location in the ADB of the entity for this attribute or (2) the location of this reference in the CL.

ADB - An integer number signifying the starting location in the ADB of the entity for this attribute. The example below was taken from the definition of the entity IMPL_B_HOLE.

```
#DIAMETER        , 9,1,  0,2,  4,  24
```

CL -An integer number signifying the location of this reference in the CL. The domain of this attribute type is enumerated on a continuation line following the attribute. The example below was taken from the definition of the entity CIRCULAR_RUNOUT.

```
#TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
```

ENTITY TYPE CODES

Miscellaneous Types

1100	Array

Subentity Types

1201	Segment
1202	Edges
1203	Implied Thru Hole
1204	Implied Blind Hole
1205	Datum Frame
1206	Mating Corner
1207	Implied Chamfer
1208	Implied Fillet
1209	Implied Inside Corner
1210	Bend
1211	Radius

Other

1300	Other Class
1301	Pick Token
1302	Text Line
1303	Text
1304	Group
1305	Display Properties
1306	Pick Point
1307	Nil Entity
1308	Menu Pick Item
1309	Erase

Geometry

2000	Geometry Class

Quasi Geometry

3000	Quasi Geometry Class
3001	Coordinate
3002	Vector
3003	MAT43
3004	MAT44
3005	Point Vector
3006	Iplane
3007	Sphere

Points

4000	Point Class
4001	Point
4002	Control Point

Curves

5000	Curve Class
5001	Circle
5002	Circular Arc
5003	Conic Arc
5004	Cubic
5005	Curve Segment
5006	Curve String
5007	Ellipse
5008	Line
5009	RB-Spline
5011	Knot

Surfaces

6000	Surface Class
6001	Cone
6002	Parametric-Bi-Cubic
6003	Cylinder Surface
6004	Thick Surface
6005	Plane
6006	Ruled Surface
6007	Surface of Rotation
6008	Surface of Translation
6009	RB-Spline-Surf

Solids

7000	Solids Class

Topology

8000	Topology Class
8001	Vertex
8002	Edge
8003	Loop
8004	Face
8005	Shell
8006	Object
8007	Super Face

Tolerance

9000	Tolerance Class
9001	Coordinate Tolerance Class
9002	Location
9003	Size
9004	Angle
9005	Geometric Tolerance Class
9006	Circular Runout
9007	Concentricity
9008	Cylindricity
9009	Flatness
9010	Line Profile
9011	Parallelism
9012	Perpendicularity
9013	Position
9014	Roundness
9015	Straightness
9016	Surface Profile
9017	Total Runout
9018	Angularity
9019	Other Tolerance Class
9020	Datum
9021	Feature of Size

Administrative Data

10000	Administrative Data
10001	Administration
10002	Approval
10003	Characteristic
10004	Detail Model
10005	Effectivity
10006	Material
10007	Note
10008	Specification
10009	Next Assy

Features

11000	Feature Class

Composite Features

12000	Composite Class
12001	Ply Detail
12002	Ply
12003	Ply Table
12004	Laminate
12005	Composite Flange
12006	Composite Hole
12007	Composite Transition
12008	Composite Rabbet
12009	Comp Flat Pat

Machine Features

13000	Machine Class
13001	Feature Edge
13002	Mach Chamfer
13003	Mach Cutout
13004	Mach Fillet
13005	Mach Flange
13006	Thru Hole
13007	Blind Hole
13008	Mach Inside Corner
13009	Mach Periphery
13010	Mach Pocket
13011	Mach Transition
13012	Mach Trim
13013	Mach Web

Sheet Metal Features

14000	Sheet Metal Class
14001	Sheet Metal Body
14002	Sheet Metal Flange
14003	Sheet Metal Web
14004	Sheet Metal Pocket
14005	Sheet Metal Notch
14006	Sheet Metal Joggle
14007	Sheet Metal Crimp
14008	Sheet Metal Flat Pattern
14009	Sheet Metal Cutout
14010	Sheet Metal Flat Hole
14011	Sheet Metal Flat Web
14012	Sheet Metal Flat Flange
14013	Sheet Metal Flat Notch
14014	Sheet Metal Bend

Turned Features

15000	Turned Class
15001	Turn End
15002	Turn Groove
15003	Turn Open Diameter
15004	Turn Recessed Diameter
15005	Turn Relief
15006	Turn Taper
15007	Turn Transition
15008	Turn Thread
15009	Turn Undercut
15010	Turn Open Face
15011	Turn Recessed Face
15012	Turn Semiopen Face
15013	Turn Semiopen Diameter
15014	Turn Axis
15015	Gage Point
15016	Turned Chamfer
15017	Turned Corn Rnd
15018	Turned Fillet
15019	Turned Profile
15020	Edge Break

```
********** #ALL     MEMBER **********
1100 ,ARRAY_ENTITY
1201 ,SEGMENT
1202 ,EDGES
1203 ,IMPL_T_HOLE
1204 ,IMPL_B_HOLE
1205 ,DATUM_FRAME
1206 ,MATING_CORNER
1207 ,IMPL_CHAMFER
1208 ,IMPL_FILLET
1209 ,IMPL_IN_CORNER
1210 ,BEND
1211 ,RADIUS
1300 ,OTHER_CLASS
1301 ,PICK_TOKEN
1302 ,TEXT_LINE
1303 ,TEXT
1304 ,GROUP
1305 ,DISP_PROP
1306 ,PICK_PT
1307 ,NIL_ENTITY
1308 ,MENU_PICK_ITEM
1309 ,ERASE
2000 ,GEOMETRY_CLASS
3000 ,QUASI_GEOM_CLASS
3001 ,COORDINATE
3002 ,VECTOR
3003 ,MAT43
3004 ,MAT44
3005 ,POINT_VECTOR
3006 ,IPLANE
3007 ,SPHERE
4000 ,POINT_CLASS
4001 ,POINT
4002 ,CONTROL_POINT
5000 ,CURVE_CLASS
5001 ,CIRCLE
5002 ,CIRCULAR_ARC
5003 ,CONIC_ARC
5004 ,CUBIC
5005 ,CURVE_SEGMENT
5006 ,CURVE_STRING
5007 ,ELLIPSE
5008 ,LINE
5009 ,RB_SPLINE
5011 ,KNOT
```

```
6000 ,SURFACE_CLASS
6001 ,CONE
6002 ,PARM_BI_CUBIC
6003 ,CYLINDER_SURFACE
6004 ,THICK_SURFACE
6005 ,PLANE
6006 ,RULED_SURFACE
6007 ,SURF_OF_ROTATION
6008 ,SURF_OF_TRANS
6009 ,RB_SPLINE_SURF
7000 ,SOLIDS_CLASS
8000 ,TOPOLOGY_CLASS
8001 ,VERTEX
8002 ,EDGE
8003 ,LOOP
8004 ,FACE
8005 ,SHELL
8006 ,OBJECT
8007 ,SUPER_FACE
9000 ,TOLERANCE_CLASS
9001 ,COORD_TOL_CLASS
9002 ,LOCATION
9003 ,SIZE
9004 ,ANGLE
9005 ,GEOM_TOL_CLASS
9006 ,CIRCULAR_RUNOUT
9007 ,CONCENTRICITY
9008 ,CYLINDRICITY
9009 ,FLATNESS
9010 ,LINE_PROFILE
9011 ,PARALLELISM
9012 ,PERPENDICULARITY
9013 ,POSITION
9014 ,ROUNDNESS
9015 ,STRAIGHTNESS
9016 ,SURFACE_PROFILE
9017 ,TOTAL_RUNOUT
9018 ,ANGULARITY
9019 ,OTHER_TOL_CLASS
9020 ,DATUM
9021 ,FEATURE_OF_SIZE
10000,ADMIN_DATA_CLASS
10001,ADMINISTRATION
10002,APPROVAL
10003,CHARACTERISTIC
10004,DETAIL_MODEL
10005,EFFECTIVITY
10006,MATERIAL
10007,NOTE
10008,SPECIFICATION
10009,NEXT_ASSEMBLY
```

```
11000,FEATURE_CLASS
12000,COMPOSITE_CLASS
12001,PLY_DETAIL
12002,PLY
12003,PLY_TABLE
12004,LAMINATE
12005,COMP_FLANGE
12006,COMP_HOLE
12007,COMP_TRANSITION
12008,COMP_RABBET
12009,COMP_FLAT_PAT
13000,MACHINE_CLASS
13001,FEATURE_EDGE
13002,MACH_CHAMFER
13003,MACH_CUTOUT
13004,MACH_FILLET
13005,MACH_FLANGE
13006,THRU_HOLE
13007,BLIND_HOLE
13008,MACH_INSIDE_CORN
13009,MACH_PERIPHERY
13010,MACH_POCKET
13011,MACH_TRANSITION
13012,MACH_TRIM
13013,MACH_WEB
14000,S_M_CLASS
14001,S_M_BODY
14002,S_M_FLANGE
14003,S_M_WEB
14004,S_M_POCKET
14005,S_M_NOTCH
14006,S_M_JOGGLE
14007,S_M_CRIMP
14008,S_M_FLAT_PATTERN
14009,S_M_CUTOUT
14010,S_M_FLAT_HOLE
14011,S_M_FLAT_WEB
14012,S_M_FLAT_FLANGE
14013,S_M_FLAT_NOTCH
14014,S_M_BEND
15000,TURNED_CLASS
15001,TRN_END
15002,TRN_GROOVE
15003,TRN_OPEN_DIA
15004,TRN_REC_DIA
15005,TRN_RELIEF
15006,TRN_TAPER
15007,TRN_TRANS
15008,TRN_THREAD
15009,TRN_UNDERCUT
```

```
15010,TRN_OPEN_FACE
15011,TRN_REC_FACE
15012,TRN_S_O_FACE
15013,TRN_S_O_DIA
15014,TRN_AXIS
15015,GAGE_POINT
15016,TURNED_CHAMFER
15017,TURNED_CORN_RND
15018,TURNED_FILLET
15019,TURNED_PROFILE
15020,EDGE_BREAK
************************************************************
```

```
********************* #CLASS   MEMBER *********************
1300 ,OTHER_CLASS
2000 ,GEOMETRY_CLASS
3000 ,QUASI_GEOM_CLASS
4000 ,POINT_CLASS
5000 ,CURVE_CLASS
6000 ,SURFACE_CLASS
8000 ,TOPOLOGY_CLASS
9000 ,TOLERANCE_CLASS
9001 ,COORD_TOL_CLASS
9019 ,OTHER_TOL_CLASS
10000,ADMIN_DATA_CLASS
11000,FEATURE_CLASS
12000,COMPOSITE_CLASS
13000,MACHINE_CLASS
14000,S_M_CLASS
15000,TURNED_CLASS
************************************************************

******************** #1100    MEMBER ************************
 ARRAY_ENTITY     , 1100, 7,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 CL_ENTITIES      , 7,1, 1,7,  4,   1
X 9, 2000, 3000, 4000, 5000, 6000, 8000, 9000,10000,11000
A  1,254
************************************************************

******************** #1201    MEMBER ************************
 SEGMENT          , 1201, 9,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 PRM              , 8,1, 0,2,  8,  24
 REV              , 9,1, 0,4,  1,  32
 CRV              , 7,1, 0,7,  4,   1
X 7, 5002, 5003, 5004, 5005, 5006, 5008, 5009
************************************************************
```

```
********** #1202    MEMBER **********
 EDGES           , 1202, 8,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 REVERSE         , 8,1, 0,4,  1,  24
 EDGREF          , 7,1, 0,7,  4,   1
X 1, 8002
****************************************************************

********** #1203    MEMBER **********
 IMPL_T_HOLE     , 1203,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 DIAMETER        , 9,1, 0,2,  4,  24
 LOCATE          , 7,1, 0,7,  4,   1
X 1, 4000
 AXIS            , 8,1, 0,7,  4,   2
X 1, 3002
 ENTRY_A         ,10,1, 1,7,  4,   3
X 1, 8004
A  1,254
 ENTRY_B         ,11,1, 1,7,  4,   4
X 1, 8004
A  1,254
****************************************************************
```

```
********************* #1204    MEMBER **********************
 IMPL_B_HOLE      , 1204,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 DIAMETER         , 9,1, 0,2,  4,  24
 PARM             ,12,1, 0,2,  4,  28
 HOLE_TYPE        ,11,1, 0,5,  1,  32
X 3,CONE             ,BULL              ,BALL
 LOCATE           , 7,1, 0,7,  4,   1
X 1, 4000
 AXIS             , 8,1, 0,7,  4,   2
X 1, 3002
 BOTTOM           ,10,1, 0,7,  4,   3
X 1, 4000
 ENTRY            ,13,1, 1,7,  4,   4
X 1, 8004
A  1,254
************************************************************

********************* #1205    MEMBER **********************
 DATUM_FRAME      , 1205, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 MATL_COND        , 8,1, 0,5,  1,  24
X 4,N                ,M                 ,L                 ,S
 DATUM_REF        , 7,1, 0,7,  4,   1
X 1, 9020

************************************************************

********************* #1206    MEMBER **********************
 MATING_CORNER    , 1206, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 RADIUS           , 7,1, 0,2,  4,  24
 ENT_REF          , 8,1, 1,7,  4,   1
X 2, 8002, 8004
A  1,255
************************************************************
```

```
********************* #1207    MEMBER *********************
 IMPL_CHAMFER     , 1207, 9,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 ANGLE            , 7,1, 0,2,  4,  24
 SETBACK          , 8,1, 0,2,  4,  28
 C_EDGES          , 9,1, 0,7,  4,   1
X 1,13001
*****************************************************************

********************* #1208    MEMBER *********************
 IMPL_FILLET      , 1208, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 RADIUS           , 7,1, 0,2,  4,  24
 FIL_EDGE         , 8,1, 0,7,  4,   1
X 1,13001
*****************************************************************

********************* #1209    MEMBER *********************
 IMPL_IN_CORNER   , 1209, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 RADIUS           , 8,1, 0,2,  4,  24
 CORNER           , 7,1, 0,7,  4,   1
X 1,13001
*****************************************************************

********************* #1210    MEMBER *********************
 BEND             , 1210, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 ANGLE            , 8,1, 0,2,  4,  24
 LOCATE           , 7,1, 0,7,  4,   1
X 1, 4000
*****************************************************************
```

```
********** #1211    MEMBER **********
 RADIUS          , 1211, 8,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 RADVAL          , 8,1, 0,2,  4,  24
 LOCATION        , 7,1, 0,7,  4,   1
X 1, 4000
************************************************************

********** #1300    MEMBER **********
 OTHER_CLASS     , 1300, 8
  1301           , 1,0, 0,0,  0,   0
  1302           , 2,0, 0,0,  0,   0
  1303           , 3,0, 0,0,  0,   0
  1304           , 4,0, 0,0,  0,   0
  1305           , 5,0, 0,0,  0,   0
  1306           , 6,0, 0,0,  0,   0
  1307           , 7,0, 0,0,  0,   0
  1308           , 8,0, 0,0,  0,   0
************************************************************

********** #1301    MEMBER **********
 PICK_TOKEN      , 1301,10,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 LABL            , 7,1, 0,7,  4,   1
X 1, 1302
 LEND1           , 8,1, 0,7,  4,   2
X 1, 3001
 LEND2           , 9,1, 0,7,  4,   3
X 1, 3001
 ENTREF          ,10,1, 0,7,  4,   4
X11, 2000, 3000, 4000, 5000, 6000, 8000, 9000,12000,13000,14000,15000
************************************************************
```

```
********************* #1302    MEMBER *********************
 TEXT_LINE         , 1302, 8,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
 LINE_LEN          , 7,1, 0,1,  1,  24
 A_LINE            , 8,1, 0,3,132,  25
*************************************************************

********************* #1303    MEMBER *********************
 TEXT              , 1303, 7,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
 LINES_OF_TEXT     , 7,1, 1,7,  4,   1
X 1, 1302
A  1,254
*************************************************************

********************* #1304    MEMBER *********************
 GROUP             , 1304,11,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 MEMBERS           ,13,0, 1,7,  4,   1
X11, 2000, 3000, 4000, 5000, 6000, 8000, 9000,12000,13000,14000,15000
A  0,999
*************************************************************
```

```
********** #1305    MEMBER **********
 DISP_PROP        , 1305,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 DISPLAYED        , 7,1, 0,4,  1,  24
 RBG_LEVEL        , 8,3, 1,1,  1,  25
A  3,  3
 INTENSITY        , 9,1, 0,1,  1,  28
 SYMBOL           ,10,1, 0,1,  1,  29
 MEMBER           ,11,1, 0,7,  4,   1
X11, 3000, 4000, 5000, 6000, 8000, 9000,10000,12000,13000,14000,15000
************************************************************

********** #1306    MEMBER **********
 PICK_PT          , 1306, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 PKPT             , 7,3, 1,2,  4,  24
A  3,  3
 PKRF             , 8,1, 0,7,  4,   1
X11, 3000, 4000, 5000, 6000, 8000, 9000,10000,12000,13000,14000,15000
************************************************************

********** #1307    MEMBER **********
 NIL_ENTITY       , 1307, 6,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
************************************************************
```

```
********** #1308    MEMBER **********
 MENU_PICK_ITEM  , 1308, 8,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 SEQ_ID          , 8,1, 0,1,  4,  24
 MAS_KEY         , 7,1, 0,7,  4,   1
X 9, 2000, 3000, 4000, 5000, 6000, 8000, 9000,10000,11000
************************************************************

********** #1309    MEMBER **********
 ERASE           , 1309, 8,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 ERASE_LEVEL     , 7,1, 0,1,  1,  24
 ERASE_LIST      , 8,1, 1,7,  4,   1
X 9, 2000, 3000, 4000, 5000, 6000, 8000, 9000,10000,11000
A  1,254
************************************************************

********** #2000    MEMBER **********
 GEOMETRY_CLASS  , 2000, 4
  3000           , 1,0, 0,0,  0,   0
  4000           , 2,0, 0,0,  0,   0
  5000           , 3,0, 0,0,  0,   0
  6000           , 4,0, 0,0,  0,   0
************************************************************

********** #3000    MEMBER **********
 QUASI_GEOM_CLASS, 3000, 7
  3001           , 1,0, 0,0,  0,   0
  3002           , 2,0, 0,0,  0,   0
  3003           , 3,0, 0,0,  0,   0
  3004           , 4,0, 0,0,  0,   0
  3005           , 5,0, 0,0,  0,   0
  3006           , 6,0, 0,0,  0,   0
  3007           , 7,0, 0,0,  0,   0
************************************************************
```

```
********** #3001    MEMBER **********
 COORDINATE      , 3001, 9,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 X               , 7,1, 0,2,  8,  24
 Y               , 8,1, 0,2,  8,  32
 Z               , 9,1, 0,2,  8,  40
************************************************************

********** #3002    MEMBER **********
 VECTOR          , 3002,10,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 I               , 7,1, 0,2,  8,  24
 J               , 8,1, 0,2,  8,  32
 K               , 9,1, 0,2,  8,  40
 L               ,10,1, 0,2,  8,  48
************************************************************

********** #3003    MEMBER **********
 MAT43           , 3003, 7,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 MAT             , 7,1, 2,2,  8,  24
A  1,  4,  1,  3
************************************************************

********** #3004    MEMBER **********
 MAT44           , 3004, 7,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 MAT             , 7,4, 2,2,  8,  24
A  4,  4,  4,  4
************************************************************
```

```
********************* #3005    MEMBER *********************
 POINT_VECTOR     , 3005,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 POS              ,13,1, 0,7,  4,   1
X 1, 4001
 DIR              ,14,1, 0,7,  4,   2
X 1, 3002
************************************************************

********************* #3006    MEMBER *********************
 IPLANE           , 3006,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 ORG              ,13,1, 0,7,  4,   1
X 1, 3001
 DIR              ,14,1, 0,7,  4,   2
X 1, 3002
************************************************************
```

```
********** #3007    MEMBER **********
 SPHERE          , 3007,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 CENTER          ,13,1, 0,7,  4,   1
X 1, 3001
 RADVEC          ,14,1, 0,7,  4,   2
X 1, 3002
************************************************************

********** #4000    MEMBER **********
 POINT_CLASS     , 4000, 1
  4001           , 1,0, 0,0,  0,   0
************************************************************

********** #4001    MEMBER **********
 POINT           , 4001,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 X               ,13,1, 0,2,  8,  32
 Y               ,14,1, 0,2,  8,  40
 Z               ,15,1, 0,2,  8,  48
************************************************************
```

```
********************* #4002    MEMBER *********************
 CONTROL_POINT    , 4002, 8,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 WEIGHT           , 7,1, 0,2,  8,  24
 POINT_REF        , 8,1, 0,7,  4,   1 X 2, 4001, 3001
*************************************************************

********************* #5000    MEMBER *********************
 CURVE_CLASS      , 5000, 9
  5001            , 1,0, 0,0,  0,   0
  5002            , 2,0, 0,0,  0,   0
  5003            , 3,0, 0,0,  0,   0
  5004            , 4,0, 0,0,  0,   0
  5005            , 5,0, 0,0,  0,   0
  5006            , 6,0, 0,0,  0,   0
  5007            , 7,0, 0,0,  0,   0
  5008            , 8,0, 0,0,  0,   0
  5009            , 9,0, 0,0,  0,   0
*************************************************************

********************* #5001    MEMBER *********************
 CIRCLE           , 5001,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 PC               ,13,1, 0,7,  4,   1
X 1, 3001
 AXIS             ,14,1, 0,7,  4,   2
X 1, 3002
 PO               ,15,1, 0,7,  4,   3
X 1, 3001
*************************************************************
```

```
********************* #5002    MEMBER *********************
 CIRCULAR_ARC     , 5002,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
BEND_POINTS       ,13,1, 0,9,  0,   1
 P0               ,14,1, 0,7,  4,   1
X 1, 4000
 P1               ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS       ,16,1, 0,9,  8,   3
 PM               ,17,1, 0,7,  4,   3
X 1, 3001
************************************************************

********************* #5003    MEMBER *********************
 CONIC_ARC        , 5003,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
BEND_POINTS       ,13,1, 0,9,  0,   1
 P0               ,14,1, 0,7,  4,   1
X 1, 4000
 P1               ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS       ,16,1, 0,9,  8,   3
 PA               ,17,1, 0,7,  4,   3
X 1, 3001
 PB               ,18,1, 0,7,  4,   4
X 1, 3001
 PC               ,19,1, 0,7,  4,   5
X 1, 3001
************************************************************
```

```
********************* #5004    MEMBER *********************
 CUBIC             , 5004,14,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
BEND_POINTS        ,13,1, 0,9,  0,   1
 P0                ,14,1, 0,7,  4,   1
X 1, 4000
 P1                ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS        ,16,1, 0,9,  8,   3
 V0                ,17,1, 0,7,  4,   3
X 1, 3002
 V1                ,18,1, 0,7,  4,   4
X 1, 3002
************************************************************

********************* #5005    MEMBER *********************
 CURVE_SEGMENT     , 5005,13,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
BEND_POINTS        ,13,1, 0,9,  0,   1
 P0                ,14,1, 0,7,  4,   1
X 1, 4000
 P1                ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS        ,16,1, 0,9,  8,   3
 BASE              ,17,1, 0,7,  4,   3
X 1, 5000
************************************************************
```

```
********************* #5006    MEMBER *********************
 CURVE_STRING     , 5006,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
BEND_POINTS       ,13,1, 0,9,  0,   1
 P0               ,14,1, 0,7,  4,   1
X 1, 4000
 P1               ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS       ,16,1, 0,9,  8,   3
 SEGMENT          ,17,1, 1,8,  4,   3
X 1, 1201
A  1,1000
************************************************************

********************* #5007    MEMBER *********************
 ELLIPSE          , 5007,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 PC               ,13,1, 0,7,  4,   1
X 1, 3001
 MAJOR            ,14,1, 0,7,  4,   2
X 1, 3001
 MINOR            ,15,1, 0,7,  4,   3
X 1, 3001
************************************************************
```

```
********************* #5008    MEMBER *********************
 LINE              , 5008,12,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
BEND_POINTS        ,13,1, 0,9,  0,   1
 P0                ,14,1, 0,7,  4,   1
X 1, 4000
 P1                ,15,1, 0,7,  4,   2
X 1, 4000
EEND_POINTS        ,16,1, 0,9,  8,   3
************************************************************
```

```
********** #5009    MEMBER **********
 RB_SPLINE        , 5009,18,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 DOMAIN_S         ,13,1, 0,2,  8,  32
 DOMAIN_E         ,14,1, 0,2,  8,  40
 DEGREE           ,15,1, 0,1,  1,  48
 SPAN             ,16,1, 0,1,  2,  50
 P_C_TYPE         ,17,1, 0,5,  1,  52
X  6,LINE             ,ARC              ,CIRCLE           ,CONIC            ,
X  6,ELLIPSE          ,OTHER
 PERIODIC         ,18,1, 0,4,  1,  53
 UNIFORM          ,19,1, 0,4,  1,  54
 CONTROL          ,20,2, 1,7,  4,   1
X 3, 4001, 3001, 4002
A  2,254
 KNOTS            ,21,0, 1,7,  4,   2
X 1, 5011
A  0,254
************************************************************

********** #5011    MEMBER **********
 KNOT             , 5011, 7,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
KNOT              , 7,1, 1,2,  8,  24
A  0, 10
************************************************************
```

```
********** #6000    MEMBER **********
 SURFACE_CLASS    , 6000,10
  6001            , 1,0, 0,0,  0,   0
  6002            , 2,0, 0,0,  0,   0
  6003            , 3,0, 0,0,  0,   0
  6004            , 4,0, 0,0,  0,   0
  6005            , 5,0, 0,0,  0,   0
  6006            , 6,0, 0,0,  0,   0
  6007            , 7,0, 0,0,  0,   0
  6008            , 8,0, 0,0,  0,   0
  6009            , 9,0, 0,0,  0,   0
  6010            ,10,0, 0,0,  0,   0
************************************************************

********** #6001    MEMBER **********
 CONE             , 6001,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 BASE             ,13,1, 0,7,  4,   1
X 1, 5001
 APEX             ,14,1, 0,7,  4,   2
X 1, 4000
************************************************************
```

```
********** #6002    MEMBER **********
 PARM_BI_CUBIC    , 6002,18,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
BBORDER           ,13,1, 0,9,  0,   1
 UOCRV            ,14,1, 0,7,  4,   1
X 3, 5004, 5008, 5005
 U1CRV            ,15,1, 0,7,  4,   2
X 3, 5004, 5008, 5005
 VOCRV            ,16,1, 0,7,  4,   3
X 3, 5004, 5008, 5005
 V1CRV            ,17,1, 0,7,  4,   4
X 3, 5004, 5008, 5005
EBORDER           ,18,1, 0,9, 16,   5
BTWIST            ,19,1, 0,9,  0,   5
 NUOVO            ,20,1, 0,7,  4,   5
X 1, 3002
 NUOV1            ,21,1, 0,7,  4,   6
X 1, 3002
 NU1VO            ,22,1, 0,7,  4,   7
X 1, 3002
 NU1V1            ,23,1, 0,7,  4,   8
X 1, 3002
ETWIST            ,24,1, 0,9, 16,   9
************************************************************
```

```
********** #6003    MEMBER **********
 CYLINDER_SURFACE, 6003,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 BASE             ,13,1, 0,7,  4,   1
X 1, 5001
 TOP              ,14,1, 0,7,  4,   2
X 1, 5001
************************************************************

********** #6004    MEMBER **********
 THICK_SURFACE    , 6004,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 OFFSET           ,14,1, 0,2,  8,  32
 BASE             ,13,1, 0,7,  4,   1
X 1, 6000
************************************************************
```

```
********************* #6005    MEMBER *********************
 PLANE             , 6005,13,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 ORG               ,13,1, 0,7,  4,   1
X 1, 3001
 VLEG              ,14,1, 0,7,  4,   2
X 1, 3002
 ULEG              ,15,1, 0,7,  4,   3
X 1, 3002
************************************************************

********************* #6006    MEMBER *********************
 RULED_SURFACE     , 6006,14,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 CU0               ,13,1, 0,7,  4,   1
X 3, 4000, 5000, 3002
 CU1               ,14,1, 0,7,  4,   2
X 3, 4000, 5000, 3002
 CV0               ,15,1, 0,7,  4,   3
X 3, 5008, 5004, 5006
 CV1               ,16,1, 0,7,  4,   4
X 3, 5008, 5004, 5006
************************************************************
```

```
********** #6007    MEMBER **********
 SURF_OF_ROTATION, 6007,14,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 UOCRV           ,13,1, 0,7,  4,   1
X 1, 5000
 U1CRV           ,14,1, 0,7,  4,   2
X 1, 5000
 VOCRV           ,15,1, 0,7,  4,   3
X 1, 5002
 V1CRV           ,16,1, 0,7,  4,   4
X 1, 5002
************************************************************

********** #6008    MEMBER **********
 SURF_OF_TRANS   , 6008,14,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 UOCRV           ,13,1, 0,7,  4,   1
X 1, 5000
 U1CRV           ,14,1, 0,7,  4,   2
X 1, 5000
 VOCRV           ,15,1, 0,7,  4,   3
X 1, 5000
 V1CRV           ,16,1, 0,7,  4,   4
X 1, 5000
************************************************************
```

```
********************* #6009    MEMBER *********************
 RB_SPLINE_SURF   , 6009,21,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
 EDISPLAY         ,12,1, 0,9,  6,  30
 DOMAIN_S_U       ,13,1, 0,2,  8,  32
 DOMAIN_E_U       ,14,1, 0,2,  8,  40
 DOMAIN_S_V       ,15,1, 0,2,  8,  48
 DOMAIN_E_V       ,16,1, 0,2,  8,  56
 DEGREE_U         ,17,1, 0,1,  1,  64
 DEGREE_V         ,18,1, 0,1,  1,  65
 SPANS_U          ,19,1, 0,1,  2,  66
 SPANS_V          ,20,1, 0,1,  2,  68
 P_S_TYPE         ,21,1, 0,5,  1,  70
X  6,PLANE            ,CONEX              ,CYLINDER         ,TORUS            ,
X  6,SPHERE           ,OTHERX
 NORMAL           ,22,1, 0,4,  1,  71
 PERIODIC_U       ,23,1, 0,4,  1,  72
 PERIODIC_V       ,24,1, 0,4,  1,  73
 UNIFORM_U        ,25,1, 0,4,  1,  74
 UNIFORM_V        ,26,1, 0,4,  1,  75
 CONTROL          ,27,2, 1,7,  4,   1
X 3, 4001, 3001, 4002
A  2,254
 KNOTS_U          ,28,0, 1,7,  4,   2
X 1, 5011
A  0,254
 KNOTS_V          ,29,0, 1,7,  4,   3
X 1, 5011
A  0,254
************************************************************
```

```
********** #7000    MEMBER **********
 SOLIDS_CLASS     , 7000, 7
  8001            , 1,0, 0,0,  0,   0
  8002            , 2,0, 0,0,  0,   0
  8003            , 3,0, 0,0,  0,   0
  8004            , 4,0, 0,0,  0,   0
  8005            , 5,0, 0,0,  0,   0
  8006            , 6,0, 0,0,  0,   0
  8007            , 7,0, 0,0,  0,   0
************************************************************
```

```
******************** #8000    MEMBER *********************
 TOPOLOGY_CLASS  , 8000, 7
  8001           , 1,0, 0,0,  0,   0
  8002           , 2,0, 0,0,  0,   0
  8003           , 3,0, 0,0,  0,   0
  8004           , 4,0, 0,0,  0,   0
  8005           , 5,0, 0,0,  0,   0
  8006           , 6,0, 0,0,  0,   0
  8007           , 7,0, 0,0,  0,   0
************************************************************

******************** #8001    MEMBER *********************
 VERTEX          , 8001,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PTREF           ,13,1, 0,7,  4,   1
X 1, 4000
************************************************************

******************** #8002    MEMBER *********************
 EDGE            , 8002,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 CRVREF          ,13,1, 0,7,  4,   1
X 1, 5000
 VERT0           ,14,1, 0,7,  4,   2
X 1, 8001
 VERT1           ,15,1, 0,7,  4,   3
X 1, 8001
************************************************************
```

```
********************* #8003    MEMBER *********************
 LOOP              , 8003,11,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 EDGES             ,13,2, 1,8,  4,   1
X 1, 1202
A  2,254
************************************************************

********************* #8004    MEMBER *********************
 FACE              , 8004,14,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 REVERSE           ,14,1, 0,4,  1,  30
 SRFREF            ,13,1, 0,7,  4,   1
X 2, 6000, 3006
 PERIPH            ,15,1, 0,7,  4,   2
X 1, 8003
 CUTOUT            ,16,0, 1,7,  4,   3
X 1, 8003
A  0,254
************************************************************
```

```
********************* #8005    MEMBER *********************
 SHELL             , 8005,11,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 FACREF            ,13,2, 1,7,  4,   1
X 1, 8004
A  2,254
*************************************************************

********************* #8006    MEMBER *********************
 OBJECT            , 8006,12,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PERIPH            ,13,1, 0,7,  4,   1
X 1, 8005
 VOIDS             ,14,0, 1,7,  4,   2
X 1, 8005
A  0,254
*************************************************************
```

```
********** #8007    MEMBER **********
 SUPER_FACE       , 8007,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 FACES            ,13,2, 1,7,  4,   1
X 1, 8004
A  2,254
 PERIPH           ,14,1, 0,7,  4,   2
X 1, 8003
 CUTOUT           ,15,0, 1,7,  4,   3
X 1, 8003
A  0,254
**************************************************************

********** #9000    MEMBER **********
 TOLERANCE_CLASS , 9000, 3
  9001            , 1,0, 0,0,  0,   0
  9005            , 2,0, 0,0,  0,   0
  9019            , 3,0, 0,0,  0,   0
**************************************************************

********** #9001    MEMBER **********
 COORD_TOL_CLASS , 9001, 3
  9002            , 1,0, 0,0,  0,   0
  9003            , 2,0, 0,0,  0,   0
  9004            , 3,0, 0,0,  0,   0
**************************************************************
```

```
******************** #9002    MEMBER *********************
 LOCATION          , 9002,16,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PLUS_TOL          ,15,1, 0,2,  4,  32
 MINUS_TOL         ,16,1, 0,2,  4,  36
 BASIC             ,14,1, 0,4,  1,  40
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 5,11000, 8004, 8002, 8001, 2000
A  1,254
 PATH              ,17,0, 0,7,  4,   2
X 2, 3002, 5000
 ORIGIN            ,18,1, 0,7,  4,   3
X 5,15015, 9020, 8004, 8002, 8001
************************************************************

******************** #9003    MEMBER *********************
 SIZE              , 9003,13,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PLUS_TOL          ,14,1, 0,2,  4,  32
 MINUS_TOL         ,15,1, 0,2,  4,  36
 TOLERANCE_ENTITY,13,1, 0,7,  4,   1
X 6,15015,13007,13006, 9021, 8004, 8002
************************************************************
```

```
********** #9004    MEMBER **********
 ANGLE            , 9004,16,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 PLUS_TOL         ,15,1, 0,2,  4,  32
 MINUS_TOL        ,16,1, 0,2,  4,  36
 BASIC            ,14,1, 0,4,  1,  40
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 8,12006,13006,13007,14010, 9021, 8004, 8002, 2000
A  1,254
 PATH             ,17,0, 0,7,  4,   2
X 1, 6005
 ORIGIN           ,18,1, 0,7,  4,   3
X 3, 9020, 8004, 8002
************************************************************

********** #9005    MEMBER **********
 GEOM_TOL_CLASS   , 9005,13
  9006            , 1,0, 0,0,  0,   0
  9007            , 2,0, 0,0,  0,   0
  9008            , 3,0, 0,0,  0,   0
  9009            , 4,0, 0,0,  0,   0
  9010            , 5,0, 0,0,  0,   0
  9011            , 6,0, 0,0,  0,   0
  9012            , 7,0, 0,0,  0,   0
  9013            , 8,0, 0,0,  0,   0
  9014            , 9,0, 0,0,  0,   0
  9015            ,10,0, 0,0,  0,   0
  9016            ,11,0, 0,0,  0,   0
  9017            ,12,0, 0,0,  0,   0
  9018            ,13,0, 0,0,  0,   0
************************************************************
```

```
********************* #9006    MEMBER *********************
 CIRCULAR_RUNOUT , 9006,14,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 TOLERANCE       ,14,1, 0,2,  4,  32
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
 PRIMARY         ,15,1, 0,7,  4,   2
X 1, 9020
 CO_DATUM        ,16,0, 0,7,  4,   3
X 1, 9020
************************************************************

********************* #9007    MEMBER *********************
 CONCENTRICITY   , 9007,14,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERS ON         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 TOLERANCE       ,14,1, 0,2,  4,  32
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 3,11000, 8004, 2000
A  1,254
 PRIMARY         ,15,1, 0,7,  4,   2
X 1, 9020
 CO_DATUM        ,16,0, 0,7,  4,   3
X 1, 9020
************************************************************
```

```
********************* #9008    MEMBER *********************
 CYLINDRICITY     , 9008,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 3,11000, 8004, 2000
A  1,254
*************************************************************

********************* #9009    MEMBER *********************
 FLATNESS         , 9009,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 MATL_COND        ,15,1, 0,5,  1,  36
X 4,N               ,M                ,L               ,S
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 2,11000, 8004
A  1,254
*************************************************************
```

```
********************* #9010    MEMBER *********************
 LINE_PROFILE     , 9010,16,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 APPLICATION      ,18,1, 0,5,  1,  36
X 3,BILATERAL       ,INSIDE          ,OUTSIDE
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
 PRIMARY          ,15,0, 0,8,  4,   2
X 1, 1205
 SECONDARY        ,16,0, 0,8,  4,   3
X 1, 1205
 TERTIARY         ,17,0, 0,8,  4,   4
X 1, 1205
************************************************************

********************* #9011    MEMBER *********************
 PARALLELISM      , 9011,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,15,1, 0,2,  4,  32
 CYLIN_TOL_ZONE   ,14,1, 0,4,  1,  36
 MATL_COND        ,16,1, 0,5,  1,  37
X 4,N               ,M               ,L               ,S
```

```
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
 PRIMARY         ,17,1, 0,8,  4,   2
X 1, 1205
*****************************************************************

******************** #9012    MEMBER *********************
 PERPENDICULARITY, 9012,18,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,15,1, 0,2,  4,  32
 CYLIN_TOL_ZONE   ,14,1, 0,4,  1,  36
 MATL_COND        ,16,1, 0,5,  1,  37
X 4,N               ,M                ,L                ,S
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
 PRIMARY         ,17,1, 0,8,  4,   2
X 1, 1205
 SECONDARY       ,18,0, 0,8,  4,   3
X 1, 1205
 TERTIARY        ,19,0, 0,8,  4,   4
X 1, 1205
 PROJ_TOL_ZONE   ,20,0, 0,7,  4,   5
X 1, 3002
*****************************************************************
```

```
********************* #9013    MEMBER *********************
 POSITION          , 9013,18,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 TOLERANCE         ,15,1, 0,2,  4,  32
 CYLIN_TOL_ZONE    ,14,1, 0,4,  1,  36
 MATL_COND         ,16,1, 0,5,  1,  37
X 4,N                ,M                  ,L                  ,S
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 5,11000, 8004, 8002, 8001, 2000
A  1,254
 PRIMARY           ,17,1, 0,8,  4,   2
X 1, 1205
 SECONDARY         ,18,1, 0,8,  4,   3
X 1, 1205
 TERTIARY          ,19,0, 0,8,  4,   4
X 1, 1205
 PROJ_TOL_ZONE     ,20,0, 0,7,  4,   5
X 1, 3002
************************************************************

********************* #9014    MEMBER *********************
 ROUNDNESS         , 9014,12,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 TOLERANCE         ,14,1, 0,2,  4,  32
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
************************************************************
```

```
********************* #9015    MEMBER *********************
 STRAIGHTNESS     , 9015,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,15,1, 0,2,  4,  32
 CYLIN_TOL_ZONE   ,14,1, 0,4,  1,  36
 MATL_COND        ,16,1, 0,5,  1,  37
X 4,N               ,M                ,L                ,S
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
A  1,254
 DIRECTION        ,17,0, 0,7,  4,   2
X 1, 3002
************************************************************

********************* #9016    MEMBER *********************
 SURFACE_PROFILE , 9016,16,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 APPLICATION      ,18,1, 0,5,  1,  36
X 3,BILATERAL       ,INSIDE           ,OUTSIDE
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 3,11000, 8004, 2000
A  1,254
 PRIMARY          ,15,0, 0,8,  4,   2
X 1, 1205
 TERTIARY         ,16,0, 0,8,  4,   3
```

```
X 1, 1205
 SECONDARY        ,17,0, 0,8,  4,   4
X 1, 1205
**************************************************************

********************* #9017    MEMBER *********************
 TOTAL_RUNOUT     , 9017,14,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 3,11000, 8004, 2000
A  1,254
 PRIMARY          ,15,1, 0,7,  4,   2
X 1, 9020
 CO_DATUM         ,16,0, 0,7,  4,   3
X 1, 9020
**************************************************************

********************* #9018    MEMBER *********************
 ANGULARITY       , 9018,16,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TOLERANCE        ,14,1, 0,2,  4,  32
 MATL_COND        ,15,1, 0,5,  1,  36
X 4,N               ,M                ,L               ,S
 TOLERANCE_ENTITY,13,1, 1,7,  4,   1
X 4,11000, 8004, 8002, 2000
```

```
A  1,254
 PRIMARY          ,16,1, 0,8,  4,   2
X 1, 1205
 TERTIARY         ,17,0, 0,8,  4,   3
X 1, 1205
 SECONDARY        ,18,0, 0,8,  4,   4
X 1, 1205
************************************************************

******************** #9019    MEMBER *********************
 OTHER_TOL_CLASS , 9019, 2
  9020            , 1,0, 0,0,  0,   0
  9021            , 2,0, 0,0,  0,   0
************************************************************

******************** #9020    MEMBER *********************
 DATUM            , 9020,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 NAME             ,13,1, 0,3,  2,  30
 DATUM_ENTITY     ,14,1, 0,7,  4,   1
X 3,11000, 8000, 2000
************************************************************

******************** #9021    MEMBER *********************
 FEATURE_OF_SIZE , 9021,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 SIDE_1           ,13,1, 0,7,  4,   1
```

```
X 1, 8004
 SIDE_2          ,14,1, 0,7,  4,   2
X 1, 8004
*************************************************************

******************** #10000   MEMBER ********************
 ADMIN_DATA_CLASS,10000, 9
 10001           , 1,0, 0,0,  0,   0
 10002           , 2,0, 0,0,  0,   0
 10003           , 3,0, 0,0,  0,   0
 10004           , 4,0, 0,0,  0,   0
 10005           , 5,0, 0,0,  0,   0
 10006           , 6,0, 0,0,  0,   0
 10007           , 7,0, 0,0,  0,   0
 10008           , 8,0, 0,0,  0,   0
 10009           , 9,0, 0,0,  0,   0
*************************************************************

******************** #10001   MEMBER ********************
 ADMINISTRATION  ,10001,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 COST_CODE       , 7,1, 0,3, 10,  24
 CONTRACT_NO     , 8,1, 0,3, 15,  34
 SECURITY_CODE   , 9,1, 0,3,  1,  49
BCREATING_DWG    ,10,1, 0,9,  0,  50
 PART_NUMBER     ,11,1, 0,3, 15,  50
 PART_VERSION    ,12,1, 0,3,  3,  65
 FSCM_CODE       ,13,1, 0,3,  5,  68
ECREATING_DWG    ,14,1, 0,9, 23,  73
 APPROVALS       ,15,0, 1,7,  4,   1
X 1,10002
A  0,254
*************************************************************

******************** #10002   MEMBER ********************
 APPROVAL        ,10002,10,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 APP_DATE        , 7,1, 0,3,  8,  24
 APP_BY          , 8,1, 0,3, 25,  32
 APP_FUNCTION    , 9,1, 0,3, 25,  57
 APP_DEPT        ,10,1, 0,3, 25,  82
*************************************************************
```

```
********** #10003   MEMBER **********
 CHARACTERISTIC  ,10003,22,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BWEIGHT_OF_PART  ,10,1, 0,9,  0,  24
BT_CALCULATED    ,12,1, 0,9,  0,  24
 XCG             ,14,1, 0,2,  8,  24
 YCG             ,15,1, 0,2,  8,  32
 ZCG             ,16,1, 0,2,  8,  40
 XK              ,17,1, 0,2,  8,  48
 YK              ,18,1, 0,2,  8,  56
 ZK              ,19,1, 0,2,  8,  64
 CALCULATED_WT   ,13,1, 0,2,  4,  72
ET_CALCULATED    ,20,1, 0,9, 52,  76
BTARGET          ,21,1, 0,9,  0,  76
 NOMINAL         ,22,1, 0,2,  4,  76
 LOW             ,23,1, 0,2,  4,  80
 HIGH            ,24,1, 0,2,  4,  84
 BASIS           ,25,1, 0,5,  1,  88
X 4,ACTUAL          ,TARGET          ,ESTIMATED       ,PROCURMENT
ETARGET          ,26,1, 0,9, 13,  89
 UNITS           ,11,1, 0,5,  1,  89
X 4,OUNCE           ,POUND           ,GRAM            ,KILOGRAM
EWEIGHT_OF_PART  ,27,1, 0,9, 66,  90
 CLASS_CODE      , 8,1, 0,3, 32,  90
 SPARES_CODE     , 9,1, 0,3,  2, 122
 IR_CODE         , 7,1, 0,3,  2, 124
 NOTES           ,28,0, 1,7,  4,   1
X 1,10007
A  0, 75
************************************************************

********** #10004   MEMBER **********
 DETAIL_MODEL    ,10004,27,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 DEFAULT_TOL     ,16,1, 0,2,  4,  24
 DEFAULT_ANG_TOL ,17,1, 0,2,  4,  28
 PART_TYPE       ,13,1, 0,3,  3,  32
 STATUS          ,14,1, 0,3,  1,  35
 UNITS_D_P       ,15,1, 0,5,  1,  36
X 2,INCH            ,MM
```

```
BPARTNO           , 7,1, 0,9,  0,  37
 PART_NUMBER      , 8,1, 0,3, 15,  37
 PART_VERSION     , 9,1, 0,3,  3,  52
 FSCM_CODE        ,10,1, 0,3,  5,  55
EPARTNO           ,11,1, 0,9, 23,  60
 PART_NAME        ,12,1, 0,3, 60,  60
BSAME_AS_EXCEPT   ,18,1, 0,9,  0, 120
 PART_NUMBER      ,19,1, 0,3, 15, 120
 PART_VERSION     ,20,1, 0,3,  3, 135
 FSCM_CODE        ,21,1, 0,3,  5, 138
ESAME_AS_EXCEPT   ,22,1, 0,9, 23, 143
 CHARACTERISTIC   ,23,1, 0,7,  4,   1
X 1,10003
 NEXT_ASSY        ,24,0, 1,7,  4,   2
X 1,10009
A  0,254
 ADMIN_INFO       ,25,1, 0,7,  4,   3
X 1,10001
 MATERIALS        ,26,0, 1,7,  4,   4
X 1,10006
A  0,254
 SPECIFICATIONS   ,27,0, 1,7,  4,   5
X 1,10008
A  0,254
 NOTES            ,28,0, 1,7,  4,   6
X 1,10007
A  0, 75
 PART_TOL         ,29,0, 1,7,  4,   7
X 1, 9000
A  0,254
 PART_FEATURE     ,30,0, 1,7,  4,   8
X 1,11000
A  0,254
 PART_TOPOLOGY    ,31,1, 0,7,  4,   9
X 1, 8006
*************************************************************

********************* #10005   MEMBER *********************
 EFFECTIVITY      ,10005,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
 QTY_END_ITEM     ,11,1, 0,1,  2,  24
 END_ITEM         , 8,1, 0,3, 16,  26
 FR_SERIAL        , 9,1, 0,3,  6,  42
 TO_SERIAL        ,10,1, 0,3,  6,  48
 PART_USAGE       , 7,1, 0,3,  2,  54
*************************************************************
```

```
********** #10006   MEMBER **********
 MATERIAL          ,10006,17,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
 ENVELOPE          ,10,3, 1,2,  4,  24
A  3,  3
 MATERIAL_NAME     , 8,1, 0,3, 15,  36
 MATERIAL_TITLE    , 9,1, 0,3, 30,  51
 MATERIAL_CODE     , 7,1, 0,3, 10,  81
BMAKE_FROM         ,11,1, 0,9,  0,  91
 PART_NUMBER       ,12,1, 0,3, 15,  91
 PART_VERSION      ,13,1, 0,3,  3, 106
 FSCM_CODE         ,14,1, 0,3,  5, 109
EMAKE_FROM         ,15,1, 0,9, 23, 114
 CONDITION_AFTER   ,16,1, 0,3, 10, 114
 STOCK_STRUCTURE   ,17,1, 0,3, 10, 124
 NOTES             ,18,0, 1,7,  4,   1
X 1,10007
A  0, 75
 SPECIFICATIONS    ,19,0, 1,7,  4,   2
X 1,10008
A  0,254
**************************************************************

********** #10007   MEMBER **********
 NOTE              ,10007, 9,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
 NOTE_NO           , 7,1, 0,1,  4,  24
 NOTE_TEXT         , 8,1, 1,7,  4,   1
X 1, 1302
A  1,254
 FLAGGED           , 9,0, 1,7,  4,   2
X 2,11000, 8000
A  0,254
**************************************************************
```

```
********** #10008   MEMBER **********
 SPECIFICATION   ,10008,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 SPEC_IDENT      , 7,1, 0,3, 15,  24
 SPEC_PARM       , 8,1, 0,3, 15,  39
 SPEC_TYPE       , 9,1, 0,3,  3,  54
 SPEC_NAME       ,10,1, 0,3, 25,  57
 NOTES           ,11,0, 1,7,  4,   1
X 1,10007
A  0, 75
************************************************************

********** #10009   MEMBER **********
 NEXT_ASSEMBLY   ,10009,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
 QTY_N_A         ,12,1, 0,1,  2,  24
BNEXT_ASSY       , 7,1, 0,9,  0,  26
 PART_NUMBER     , 8,1, 0,3, 15,  26
 PART_VERSION    , 9,1, 0,3,  3,  41
 FSCM_CODE       ,10,1, 0,3,  5,  44
ENEXT_ASSY       ,11,1, 0,9, 23,  49
 NOTES           ,13,0, 1,7,  4,   1
X 1,10007
A  0, 75
 EFFECTIVITY     ,14,0, 1,7,  4,   2
X 1,10005
A  0,254
************************************************************

********** #11000   MEMBER **********
 FEATURE_CLASS   ,11000, 4
 12000           , 1,0, 0,0,  0,   0
 13000           , 2,0, 0,0,  0,   0
 14000           , 3,0, 0,0,  0,   0
 15000           , 4,0, 0,0,  0,   0
************************************************************
```

```
********** #12000   MEMBER **********
 COMPOSITE_CLASS ,12000, 8
 12001           , 1,0, 0,0,  0,   0
 12002           , 2,0, 0,0,  0,   0
 12003           , 3,0, 0,0,  0,   0
 12004           , 4,0, 0,0,  0,   0
 12005           , 5,0, 0,0,  0,   0
 12006           , 6,0, 0,0,  0,   0
 12007           , 7,0, 0,0,  0,   0
 12008           , 8,0, 0,0,  0,   0
************************************************************

********** #12001   MEMBER **********
 PLY_DETAIL      ,12001,19,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 STACK_NO        ,19,1, 0,1,  4,  32
 CUT             ,18,1, 0,5,  1,  36
X 3,MANUAL          ,AUTO             ,NA
 DASH_NO         ,13,1, 0,3,  5,  37
 INVERTED        ,20,1, 0,5,  1,  42
X 3,YES             ,NO               ,NOT_APPLICABLE
BORIENTATION     ,14,1, 0,9,  0,   1
 COL_REF_PNT     ,15,1, 0,7,  4,   1
X 1, 3001
 FILAMENT_DIR    ,16,1, 0,7,  4,   2
X 1, 3002
EORIENTATION     ,17,1, 0,9,  8,   3
 SHAPE           ,21,1, 1,7,  4,   3
X 1, 8004
A  1,254
 FLAT_PAT        ,22,1, 1,7,  4,   4
X 1,12009
A  1,254
 PLY_MATERIAL    ,23,0, 0,7,  4,   5
X 1,10006
************************************************************
```

```
********** #12002   MEMBER **********
 PLY             ,12002,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PLY_NO          ,13,1, 0,3, 15,  30
 DETAIL          ,14,1, 1,7,  4,   1
X 1,12001
A  1,254
************************************************************

********** #12003   MEMBER **********
 PLY_TABLE       ,12003,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 LOCATION        ,13,1, 0,7,  4,   1
X 1, 3001
 ENTRY           ,14,1, 1,7,  4,   2
X 1,12001
A  1,254
************************************************************
```

```
********************* #12004   MEMBER *********************
 LAMINATE          ,12004,14,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PLYS              ,13,1, 1,7,  4,   1
X 1,12002
A  1,254
 PRIMARY           ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
 SECONDARY         ,15,1, 1,7,  4,   3
X 1, 8004
A  1,254
 PLY_TABLE_REF     ,16,1, 1,7,  4,   4
X 1,12003
A  1,254
************************************************************

********************* #12005   MEMBER *********************
 COMP_FLANGE       ,12005,13,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 P_FACE            ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 S_FACE            ,14,1, 2,7,  4,   2
X 1, 8004
A  1,254,  1,254
```

```
 T_FACE           ,15,1, 2,7,  4,   3
X 1, 8004
A  1,254,  1,254
*************************************************************

********************* #12006   MEMBER *********************
 COMP_HOLE        ,12006,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 IMPLICITE_C_HOLE,13,1, 0,8,  4,   1
X 1, 1203
BEXPLICITE_C_HOLE,14,0, 0,9,  0,   2
 WALL             ,15,1, 1,7,  4,   2
X 1, 8004
A  1,254
EEXPLICITE_C_HOLE,16,0, 0,9,  4,   3
*************************************************************

********************* #12007   MEMBER *********************
 COMP_TRANSITION ,12007,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TRAN_SURF        ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 PLY_DROPPED      ,14,1, 1,7,  4,   2
X 1,12001
A  1,254
*************************************************************
```

```
********** #12008   MEMBER **********
 COMP_RABBET      ,12008,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 MATING_CORNER    ,13,2, 1,8,  4,   1
X 1, 1206
A  2,  2
 BENT_PLYS        ,14,1, 1,7,  4,   2
X 1,12001
A  1,254
 CONT_PLYS        ,15,1, 1,7,  4,   3
X 1,12001
A  1,254
 STOPPED_PLYS     ,16,1, 1,7,  4,   4
X 1,12001
A  1,254
 RABBET_SURF      ,17,1, 1,7,  4,   5
X 1, 8004
A  1,254
************************************************************
```

```
********** #12009   MEMBER **********
 COMP_FLAT_PAT   ,12009,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
BORIENTATION     ,13,1, 0,9,  0,   1
 COL_REF_PNT     ,14,1, 0,7,  4,   1
X 1, 3001
 FILAMENT_DIR    ,15,1, 0,7,  4,   2
X 1, 3002
EORIENTATION     ,16,1, 0,9,  8,   3
 PERIPHERY       ,17,1, 0,7,  4,   3
X 1, 8004
************************************************************
```

```
********** #13000   MEMBER **********
 MACHINE_CLASS    ,13000,13
 13001            , 1,0, 0,0,  0,   0
 13002            , 2,0, 0,0,  0,   0
 13003            , 3,0, 0,0,  0,   0
 13004            , 4,0, 0,0,  0,   0
 13005            , 5,0, 0,0,  0,   0
 13006            , 6,0, 0,0,  0,   0
 13007            , 7,0, 0,0,  0,   0
 13008            , 8,0, 0,0,  0,   0
 13009            , 9,0, 0,0,  0,   0
 13010            ,10,0, 0,0,  0,   0
 13011            ,11,0, 0,0,  0,   0
 13012            ,12,0, 0,0,  0,   0
 13013            ,13,0, 0,0,  0,   0
************************************************************

********** #13001   MEMBER **********
 FEATURE_EDGE     ,13001,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 START            ,13,1, 0,7,  4,   1
X 2, 8001, 4000
 FINISH           ,14,1, 0,7,  4,   2
X 2, 8001, 4000
 EDGREF           ,15,1, 0,7,  4,   3
X 2, 8002, 5000
************************************************************
```

```
********** #13002   MEMBER **********
 MACH_CHAMFER     ,13002,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 P_FACE           ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 IMPLICITE_M_CHAM,14,0, 0,8,  4,   2
X 1, 1207
BEXPLICITE_M_CHAM,15,0, 0,9,  0,   3
 C_FACE           ,16,1, 1,7,  4,   3
X 1, 8004
A  1,254
EEXPLICITE_M_CHAM,17,0, 0,9,  4,   4
************************************************************

********** #13003   MEMBER **********
 MACH_CUTOUT      ,13003,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 WALL             ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
************************************************************
```

```
********************* #13004   MEMBER *********************
 MACH_FILLET      ,13004,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 IMPLICITE_M_FILT,13,0, 0,8,  4,   1
X 1, 1208
BEXPLICITE_M_FILT,14,0, 0,9,  0,   2
 FIL_FACE         ,15,0, 1,7,  4,   2
X 1, 8004
A  1,254
EEXPLICITE_M_FILT,16,0, 0,9,  4,   3
************************************************************

********************* #13005   MEMBER *********************
 MACH_FLANGE      ,13005,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 P_FACE           ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 S_FACE           ,14,1, 2,7,  4,   2
X 1, 8004
A  1,254,  1,254
 T_FACE           ,15,1, 2,7,  4,   3
X 1, 8004
A  1,254,  1,254
************************************************************
```

```
********************* #13006   MEMBER *********************
 THRU_HOLE        ,13006,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 IMPLICITE_T_HOLE,13,0, 0,8,  4,   1
X 1, 1203
BEXPLICITE_T_HOLE,14,0, 0,9,  0,   2
 WALL             ,15,0, 1,7,  4,   2
X 1, 8004
A  1,254
EEXPLICITE_T_HOLE,16,0, 0,9,  4,   3
************************************************************

********************* #13007   MEMBER *********************
 BLIND_HOLE       ,13007,13,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 IMPLICITE_B_HOLE,13,0, 0,8,  4,   1
X 1, 1204
BEXPLICITE_B_HOLE,14,0, 0,9,  0,   2
 WALL             ,15,0, 1,7,  4,   2
X 1, 8004
A  1,254
 BOTTOM           ,16,0, 1,7,  4,   3
X 1, 8004
A  1,254
EEXPLICITE_B_HOLE,17,0, 0,9,  8,   4
************************************************************
```

```
********** #13008   MEMBER **********
 MACH_INSIDE_CORN,13008,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 IMPLICITE_M_I_C ,13,0, 0,8,  4,   1
X 1, 1209
BEXPLICITE_M_I_C ,14,0, 0,9,  0,   2
 C_FACE          ,15,0, 1,7,  4,   2
X 1, 8004
A  1,254
EEXPLICITE_M_I_C ,16,0, 0,9,  4,   3
**************************************************************

********** #13009   MEMBER **********
 MACH_PERIPHERY  ,13009,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PERIPHERY       ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
**************************************************************
```

```
********************* #13010   MEMBER *********************
 MACH_POCKET      ,13010,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 WALL             ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 FLOOR            ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
************************************************************

********************* #13011   MEMBER *********************
 MACH_TRANSITION ,13011,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 T_FACE           ,13,1, 2,7,  4,   1
X 1, 8004
A  1,254,  1,254
************************************************************
```

```
********************* #13012   MEMBER *********************
 MACH_TRIM          ,13012,11,1
 KIND               , 1,1, 0,1,  4,   0
 LENGTH             , 2,1, 0,1,  4,   4
 SYSUSE             , 3,1, 0,1,  4,   8
 VERSION            , 4,1, 0,1,  4,  12
 SYS_IDENT          , 5,1, 0,1,  4,  16
 IDENT              , 6,1, 0,1,  4,  20
BDISPLAY            , 7,1, 0,9,  0,  24
 DISPLAYED          , 8,1, 0,4,  1,  24
 RBG_LEVEL          , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY          ,10,1, 0,1,  1,  28
 SYMBOL             ,11,1, 0,1,  1,  29
EDISPLAY            ,12,1, 0,9,  6,  30
 TRIM               ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
************************************************************

********************* #13013   MEMBER *********************
 MACH_WEB           ,13013,12,1
 KIND               , 1,1, 0,1,  4,   0
 LENGTH             , 2,1, 0,1,  4,   4
 SYSUSE             , 3,1, 0,1,  4,   8
 VERSION            , 4,1, 0,1,  4,  12
 SYS_IDENT          , 5,1, 0,1,  4,  16
 IDENT              , 6,1, 0,1,  4,  20
BDISPLAY            , 7,1, 0,9,  0,  24
 DISPLAYED          , 8,1, 0,4,  1,  24
 RBG_LEVEL          , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY          ,10,1, 0,1,  1,  28
 SYMBOL             ,11,1, 0,1,  1,  29
EDISPLAY            ,12,1, 0,9,  6,  30
 PRIMARY_FACE       ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 SECONDARY_FACE     ,14,1, 2,7,  4,   2
X 1, 8004
A  1,254,  1,254
************************************************************
```

```
********** #14000   MEMBER **********
 S_M_CLASS        ,14000,14
 14001            , 1,0, 0,0,  0,   0
 14002            , 2,0, 0,0,  0,   0
 14003            , 3,0, 0,0,  0,   0
 14004            , 4,0, 0,0,  0,   0
 14005            , 5,0, 0,0,  0,   0
 14006            , 6,0, 0,0,  0,   0
 14007            , 7,0, 0,0,  0,   0
 14008            , 8,0, 0,0,  0,   0
 14009            , 9,0, 0,0,  0,   0
 14010            ,10,0, 0,0,  0,   0
 14011            ,11,0, 0,0,  0,   0
 14012            ,12,0, 0,0,  0,   0
 14013            ,13,0, 0,0,  0,   0
 14014            ,14,0, 0,0,  0,   0
************************************************************

********** #14001   MEMBER **********
 S_M_BODY         ,14001,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 PRIMARY_FACE     ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 SECONDARY_FACE   ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
************************************************************
```

```
********************* #14002   MEMBER *********************
 S_M_FLANGE        ,14002,12,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PRIMARY_FACE      ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 SECONDARY_FACE    ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
************************************************************

********************* #14003   MEMBER *********************
 S_M_WEB           ,14003,12,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 PRIMARY_FACE      ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 SECONDARY_FACE    ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
************************************************************
```

```
******************** #14004   MEMBER *********************
 S_M_POCKET       ,14004,14,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 CM_SIDE          ,13,1, C,5,  1,  30
X 2,NEAR             ,FAR
 CM_PROCESS       ,14,1, 0,5,  1,  31
X 3,SCRIBE           ,SCRIBESEAL       ,NOT_AP
 WALL             ,15,1, 1,7,  4,   1
X 1, 8004
A  1,254
 FLOOR            ,16,1, 1,7,  4,   2
X 1, 8004
A  1,254
************************************************************

******************** #14005   MEMBER *********************
 S_M_NOTCH        ,14005,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 WALL             ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 CRIMP            ,14,0, 0,7,  4,   2
X 1,14007
************************************************************
```

```
********** #14006   MEMBER **********
 S_M_JOGGLE       ,14006,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 RADIUS           ,17,1, 0,2,  4,  32
 PRIMARY_FACE     ,13,1, 1,7,  4,   1
X 1, 8004
A  1,254
 SECONDARY_FACE   ,14,1, 1,7,  4,   2
X 1, 8004
A  1,254
 BEND1            ,15,1, 0,7,  4,   3
X 1,13001
 BEND2            ,16,1, 0,7,  4,   4
X 1,13001
************************************************************

********** #14007   MEMBER **********
 S_M_CRIMP        ,14007,15,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 BEND_RAD         ,13,1, 0,2,  4,  32
 CR_SETBACK       ,14,1, 0,2,  4,  36
 CR_ANGLE         ,15,1, 0,2,  4,  40
 CR_ORINTATION    ,16,1, 0,7,  4,   1
X 1, 3002
 FORM_LINE        ,17,1, 0,7,  4,   2
X 1,13001
************************************************************
```

```
********** #14008   MEMBER **********
 S_M_FLAT_PATTERN,14008,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 BASE            ,13,1, 0,7,  4,   1
X 1, 6005
 PERIPH          ,14,2, 1,7,  4,   2
X 1, 5000
A  2,254
************************************************************

********** #14009   MEMBER **********
 S_M_CUTOUT      ,14009,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 FLANGE_ANGLE    ,14,1, 0,2,  4,  32
 PERIPH          ,13,2, 1,7,  4,   1
X 1, 5000
A  2,254
 FLANGE_PERIPHERY,15,0, 1,7,  4,   2
X 1, 5000
A  0,254
************************************************************
```

```
********** #14010   MEMBER **********
 S_M_FLAT_HOLE   ,14010,15,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 DIAMETER        ,14,1, 0,2,  4,  32
 CM_SIDE         ,15,1, 0,5,  1,  36
X 3,NEAR_SIDE      ,FAR_SIDE       ,NOT_APLIC
 CM_PROCESS      ,16,1, 0,5,  1,  37
X 3,SCRIBE         ,SCRIBESEAL     ,NOT_AP
 HOLE_TYPE       ,17,1, 0,5,  1,  38
X 4,DRILLED        ,TOOLING        ,CUTOUT          ,CHEMMILL
 LOCATION        ,13,1, 0,7,  4,   1
X 1, 4000
************************************************************

********** #14011   MEMBER **********
 S_M_FLAT_WEB    ,14011,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PERIPH          ,13,2, 1,7,  4,   1
X 1, 5000
A  2,254
 BENDS           ,14,0, 1,7,  4,   2
X 1,14014
A  0,254
************************************************************
```

```
********** #14012   MEMBER **********
 S_M_FLAT_FLANGE ,14012,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PERIPH          ,13,2, 1,7,  4,   1
X 1, 5000
A  2,254
 BENDS           ,14,0, 1,7,  4,   2
X 1,14014
A  0,254
************************************************************

********** #14013   MEMBER **********
 S_M_FLAT_NOTCH  ,14013,12,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PERIPH          ,13,2, 1,7,  4,   1
X 1, 5000
A  2,254
 CRIMP           ,14,0, 0,7,  4,   2
X 1,14007
************************************************************
```

```
********************* #14014   MEMBER *********************
 S_M_BEND          ,14014,13,1
 KIND              , 1,1, 0,1,  4,   0
 LENGTH            , 2,1, 0,1,  4,   4
 SYSUSE            , 3,1, 0,1,  4,   8
 VERSION           , 4,1, 0,1,  4,  12
 SYS_IDENT         , 5,1, 0,1,  4,  16
 IDENT             , 6,1, 0,1,  4,  20
BDISPLAY           , 7,1, 0,9,  0,  24
 DISPLAYED         , 8,1, 0,4,  1,  24
 RBG_LEVEL         , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY         ,10,1, 0,1,  1,  28
 SYMBOL            ,11,1, 0,1,  1,  29
EDISPLAY           ,12,1, 0,9,  6,  30
 BENDLINE          ,13,1, 1,7,  4,   1
X 1, 5000
A  1,254
 RADIUS            ,14,1, 1,8,  4,   2
X 1, 1211
A  1,254
 BEND              ,15,1, 1,8,  4,   3
X 1, 1210
A  1,254
************************************************************

********************* #15000   MEMBER *********************
 TURNED_CLASS      ,15000,19
 15001             , 1,0, 0,0,  0,   0
 15002             , 2,0, 0,0,  0,   0
 15003             , 3,0, 0,0,  0,   0
 15004             , 4,0, 0,0,  0,   0
 15005             , 5,0, 0,0,  0,   0
 15006             , 6,0, 0,0,  0,   0
 15007             , 7,0, 0,0,  0,   0
 15008             , 8,0, 0,0,  0,   0
 15009             , 9,0, 0,0,  0,   0
 15010             ,10,0, 0,0,  0,   0
 15011             ,11,0, 0,0,  0,   0
 15012             ,12,0, 0,0,  0,   0
 15013             ,13,0, 0,0,  0,   0
 15014             ,14,0, 0,0,  0,   0
 15015             ,15,0, 0,0,  0,   0
 15016             ,16,0, 0,0,  0,   0
 15017             ,17,0, 0,0,  0,   0
 15018             ,18,0, 0,0,  0,   0
 15019             ,19,0, 0,0,  0,   0
************************************************************
```

```
********** #15001   MEMBER **********
 TRN_END          ,15001,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 EOP              ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************

********** #15002   MEMBER **********
 TRN_GROOVE       ,15002,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 GROOVE           ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************
```

```
********************* #15003   MEMBER *********************
 TRN_OPEN_DIA     ,15003,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 OPEN_DIA         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************

********************* #15004   MEMBER *********************
 TRN_REC_DIA      ,15004,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 RECESS_DIA       ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************
```

```
********** #15005   MEMBER **********
 TRN_RELIEF       ,15005,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 RELIEF           ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************

********** #15006   MEMBER **********
 TRN_TAPER        ,15006,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TAPER            ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************
```

```
********************* #15007   MEMBER *********************
 TRN_TRANS        ,15007,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TRANSITION       ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
*************************************************************

********************* #15008   MEMBER *********************
 TRN_THREAD       ,15008,14,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 TPU              ,15,1, 0,2,  4,  32
 MAJOR            ,16,1, 0,2,  4,  36
 FORM             ,13,1, 0,5,  1,  40
X 9,SHARP_V          ,UNIFIED_NATIONAL,SQUARE          ,SIXTY_DEG_STUB  ,
X 9,ACME             ,STUB_ACME       ,BUTTRESS        ,KNUCKLE         ,
X 9,BRIT_STAND_WIT
 CLASS            ,14,1, 0,5,  1,  41
X 6,ONE_A            ,TWO_A           ,THREE_A         ,ONE_B           ,
X 6,TWO_B            ,THREE_B
*************************************************************
```

```
********************* #15009   MEMBER *********************
 TRN_UNDERCUT     ,15009,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 UNDERCUT         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
*************************************************************

********************* #15010   MEMBER *********************
 TRN_OPEN_FACE    ,15010,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 OPEN_FACE        ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
*************************************************************
```

```
********************* #15011   MEMBER *********************
 TRN_REC_FACE     ,15011,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 RECESS           ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************

********************* #15012   MEMBER *********************
 TRN_S_O_FACE     ,15012,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 SEMIOPEN         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************
```

```
********** #15013   MEMBER **********
 TRN_S_O_DIA      ,15013,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 SEMIOPEN         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
**************************************************************

********** #15014   MEMBER **********
 TRN_AXIS         ,15014,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 PO               ,13,1, 0,7,  4,   1
X 1, 4000
 P1               ,14,1, 0,7,  4,   2
X 1, 4000
**************************************************************
```

```
********************* #15015   MEMBER *********************
 GAGE_POINT       ,15015,12,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 GAGE_LOC         ,13,1, 0,7,  4,   1
X 1, 4001
 GAGE_REF         ,14,1, 0,7,  4,   2
X 4, 3000, 4000, 5000, 6000
************************************************************

********************* #15016   MEMBER *********************
 TURNED_CHAMFER   ,15016,11,1
 KIND             , 1,1, 0,1,  4,   0
 LENGTH           , 2,1, 0,1,  4,   4
 SYSUSE           , 3,1, 0,1,  4,   8
 VERSION          , 4,1, 0,1,  4,  12
 SYS_IDENT        , 5,1, 0,1,  4,  16
 IDENT            , 6,1, 0,1,  4,  20
BDISPLAY          , 7,1, 0,9,  0,  24
 DISPLAYED        , 8,1, 0,4,  1,  24
 RBG_LEVEL        , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY        ,10,1, 0,1,  1,  28
 SYMBOL           ,11,1, 0,1,  1,  29
EDISPLAY          ,12,1, 0,9,  6,  30
 CHAM             ,13,1, 1,7,  4,   1
X 1, 3002
A  1,254
************************************************************
```

```
******************** #15017   MEMBER *********************
 TURNED_CORN_RND ,15017,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 C_ROUND         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************

******************** #15018   MEMBER *********************
 TURNED_FILLET   ,15018,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 C_FILET         ,13,1, 1,7,  4,   1
X 1, 8002
A  1,254
************************************************************
```

```
********************* #15019   MEMBER *********************
 TURNED_PROFILE  ,15019,11,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 PROF            ,13,1, 1,7,  4,   1
X 1, 5000
A  1,254
************************************************************

********************* #15020   MEMBER *********************
 EDGE_BREAK      ,15020,13,1
 KIND            , 1,1, 0,1,  4,   0
 LENGTH          , 2,1, 0,1,  4,   4
 SYSUSE          , 3,1, 0,1,  4,   8
 VERSION         , 4,1, 0,1,  4,  12
 SYS_IDENT       , 5,1, 0,1,  4,  16
 IDENT           , 6,1, 0,1,  4,  20
BDISPLAY         , 7,1, 0,9,  0,  24
 DISPLAYED       , 8,1, 0,4,  1,  24
 RBG_LEVEL       , 9,3, 1,1,  1,  25
A  3,  3
 INTENSITY       ,10,1, 0,1,  1,  28
 SYMBOL          ,11,1, 0,1,  1,  29
EDISPLAY         ,12,1, 0,9,  6,  30
 MINRAD          ,13,1, 0,2,  4,  32
 MAXRAD          ,14,1, 0,2,  4,  36
 E_B_EDGES       ,15,1, 1,7,  4,   1
X 2, 8001, 8002
A  1,254
************************************************************
```

SPECIFICATION CHANGE NOTICE

DATE PREPARED

1. ORIGINATOR NAME AND ADDRESS	2. [X] PROPOSED [ ] APPROVED	3. SPEC NO. PS 560130000 4. (ICAM OFFICE ONLY) SCN NO.	5. CMO USE ONLY DATE LOGGED DATE CLOSED INITIALS

6. SYSTEM DESIGNATION	7. RELATED ECP NO.	8. ICAM PROJECT NO.	9. CONTRACTUAL ACTIVITY
PDDI	5601		Materials Laboratory Air Force, Wright Aeronautical

10. CONFIGURATION ITEM NOMENCLATURE	11. EFFECTIVITY (CI NO.'S OF ALL ITEMS AFFECTED BY THIS SCN)
Product Definition Data Interface PDDI	

THIS NOTICE INFORMS RECIPIENTS THAT THE SPECIFICATION IDENTIFIED BY THE NUMBER (AND REVISION LETTER) SHOWN IN BLOCK 3 HAS BEEN CHANGED, THE PAGES CHANGED BY THIS SCN BEING THOSE FURNISHED HEREWITH AND CARRYING THE SAME DATE AS THIS SCN. THE PAGE NUMBERS AND DATES LISTED BELOW IN THE SUMMARY OF CHANGED PAGES, COMBINED WITH NON-LISTED PAGES OF THE ORIGINAL ISSUE OF THE REVISION SHOWN IN BLOCK 3, CONSTITUTE THE CURRENT VERSION OF THIS SPECIFICATION.

12. SCN NO.	13. PAGES CHANGED (INDICATE DELETIONS)	DEL	ADD	14. DATE
	PAGES CHANGED AND TRANSMITTED HEREWITH Appendices H, I, J.		X	22 Dec 87
	SUMMARY OF CHANGED PAGES The appendix reflects additional PDDI systems software developed under the Extensions effort. This PS in its entirety represents the "As Built" Software PDDI Version 3.			

15. ICAM CMB APPROVAL	DATE APPROVAL DATE

APPENDIX H

SCHEMA MANAGER

SOFTWARE DESCRIPTION

TABLE OF CONTENTS

			Page
SECTION 1		SCOPE	
	1.2	Identification	H-3
	1.2	Introduction	H-3
	1.3	System Environment	H-3
SECTION 2		REFERENCES	
	2.1	Applicable Documemts	H-5
	2.1.1	Other Publications	H-5
	2.2	Terms and Abbreviations.	H-5
SECTION 3		SYSTEM OPERATIONS	
	3.1	System Overview.	H-6
	3.2	Functional Description	H-6
SECTION 4		SCHEMA MANAGER HIERARCHY	

LIST OF ILLUSTRATIONS

Figure H-1 Schema Manager Architecture H-8

SECTION 1

SCOPE

1.1 Identification

This Appendix describes the "As Built" software specifications for the Schema Manager. The Schema Manager was developed under the Geometric Modeling Applications Interface Program (GMAP) and enhanced under this project - Product Definition Data Interface (PDDI). This project, 5601, was conducted under the Air Force Contract F33516-82-C-5036.

1.2 Introduction

The Schema Manager software is presented in its entirety in order to aid in the understanding of this software package. This manual is a reference document for programming personnel who maintain and enhance the PDDI software. The two other appendices present the Schema Manager in detail at three levels of complexity; the individual data entities (data dictionaries), the routines comprising the software and the relationships between these routines.

A data dictionary is presented in Appendix J to provide a complete listing of data within the Schema Manager software.

The software routines are listed in Appendix I for the Schema Manager. Routines are groups of code which perform a specific function. The software hierarchy presents the relationships between these routines.

1.3 System Environment

The PDDI system was developed in the following computing environment:

Computer/Operating System

IBM 43XX/MVS with TSO and associated tape drives, disk drives and terminals.

DEC VAX 11/780 VMS with associated tape drives, disk drives and terminals.

Storage (Core) Requirements

The development PDDI system required the following core requirements:

Model (large)	.57M	
PDDI Software	.66M	(No Overlay)
Database	3.00M	
	4.23M	

Compilers

IBM-Pascal/VS Release 2.2
DEC-Pascal V3.3, FORTRAN 77 V4.4

Terminals

327x (or equivalent) for non-graphics applications
E&S PS300 (or equivalent) for graphics applications

The PDDI system is transportable to other computing systems. However, appropriate local (native) interfaces must be provided.

SECTION 2

REFERENCES

2.1 Applicable Documents

This section provides applicable documents in addition to those listed in the main body of this document.

2.1.1 Other Publications

Product Definition Data Interface (PDDI)

UM 560130002	User's Manual - Schema Manager

Geometric Modelling Applications Interface Program (GMAP)

PS 560240031U	Product Specification
SS 560240001U	System Specification
SDS560240001U	System Design Specification

2.2 Terms and Abbreviations

This section provides applicable terms and abbreviations in addition to those listed in the main body of this report.

SCHEMA MANAGER - Software for creating, managing, and querying entity definitions for CAD/CAM models.

SCE - Schema Manager

SECTION 3

SYSTEM OPERATIONS

3.1 System Overview

The purpose of the PDDI Software System is to provide a prototype for the communication of complete Production Definition Data (PDD) between dissimilar CAD/CAM Systems. This system will serve as the information interface between Engineering and Manufacturing functions. It is composed of Access Software, Conceptual Schema, Schema Manager, Exchange Format and a Translator.

The Access Software is a set of callable utility programs that will allow applications to manipulate and query PDD. The Conceptual Schema is a data dictionary that contains the data needed to define a CAD/CAM model. The Schema Manager is the software mechanism for managing the data needed to define a CAD/CAM model, and for producing the data dictionary. The Exchange Format is a neutral physical sequential format for passing data between dissimilar systems. The PDDI Translator is the software mechanism for passing this data between the Exchange Format and the Working Form of the PDD.

This Appendix addresses the Schema Manager software for the PDDI Software System.

3.2 Functional Description

The Schema Manager is the software package used to manage the definitions of the entities contained in the Working Form. It has three major functions:

- o model a concrete conceptual schema
- o transform a concrete conceptual schema into a physical schema suitable for the Working Form of the PDDI Access Software
- o generate subschema forms of the physical schema for use by application programs at compile-time and/or run-time

The Schema Manager consists of three main sub-packages:

Interactive Interface
Batch Interface
Model Query Utility

The functions of the Interactive Interface include the creation, review, update, reporting, and retrieving of entity definitions. The Interactive Interface makes use of the IBM/SPF Dialog Manager for full-screen terminal menus. This interface was developed under the GMAP.

The Batch Interface provides a mechanism for the creation, reporting, and filing of entity definitions in a non-interactive mode. The Batch Interface uses the syntax of the EXPRESS information modeling language (PDES/STEP) for input.

The Model Query Utility provides a mechanism for querying the (part model) entities in the Working Form. The entity definitions are used to translate the Working Form binary representation of the (part model) entities into a list of the attribute names and their values. The Model Query Utility makes use of the IBM/SPF Dialog Manager for full-screen terminal menus.

The Schema Manager was developed for the IBM/MVS system environment. It was designed and implemented so that it could be transported to other computing systems. The machine dependencies are limited to the vendor-supplied Dialog Manager routines.

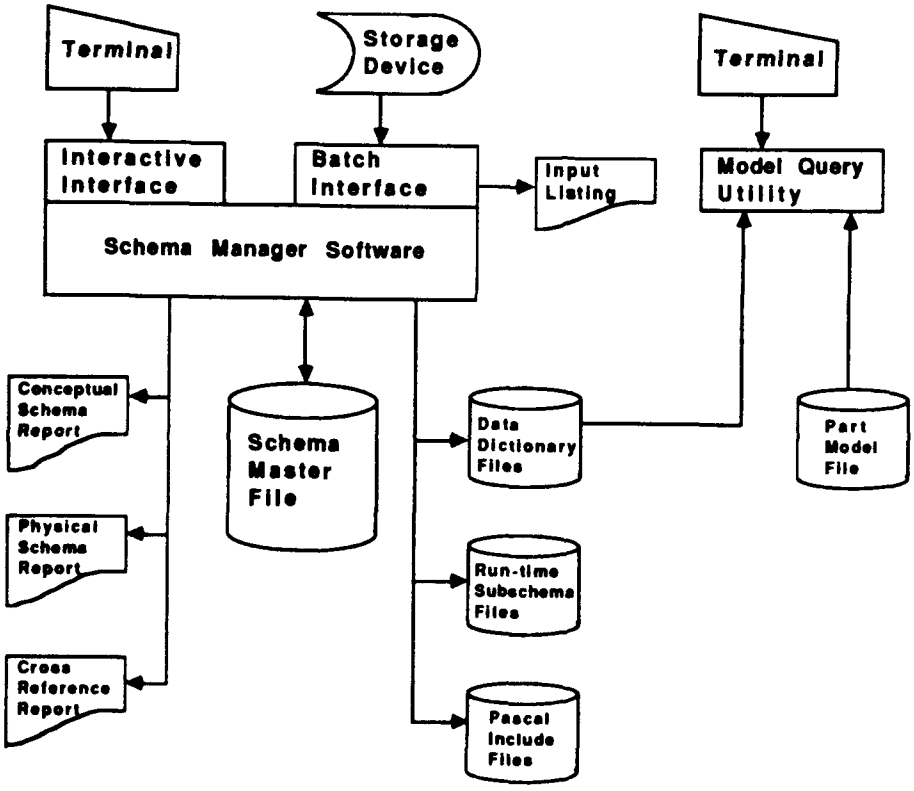


Figure H-1 Schema Manager Architecture

SECTION 4

SCHEMA MANAGER HIERARCHY

The balance of this appendix provides a cross-reference listing for Schema Manager routines. The Control Sections (CSECTs) that are referred to by a particular CSECT (routine) are provided.

Routine:	Refers to:
ADDENUM	SCELAB
	ISPLNK
	ISPLNKID

Routine:	Refers to:
ADDFIELD	SCELAB
	ISPLNK
	ISPLNK9
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
APPROVE	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK9
	ISPLNK12
	ISPLNKID

Routine:	Refers to:
BATDVR	FILRTV
	BATERR
	BATRPT
	BLDCLS
	BLDSUB
	BLEXICAL
	CLRSTK
	DEFCLS
	DEFENT
	DEFGBL
	DEFSUB
	DEFSUP
	DEFTYP
	MAECTK
	MAEGTK
	MAINIT
	MAKCNT
	MAL
	MALK
	MALRD
	MALSTF

SCXERRCK
SCXERRCK
SCXERRCK
SCEXIT

Routine: Refers to:

BATERR
ERRMSG
LEXICAL

Routine: Refers to:

BATRPT
SCELAB
BCSMAIN
DDREPORT
PHYSICAL
PSREPORT
RSFILE
BSCINCLD
BATERR
BLEXICAL
MAECTK
MAEGTK
MAEXEQ
MAKCNT
MALK
SCEXIT

Routine: Refers to:

BCSMAIN
SCELAB
CSCLSWRT
CSENTWRT
CSGBLWRT
CSINDWRT
CSRPTCVR
CSSUBWRT
CSSUPWRT
CSTYPWRT
MAL
MALD
MALK
MALSRT

Routine:	Refers to:
BLDARRAY	
	SCELAB
	MAEC
	MAECR
	MAEGTK
	MALD
	MALK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
BLDBPDEF	
	SCELAB
	MAECR

Routine:	Refers to:
BLDCLASS	
	SCELAB
	MAECR

Routine:	Refers to:
BLDCLS	
	SCELAB
	SCPUSHTR
	BSCTRSPR
	ENTCLS
	SCERRCK
	SCERRCK
	SCERRCK
	SCEXIT

Routine:	Refers to:
BLDDFTYP	
	SCELAB
	MAECR

Routine:	Refers to:
BLDEITEM	
	SCELAB
	MAECR
	MAEGTK

MALD
MALK
MALNO
MALRD
MALSTF

Routine: Refers to:

BLDENT
SCELAB
MAECR

Routine: Refers to:

BLDENUMR
SCELAB
MAECR

Routine: Refers to:

BLDFIELD
SCELAB
MAECR

Routine: Refers to:

BLDGBLFD
SCELAB
MAECR

Routine: Refers to:

BLDINT
SCELAB
MAECR
MAEGTK
MALD
MALK
MALNO
MALRD
MALSTF

Routine: Refers to:

BLDLOG
SCELAB
MAECR
MALD
MALGTK

MALK
MALNO
MALRD

Routine: Refers to:

BLDPNTR
SCELAB
MAECR

Routine: Refers to:

BLDREAL
SCELAB
MAECR
MAEGTK
MALD
MALK
MALNO
MALRD
MALSTF

Routine: Refers to:

BLDSSCMA
SCELAB
MAECR

Routine: Refers to:

BLDSTRNG
SCELAB
MAECR
MAEGTK
MALD
MALK
MALNO
MALRD
MALSTF

Routine: Refers to:

BLDSTRUC
SCELAB
MAECR

Routine:	Refers to:
BLDSUB	
	SCELAB
	SCPUSHTR
	BSCTRSPR
	ENTCLS
	SCEXIT

Routine:	Refers to:
BLDSUPER	
	SCELAB
	MAECR

Routine:	Refers to:
BLDUNRES	
	SCELAB
	MAECR

Routine:	Refers to:
BLEXICAL	
	SCELAB
	LEXICAL
	SCEXIT

Routine:	Refers to:
BSCINCLD	
	SCELAB
	PHYSICAL
	SCCONIN
	SCENTIN
	SCKEYIN
	SCMASIN
	SCPRMFL
	SCTYPIN
	MAECIK
	MAEGTK
	MAL
	MALD
	MALRDE
	MALSRT

Routine:	Refers to:
BSCTRSPR	
	SCELAB
	BLDARRAY
	BLDCLASS
	BLDDFTYP
	BLDEITEM
	BLDENT
	BLDENUMR
	BLDFIELD
	BLDGBLFD
	BLDINT
	BLDLOG
	BLDPNTR
	BLDREAL
	BLDSSCMA
	BLDSTRNG
	BLDSTRUC
	BLDSUPER
	BLDUNRES
	DEFADD
	DEFQUERY
	SCPOPKE
	SCPOPTR
	SCPUSHTR
	SCPUSHKE
	MAL
	MALATC
	MALD
	MALFND
	MALNO
	MALRD
	MALRPL
	MALRVS
	MALSTR

Routine:	Refers to:
CLRSTK	
	SCELAB
	SCPOPTR
	SCEXIT

Routine:	Refers to:
CRARRAY	
	SCELAB
	ISPLNK

Routine:	Refers to:
CRCLASS1	SCELAB
	ISPLNK
	ISPLNKID
	ISPLNK50

Routine:	Refers to:
CRCLASS2	SCELAB
	ISPLNK

Routine:	Refers to:
CRDEFTYP	SCELAB
	ISPLNK
	ISPLNKID

Routine:	Refers to:
CRENTITY	SCELAB
	ISPLNK
	ISPLNKID
	ISPLNK50

Routine:	Refers to:
CRENUM	SCELAB
	ISPLNK
	ISPLNKID

Routine:	Refers to:
CRFIELD	SCELAB
	ISPLNK
	ISPLNK9
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
CRINTGR	SCELAB
	ISPLNK

Routine:	Refers to:
CRLIST	SCELAB
	ISPLNK

Routine:	Refers to:
CRPNTR	SCELAB
	ISPLNK

Routine:	Refers to:
CRREAL	SCELAB
	ISPLNK

Routine:	Refers to:
CRSET	SCELAB
	ISPLNK

Routine:	Refers to:
CRSTRING	SCELAB
	ISPLNK

Routine:	Refers to:
CRSUBSCM	SCELAB
	ISPLNK
	ISPLNKID
	ISPLNKC8
	ISPLNK50

Routine:	Refers to:
CRSUPTYP	SCELAB
	ISPLNK
	ISPLNKID

Routine:	Refers to:
CSARYWRT	
	SCELAB
	CSDEFWRT
	CSHDGWRT
	CSINTWRT
	CSLOGWRT
	CSPTRWRT
	CSRELWRT
	CSSTGWRT
	MAEC
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSCLSHDG	
	CSNEWPG

Routine:	Refers to:
CSCLSWRT	
	SCELAB
	CSCLSHDG
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSDEFHDG	
	CSNEWPG

Routine:	Refers to:
CSDEFWRT	
	SCELAB
	MAEGTK

Routine:	Refers to:
CSENMWRT	
	SCELAB
	CSHDGWRT
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSENTHDG	
	CSNEWPG

Routine:	Refers to:
CSENTWRT	
	SCELAB
	CSARYWRT
	CSDEFWRT
	CSENTHDG
	CSINTWRT
	CSLOGWRT
	CSPTRWRT
	CSRELWRT
	CSSTGWRT
	MAEGTK
	MALD
	MALKL
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSGBLHDG	
	CSNEWPG

Routine:	Refers to:
CSGBLWRT	
	SCELAB
	CSARYWRT
	CSGBLHDG
	CSDEFWRT
	CSINTWRT
	CSLOGWRT
	CSPTRWRT

CSRELWRT
CSSTGWRT
MAEGTK
MAL
MALATC
MALNO
MALRD
MALSRT
MALSTF

Routine:	Refers to:
CSHDGWRT	
	CSCLSHDG
	CSDEFHDG
	CSENTHDG
	CSGBLHDG
	CSSUBHDG
	CSSUPHDG

Routine:	Refers to:
CSINDWRT	
	SCELAB
	CSCLSHDG
	CSENTHDG
	CSSUBHDG
	CSSUPHDG
	MAEC
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSINTWRT	
	SCELAB
	MAEGTK

Routine:	Refers to:
CSLOGWRT	
	SCELAB

Routine:	Refers to:
CSMAIN	
	SCELAB
	CSCLSWRT
	CSENTWRT
	CSGBLWRT
	CSINDWRT
	CSRPTCVR
	CSSUBWRT
	CSSUPWRT
	CSTYPWRT
	MAL
	MALD
	MALK
	MALSRT

Routine:	Refers to:
CSNEWPG	

Routine:	Refers to:
CSPTRWRT	
	SCELAB
	CSHDGWRT
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSRELWRT	
	SCELAB
	MAEGTK

Routine:	Refers to:
CSRPTCVR	

Routine:	Refers to:
CSSTGWRT	
	SCELAB
	MAEGTK

Routine:	Refers to:
CSSTRWRT	
	SCELAB
	CSARYWRT
	CSDEFWRT
	CSHDGWRT
	CSINTWRT
	CSLOGWRT
	CSPTRWRT
	CSRELWRT
	CSSTGWRT
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSSUBHDG	
	CSNEWPG

Routine:	Refers to:
CSSUBWRT	
	SCELAB
	CSSUBHDG
	MAEGTK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
CSSUPHDG	
	CSNEWPG

Routine:	Refers to:
CSSUPWRT	
	SCELAB
	CSARYWRT
	CSDEFWRT
	CSSUPHDG
	CSINTWRT
	CSLOGWRT
	CSPTRWRT
	CSRELWRT
	CSSTGWRT

MAEGTK
MALD
MALKL
MALNO
MALRD
MALSTF

Routine: Refers to:

CSTYPWRT
SCELAB
CSARYWRT
CSDEFHDG
CSDEFWRT
CSENMWRT
CSINTWRT
CSLOGWRT
CSPTRWRT
CSRELWRT
CSSTGWRT
CSSTRWRT
MAEGTK
MALNO
MALRD
MALSTF

Routine: Refers to:

DDABNDS
DDCL

Routine: Refers to:

DDADB

Routine: Refers to:

DDARRAY
DDABNDS
DDCL
DDENUM

Routine: Refers to:

DDCL

Routine:	Refers to:
DDCLASS	
	SCELAB
	MAEGTK
	MALNO
	MALRD
	MALSRT
	MALSTF
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DDENTITY	
	SCELAB
	DDWRITE
	MALRD
	MALSRT
	MALSTF
	RSGTSM
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DDENUM	

Routine:	Refers to:
DDREPORT	
	DDCLASS
	DDENTITY
	DDWRITE
	MAECIK
	MAEGTK
	PHYSICAL
	RS1100

Routine:	Refers to:
DDWRITE	
	DDADB
	DDARRAY
	DDCL
	DDENUM
	PSORDER

Routine:	Refers to:
DEFADD	
	SCELAB
	BLDBPDEF
	MAEGTK
	MAKCNT
	MALATC
	MALD
	MALK
	MALRD
	MALSTF
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DEFARR	
	SCELAB
	SCPUSHTR
	BATERR
	BLEXICAL
	CLRSTK
	DEFBAS
	DEFDEF
	SCEXIT

Routine:	Refers to:
DEFATT	
	SCELAB
	SCPUSHTR
	SCPOPTR
	SCUNQPND
	BATERR
	BLEXICAL
	CLRSTK
	DEFBAS
	DEFDEF
	MAEC
	MAEXEQ
	MALCPY
	MALD
	MALK
	MALKL
	MALNOT
	SCEXIT

Routine:	Refers to:
DEFBAS	
	SCELAB
	SCPUSHTR
	BATERR
	BLEXICAL
	CLRSTK
	DEFARR
	DEFPRE
	DEFPTR
	SCEXIT

Routine:	Refers to:
DEFCLS	
	SCELAB
	SCEXIST
	SCPUSHTR
	BSCTRSPR
	SCUNQEST
	SCUNQPND
	BATERR
	BLDUNRES
	BLEXICAL
	CLRSTK
	DEFADD
	ENTCLS
	ERRREC
	MAL
	MALATC
	MALD
	MALFND
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DEFDEF	
	SCELAB
	SCPUSHTR
	BLEXICAL
	CLRSTK
	ENTCLS
	DEFPTR
	MAKXEQ
	SCEXIT

Routine:	Refers to:
DEFENM	
	SCELAB
	SCPUSHTR
	SCUNQEST
	SCUNQPND
	BATERR
	BLEXICAL
	CLRSTK
	SCEXIT

Routine:	Refers to:
DEFENT	
	SCELAB
	SCEXIST
	SCPUSHTR
	BSCTRSPR
	SCUNQEST
	SCUNQPND
	BATERR
	BLEXICAL
	CLRSTK
	DEFATT
	ERRREC
	REFSUP
	SCEXIT

Routine:	Refers to:
DEFGBL	
	SCELAB
	SCPUSHTR
	BSCTRSPR
	BATERR
	BLEXICAL
	DEFATT
	ERRREC
	SCEXIT

Routine:	Refers to:
DEFPRE	
	SCELAB
	BATERR
	BLEXICAL
	CLRSTK
	SCEXIT

Routine:	Refers to:
DEFPTR	
	SCELAB
	SCPUSHTR
	SCEXIST
	BATERR
	BLDUNRES
	BLEXICAL
	CLRSTK
	DEFADD
	ENTCLS
	MAL
	MALATC
	MALD
	MALFND
	SCXERRCK
	SCERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DEFQUERY	
	SCELAB
	BLDPNTR
	MAEC
	MAED
	MAEGTK
	MAEU
	MAKCNT
	MALATC
	MALD
	MALFND
	MALK
	MALKL
	MALN
	MALRD
	MALRDE
	MALREP
	MALRPL
	MALSTF
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DEFSTC	
	SCELAB
	SCPUSHTR
	BATERR
	BLEXICAL
	CLRSTK
	DEFATT
	ERRREC
	SCERRCK
	SCEXIT

Routine:	Refers to:
DEFSUB	
	SCELAB
	SCEXIST
	SCPUSHTR
	BSCTRSPR
	SCUNQEST
	BATERR
	BLDUNRES
	BLEXICAL
	DEFADD
	ENTCLS
	ERRREC
	MAL
	MALATC
	MALD
	MALFND
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
DEFSUP	
	SCELAB
	SCEXIST
	SCPUSHTR
	BSCTRSPR
	SCUNQEST
	SCUNQPND
	BATERR
	BLEXICAL
	CLRSTK
	DEFATT

ERRREC
REFSUP
SCEXIT

Routine: Refers to:

DEFTYP

SCELAB
SCPUSHTR
BSCTRSPR
SCUNQEST
SCUNQPND
BATERR
BLEXICAL
CLRSTK
DEFBAS
DEFDEF
DEFENM
DEFSTC
ERRREC
MAKXEQ
SCEXIT

Routine: Refers to:

DISPLIST

SCELAB
ISPLNK
ISPLNK1
ISPLNKID
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

ENTCLS

SCELAB
MAKXEQ
SCEXIT

Routine: Refers to:

ERRMSG

Routine:	Refers to:
ERRREC	
	SCELAB
	CLRSTK
	SCERRCK
	SCEXIT

Routine:	Refers to:
GETDD	

Routine:	Refers to:
LEXICAL	

Routine:	Refers to:
LMEM23	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
MCREATE	
	SCELAB
	ISPLNK

Routine:	Refers to:
MFILMOD	
	SCELAB
	ISPLNK

Routine:	Refers to:
MINCLUD	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
MMAIN	SCELAB
	ISPLNK

Routine:	Refers to:
MNEWMOD	SCELAB
	ISPLNK

Routine:	Refers to:
MQBHALL	

Routine:	Refers to:
MQBHATT	

Routine:	Refers to:
MQBHATTS	

Routine:	Refers to:
MQBHENT	

Routine:	Refers to:
MQBHMAIN	

Routine:	Refers to:
MQCLMU	

Routine:	Refers to:
MQGETVAL	

Routine:	Refers to:
MQGTDEFN	

Routine:	Refers to:
MQIAATT	

Routine:	Refers to:
MQIAENT	

Routine:	Refers to:
MQIAMAIN	

Routine:	Refers to:
MQNCLMU	

Routine:	Refers to:
MQNUSRMU	

Routine:	Refers to:
MQUDVR	ISPLNK FILRTV MAECTK MAEKND MALK MALNO MQBHMAIN MQIAMAIN NVGTDD

Routine:	Refers to:
MQUSRMU	

Routine:	Refers to:
MREPORT	SCELAB ISPLNK

Routine:	Refers to:
MREVIEW	SCELAB ISPLNK

Routine:	Refers to:
MUPDATE	SCELAB ISPLNK

Routine: Refers to:

NVRTVRS

Routine: Refers to:

PHALFLD
SCELAB
MAEGTK
MALRD
MALSTF
MAEUD
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCEXIT

Routine: Refers to:

PHBYFPOS
SCELAB
MAEGTK
PHALFLD
PHGTFLD
PHSRTFLD
PHSRTORD
SCEXIT

Routine: Refers to:

PHDECBYT

Routine: Refers to:

PHENTITY
SCELAB
MALKL
PHBYFPOS
PHPOSITN
PHWOFPOS
SCXERRCK
SCERRCK
SCERRCK
SCERRCK
SCEXIT

Routine:	Refers to:
PHGLOBAL	
	SCELAB
	MAEC
	MALNO
	PHBYFPOS
	PHWOFPOS
	PHPOSITN
	SCEXIT

Routine:	Refers to:
PHGTFLD	
	MAEGTK
	MALRD
	MALSTF
	PHDECBYT

Routine:	Refers to:
PHPOSITN	
	SCELAB
	MAEGTK
	MAL
	MALATC
	MALRD
	MALSTF
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
PHSRTFLD	

Routine:	Refers to:
PHSRTORD	

Routine:	Refers to:
PHSUBTYP	
	SCELAB
	MAEGTK
	MALKL
	MALRD
	MALSTF
	PHBYFPOS

PHPOSITN
PHWOFPOS
SCXERRCK
SCXERRCK
SCERRCK
SCERRCK
SCERRCK
SCEXIT

Routine: Refers to:

PHWOFPOS
SCELAB
MAEGTK
MALD
MALNO
MALRD
MALSTF
PHALFLD
PHGTFLD
PHSRTFLD
SCXERRCK
SCEXIT

Routine: Refers to:

PHYSICAL
SCELAB
FILRTV
MAECIK
MAEGTK
MAEUD
MALK
MALKL
MALRD
MALSTF
PHENTITY
PHGLOBAL
PHSUBTYP
SCXERRCK
SCXERRCK
SCEXIT

Routine: Refers to:

PSORDER

Routine: Refers to:

PSRABNDS

Routine:	Refers to:
PSRADB	

Routine:	Refers to:
PSRARRAY	
	PSRABNDS
	PSRCL
	PSRENUM

Routine:	Refers to:
PSRCL	

Routine:	Refers to:
PSRENUM	

Routine:	Refers to:
PSREPORT	
	SCELAB
	MAECIK
	MAEGTK
	MALK
	MALRD
	MALSRT
	MALSTF
	PHYSICAL
	PSORDER
	PSRADB
	PSRARRAY
	PSRCL
	PSRENUM
	PSRHEAD
	PSRINDEX
	RSGTSM
	SCXERRCK
	SCXERRCK
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
PSRHEAD	

Routine:	Refers to:
PSRINDEX	
	SCELAB
	MAEGTK
	MALRD
	MALSRT
	MALSTF
	SCXERRCK
	SCXERRCK
	SCXERRCK
	SCEXIT

Routine:	Refers to:
REARRAY	
	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK12

Routine:	Refers to:
RECLASS	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNK50
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
REDEFTYP	
	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK12
	ISPLNKID

Routine:	Refers to:
REENTITY	
	SCELAB
	ISPLNK
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
REENUM	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
REFIELD	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK9
	ISPLNK12
	ISPLNKID

Routine:	Refers to:
REFIELD1	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
REFIELD2	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK9
	ISPLNK12
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
REFSUP	
	SCELAB
	BLDUNRES
	DEFADD
	SCPUSHTR
	MAKXEQ
	SCEXIT

Routine:	Refers to:
REINTGR	
	SCELAB
	ISPLNK
	ISPLNKC8

Routine:	Refers to:
RELIST	
	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK12

Routine:	Refers to:
REPNTR	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
REREAL	
	SCELAB
	ISPLNK
	ISPLNKC8

Routine:	Refers to:
RESET	
	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK12

Routine:	Refers to:
RESTRING	
	SCELAB
	ISPLNK
	ISPLNKC8

Routine:	Refers to:
RESTRUC	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
RESUBSCM	
	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNK50
	ISPLNKC8
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
RESUPTYP	
	SCELAB
	ISPLNK
	ISPLNKID

Routine:	Refers to:
RSCPAI	
	AMPXMOVE

Routine:	Refers to:
RSCPAT	
	AMPXMOVE

Routine:	Refers to:
RSCPCI	AMPXMOVE

Routine:	Refers to:
RSCPCT	AMPXMOVE

Routine:	Refers to:
RSCPEI	AMPXMOVE

Routine:	Refers to:
RSCPET	AMPXMOVE

Routine:	Refers to:
RSFILE	SCELAB MAECIK MAEGTK MALK MALRD MALSTF PHYSICAL RSGTSM RS1100 SCXERRCK SCXERRCK SCXERRCK SCEXIT

Routine:	Refers to:
RSGTSM	SCELAB MAEGTK MALKL MALNO MALRD MALSTF RSCPAI RSCPAT

RSCPCI
RSCPCT
RSCPEI
RSCPET
RSMASKND
RSTRGF
RSTRSM
RSTRST
SCXERRCK
SCERRCK
SCEXIT

Routine: Refers to:

RSMASKND

Routine: Refers to:

RSTRGF
SCELAB
MALK
MALNO
MALRD
MALSTF
RSTRSM
SCEXIT

Routine: Refers to:

RSTRSM
SCELAB
MAEC
MAECTK
MAEGTK
MALNO
MALRD
MALSTF
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCXERRCK
SCEXIT

Routine:	Refers to:
RSTRST	
	SCELAB
	MAEGTK
	MALK
	MALKL
	MALNO
	MALRD
	MALSTF
	RSTRSM
	SCXERRCK
	SCXERRCK
	SCERRCK
	SCEXIT

Routine:	Refers to:
RS1100	
	SCELAB
	MAEC
	MAECIK
	MAEGTK
	MAL
	MALATC
	MALFND
	MALK
	MALRDE
	MALKL
	MALNO
	MALRD
	MALSRT
	MALSTF
	RSCPAI
	RSCPAT
	RSCPCI
	RSCPCT
	RSCPEI
	RSCPET
	RSTRGF
	RSMASKND
	SCEXIT

Routine:	Refers to:
SCALFSRT	

Routine:	Refers to:
SCARYCR	
	SCELAB
	CRARRAY
	SCDEFCR
	SCINTCR
	SCLISTCR
	SCPOPTR
	SCPTRCR
	SCPUSHTR
	SCRELCR
	SCSETCR
	SCSTGCR

Routine:	Refers to:
SCARYUP	
	SCELAB
	SCPRMRE
	SCTYPUP
	UPARRAY
	MAEC
	MAECR
	MAEGTK
	MALATC
	MALD
	MALRD
	MALRPL
	MALSTF

Routine:	Refers to:
SCBASIN	
	SCELAB

Routine:	Refers to:
SCCHRCK	
	SCELAB

Routine:	Refers to:
SCCLSCR1	
	SCELAB
	CRCLASS1
	SCCHRCK
	SCCLSCR2
	SCPOPTR

SCPUSHTR
SCTRSPR
SCUNQEST
SCUNQPND

Routine: Refers to:

SCCLSCR2

SCELAB
CRCLASS2
SCCLSCR1
SCENTCR
SCEXIST
SCMEMAD
SCMEMLST
SCPOPTR
SCPUSHTR
SCTRSPR
MAL
MALATC
MALD
MALFND

Routine: Refers to:

SCCLSUP

SCELAB
SCCHRCK
SCMEMAD
SCPRMFL
SCPRMRE
SCUNQEST
UPCLASS1
UPCLASS2
MAEC
MAED
MAEGTK
MAEUD
MAEXEQ
MALATC
MALD
MALFND
MALNO
MALREP
MALRMV
MAUPDT

Routine:	Refers to:
SCCOMPAR	
	SCELAB

Routine:	Refers to:
SCCONIN	
	SCELAB
	MAEGTK
	MALRD
	MALSTF

Routine:	Refers to:
SCCREATE	
	SCELAB
	MCREATE
	SCCLSCR1
	SCDEFCR
	SCENTCR
	SCFLDCR
	SCPUSHTR
	SCSUBCR
	SCSUPCR
	SCTRSPR

Routine:	Refers to:
SCDEFCR	
	SCELAB
	CRDEFTYP
	SCARYCR
	SCCHRCK
	SCENUCR
	SCFLDCR
	SCINTCR
	SCKEFIND
	SCLISTCR
	SCPOPTR
	SCPTRCR
	SCPUSHTR
	SCRELCR
	SCSETCR
	SCSTGCR
	SCTRSPR
	SCUNQEST
	SCUNQPND

Routine:	Refers to:
SCDEFUP	
	SCELAB
	APPROVE
	UPDEFTYP
	SCCHRCk
	SCPRMRE
	SCTYPUP
	SCUNQEST
	MAEC
	MAECR
	MAEGTK
	MAEUD
	MAEUIK
	MALATC
	MALD
	MALNO
	MALRD
	MALRPL
	MALSTF

Routine:	Refers to:
SCENMUP	
	SCELAB
	ADDENUM
	BLDEITEM
	SCCHRCK
	SCPRMRE
	UPENUM
	MAEC
	MAECR
	MAED
	MAEGTK
	MAEXEQ
	MALAND
	MALATC
	MALD
	MALFND
	MALKL
	MALNO
	MALRD
	MALRMV
	MALRPL
	MALSTF

Routine:	Refers to:
SCENTCR	
	SCELAB
	CRENTITY
	SCCHRCK
	SCFLDCR
	SCPOPTR
	SCPUSHTR
	SCSUPCR
	SCTRSPR
	SCUNQEST
	SCUNQPND

Routine:	Refers to:
SCENTIN	
	SCELAB
	SCFLDIN
	SCFLDST
	MAEGTK
	MAL
	MALATC
	MALD
	MALKL
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
SCENTUP	
	SCELAB
	APPROVE
	SCFLDAD
	SCFLDUP
	SCPOPTR
	SCPRMRE
	SCPUSHTR
	SCUNQEST
	SCSUPCR
	SCSUPUP
	SCTRSPR
	UPENTY1
	UPENTY2
	MAED
	MAEGTK
	MAEUD
	MAEUIK

MAEXEQ
MAL
MALATC
MALD
MALFND
MALKL
MALNO
MALRD
MALREP
MALRMV
MALRPL
MALSTF
MAUPDT

Routine: Refers to:

SCENUCR
SCELAB
CRENUM
SCCHRCK
SCMEMLST
SCPOPTR
SCPUSHTR
SCUNQEST
SCUNQPND

Routine: Refers to:

SCEXIST
SCELAB
MAKXEQ

Routine: Refers to:

SCFLDAD
SCELAB
ADDFIELD
SCARYCR
SCCHRCK
SCDEFCR
SCINTCR
SCLISTCR
SCPOPTR
SCPTRCR
SCPUSHTR
SCRELCR
SCSETCR
SCSTGCR
SCTRSPR

SCUNQEST
SCUNQPND
MAEXEQ
MAL
MALD
MALK
MALKL
MALNOT
MALRD
MALSTF
MAKXEQ

Routine: Refers to:

SCFLDCR

SCELAB
APPROVE
CRFIELD
SCARYCR
SCCHRCK
SCDEFCR
SCINTCR
SCLISTCR
SCMEMLST
SCPOPTR
SCPTRCR
SCPUSHTR
SCRELCR
SCSETCR
SCSTGCR
SCUNQEST
SCUNQPND
MAEC
MAEGKN
MAEGTK
MAEUD
MAEXEQ
MAKXEQ
MALD
MALK
MALKL
MALNO
MALRD
MALSTF

Routine:	Refers to:
SCFLDIN	
	SCELAB
	SCBASIN
	MAEC
	MAEGTK
	MALRD
	MALSRT
	MALSTF

Routine:	Refers to:
SCFLDSRT	

Routine:	Refers to:
SCFLDST	
	SCELAB
	MAECIK
	MAEGTK
	MALATC
	MALD
	MALNO
	MALRD
	MALSRT
	MALSTF

Routine:	Refers to:
SCFLDUP	
	SCELAB
	APPROVE
	SCCHRCK
	SCPRMRE
	SCTYPUP
	SCUNQEST
	UPFIELD
	MAEC
	MAECR
	MAED
	MAEGTK
	MAEU
	MAEUD
	MAEXEQ
	MAKXEQ
	MALATC
	MALD
	MALFND

MALK
MALKL
MALNO
MALRD
MALRPL
MALSTF
MAUPDT

Routine:	Refers to:
SCFNDKEY	
	MAEC
	MAEXEQ
	MALD

Routine:	Refers to:
SCGENRPT	
	SCELAB
	DDREPORT
	MINCLUD
	PHYSICAL
	PSREPORT
	RSFILE
	SCPRMFL
	MAEGTK
	MAEXEQ
	MALD
	MALK

Routine:	Refers to:
SCHDVR	
	MAECTK
	MAINIT
	MAL
	FILRTV
	MFILMOD
	MMAIN
	MNEWMOD
	SCCREATE
	SCREVIEW
	SCRPTM
	SCUPDATE

Routine:	Refers to:
SCINCLD	
	SCELAB
	MINCLUD
	PHYSICAL
	SCCONIN
	SCENTIN
	SCKEYIN
	SCMASIN
	SCPRMFL
	SCTYPIN
	MAECIK
	MAEGTK
	MAEXEQ
	MAL
	MALD
	MALK
	MALRDE
	MALSRT

Routine:	Refers to:
SCINTCR	
	SCELAB
	CRINTGR
	SCPOPTR
	SCPUSHTR

Routine:	Refers to:
SCINTUP	
	SCELAB
	BLDINT
	UPINT
	MAEGTK
	MALATC

Routine:	Refers to:
SCKEFIND	
	SCELAB
	MAEC
	MAECIK
	MAEUIK
	MAKXEQ
	MALD
	MALK
	MALNOT

Routine:	Refers to:
SCKEYIN	SCELAB
	MAEGTK
	MALFND
	MALRD
	MALSTF

Routine:	Refers to:
SCLISTCR	SCELAB
	CRLIST
	SCARYCR
	SCDEFCR
	SCINTCR
	SCPOPTR
	SCPTRCR
	SCPUSHTR
	SCRELCR
	SCSETCR
	SCSTGCR

Routine:	Refers to:
SCLISTUP	SCELAB
	SCPRIRE
	SCTYPUP
	UPLIST
	MAEC
	MAECR
	MAEGTK
	MALATC
	MALD
	MALRD
	MALRPL
	MALSTF

Routine:	Refers to:
SCMASIN	SCELAB
	MAEC
	MAEGTK
	MAL
	MALD
	MALFND

MALK
MALNO
MALRD
MALSTF
SCFLDIN
SCFLDST

Routine: Refers to:

SCMEMAD

SCELAB
MAKXEQ
MALD
MALFND
MALK
MALNO
MALNOT
MALRMV

Routine: Refers to:

SCMEMLST

SCELAB
MAEGTK
MALNO

Routine: Refers to:

SCPOPKE

SCELAB

Routine: Refers to:

SCPOPTR

SCELAB

Routine: Refers to:

SCPRMFL

SCELAB
MAEGTK
MALSTF
MALRD

Routine:	Refers to:
SCPRMRE	
	SCELAB
	RECLASS
	REENTITY
	REENUM
	REPNTR
	RESUBSCM
	RESTRUC
	REARRAY
	REDEFTYP
	REFIELD1
	REFIELD2
	REINTGR
	RELIST
	REREAL
	RESET
	RESTRING
	RESUPTYP
	MAEC
	MALKL
	MAEGTK
	MAEXEQ
	MALD
	MALSTF
	MALRD

Routine:	Refers to:
SCPTRCR	
	SCELAB
	CRPNTR
	SCEXIST
	SCMEMAD
	SCMEMLST
	SCPOPTR
	SCPUSHTR
	MAL
	MALATC
	MALFND
	MALD

Routine:	Refers to:
SCPTRUP	
	SCELAB
	SCMEMAD
	SCPRMFL
	SCPRMRE

UPPNTR
MAEC
MAECR
MAEGTK
MAEXEQ
MALATC
MALD
MALFND
MALNO
MALRMV

Routine: Refers to:

SCPUSHKE
SCELAB

Routine: Refers to:

SCPUSHTR
SCELAB

Routine: Refers to:

SCRELCR
SCELAB
CRREAL
SCPOPTR
SCPUSHTR

Routine: Refers to:

SCRELUP
SCELAB
BLDREAL
UPREAL
MAEGTK
MALATC

Routine: Refers to:

SCREVIEW
SCELAB
MREVIEW
SCKEFIND
SCPRMRE

Routine:	Refers to:
SCRPTM	
	SCELAB
	XMAIN
	CSMAIN
	MREPORT
	SCINCLD
	SCGENRPT

Routine:	Refers to:
SCSETCR	
	SCELAB
	CRSET
	SCARYCR
	SCDEFCR
	SCINTCR
	SCLISTCR
	SCPOPTR
	SCPTRCR
	SCPUSHTR
	SCRELCR
	SCSTGCR

Routine:	Refers to:
SCSETUP	
	SCELAB
	SCPRMRE
	SCTYPUP
	UPSET
	MAEC
	MAECR
	MAEGTK
	MALATC
	MALD
	MALRD
	MALRPL
	MALSTF

Routine:	Refers to:
SCSTCUP	
	SCELAB
	SCFLDAD
	SCFLDUP
	SCPRMRE
	UPSTRUC

MAEC
MAECR
MAEGTK
MAEXEQ
MALAND
MALATC
MALD
MALFND
MALNO
MALRD
MALRMV
MALRPL
MALSTF

Routine: Refers to:

SCSTGCR

SCELAB
CRSTRING
SCPOPTR
SCPUSHTR

Routine: Refers to:

SCSTGUP

SCELAB
BLDSTRNG
UPSTRING
MAEGTK
MALATC

Routine: Refers to:

SCSUBCR

SCELAB
CRSUBSCM
SCCHRCK
SCEXIST
SCMEMAD
SCMEMLST
SCPOPTR
SCPUSHTR
SCTRSPR
SCUNQEST
MAL
MALATC
MALD
MALFND

Routine:	Refers to:
SCSUBUP	
	SCELAB
	APPROVE
	SCCHRCK
	SCMEMAD
	SCPRMFL
	SCPRMRE
	SCUNQEST
	UPSUB1
	UPSUB2
	MAEC
	MAED
	MAEGTK
	MAEUD
	MAEXEQ
	MALATC
	MALD
	MALFND
	MALNO
	MALRDE
	MALREP
	MALRMV
	MAUPDT

Routine:	Refers to:
SCSUPCR	
	SCELAB
	CRENTITY
	CRSUPTYP
	SCCHRCK
	SCFLDCR
	SCKEFIND
	SCPOPTR
	SCPUSHTR
	SCTRSPR
	SCUNQEST
	SCUNQPND
	MAKXEQ

Routine:	Refers to:
SCSUPUP	
	SCELAB
	APPROVE
	SCFLDAD
	SCFLDUP

SCPRMRE
SCPUSHTR
SCUNQEST
SCSUPCR
SCTRSPR
UPSUPER
UPENTY2
MAED
MAEGTK
MAEUD
MAEUIK
MAEXEQ
MAL
MALATC
MALD
MALFND
MALKL
MALNO
MALRD
MALREP
MALRMV
MALRPL
MALSTF
MAUPDT

Routine: Refers to:

SCTRSPR

SCELAB
BLDARRAY
BLDCLASS
BLDDFTYP
BLDEITEM
BLDENT
BLDENUMR
BLDFIELD
BLDGBLFD
BLDINT
BLDLOG
BLDPNTR
BLDREAL
BLDSSCMA
BLDSTRNG
BLDSTRUC
BLDSUPER
SCPOPKE
SCPOPTR
SCPUSHTR
SCPUSHKE

MAL
MALATC
MALD
MALFND
MALNO
MALRD
MALRPL
MALRVS
MALSTR

Routine: Refers to:

SCTYPIN

SCELAB
SCBASIN
MAEGTK
MALK
MALRD
MALRDE
MALSTF

Routine: Refers to:

SCTYPUP

SCELAB
SCARYCR
SCARYUP
SCDEFCR
SCDEFUP
SCENUCR
SCENMUP
SCFLDCR
SCINTCR
SCINTUP
SCLISTCR
SCLISTUP
SCPOPTR
SCPTRCR
SCPTRUP
SCPUSHTR
SCRELCR
SCRELUP
SCSETCR
SCSETUP
SCSTCUP
SCSTGCR
SCSTGUP
SCTRSPR

MAEU
MAL
MALATC
MALD
MALGTK
MALNO

Routine: Refers to:

SCUNIQUE
SCELAB

Routine: Refers to:

SCUNQEST
SCELAB
MAKXEQ

Routine: Refers to:

SCUNQPND
SCELAB

Routine: Refers to:

SCUPDATE
SCELAB
MUPDATE
SCCLSUP
SCDEFUP
SCENTUP
SCFLDUP
SCKEFIND
SCSUBUP
SCSUPUP
MAEU
MAED
MAL
MALD
MALNO
MAUPDT

Routine: Refers to:

SORTKIND

Routine: Refers to:

SORTNAME

Routine:	Refers to:
UPARRAY	SCELAB
	ISPLNK
	ISPLNK12

Routine:	Refers to:
UPCLASS1	SCELAB
	ISPLNK
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
UPCLASS2	SCELAB
	ISPLNK
	ISPLNK1
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
UPDEFTYP	SCELAB
	ISPLNK
	ISPLNK12
	ISPLNKID

Routine:	Refers to:
UPENTY1	SCELAB
	ISPLNK
	ISPLNKC8
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
UPENTY2	SCELAB
	ISPLNK
	ISPLNK1

ISPLNKID
ISPLNKC6
ISPLNKC8
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

UPENUM
SCELAB
ISPLNK
ISPLNK1
ISPLNKC8
ISPLNKID
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

UPFIELD
SCELAB
ISPLNK
ISPLNK1
ISPLNKC8
ISPLNK9
ISPLNK12
ISPLNK50
ISPLNKID

Routine: Refers to:

UPINT
SCELAB
ISPLNK

Routine: Refers to:

UPLIST
SCELAB
ISPLNK
ISPLNK12

Routine: Refers to:

UPPNTR
SCELAB
ISPLNK
ISPLNK1

ISPLNKID
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

UPREAL
SCELAB
ISPLNK

Routine: Refers to:

UPSET
SCELAB
ISPLNK
ISPLNK12

Routine: Refers to:

UPSTRING
SCELAB
ISPLNK

Routine: Refers to:

UPSTRUC
SCELAB
ISPLNK
ISPLNK1
ISPLNKID
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

UPSUB1
SCELAB
ISPLNK
ISPLNK50
ISPLNKID

Routine: Refers to:

UPSUB2
SCELAB
ISPLNK
ISPLNK1

ISPLNKID
ISPLNKTV
ISPLNKTB
ISPLNKTD

Routine: Refers to:

UPSUPER

SCELAB
ISPLNK
ISPLNKC8
ISPLNKID

Routine: Refers to:

XATTDATA

SCELAB
XATTRES
XMPRESPE
XMTYPSPE
MAEUIK
MAEXEQ
MAL
MALCPY
MALD
MALK
MALNO

Routine: Refers to:

XATTNAME

SCELAB
XMNAMSPE
XNAMENUM
XRESULT
MAEGKN
MAEU
MAKXEQ
MAL
MALATC
MALD
MALNO
MALRD
MALSTF

Routine:	Refers to:
XATTRES	SCELAB
	XMATTRES
	MAEGTK
	MAEU
	MALD
	MALRD
	MALSTF

Routine:	Refers to:
XEXPREC	SCELAB
	XMRESULT
	MAEGTK
	MALD
	MALK
	MALNO
	MALRD
	MALSTF

Routine:	Refers to:
XFNDARY	MALATC

Routine:	Refers to:
XFNDKEY	SCELAB
	MAKXEQ

Routine:	Refers to:
XFNDNAME	MALATC

Routine:	Refers to:
XFNDPREC	MALATC

Routine:	Refers to:
XLISTENT	
	SCELAB
	XATTRES
	XFNDKEY
	XMNAMSPE
	XNAMENUM
	XRESULT
	MAEU
	MAEUIK
	MALD
	MALKL

Routine:	Refers to:
XMAIN	
	SCELAB
	XATTDATA
	XATTNAME
	XEXPREC
	XLISTENT
	XMMAIN

Routine:	Refers to:
XMATTRES	
	SCELAB
	ISPLNK
	ISPLNK50
	ISPLNKRV
	ISPLNKRB
	ISPLNKTD

Routine:	Refers to:
XMMAIN	
	SCELAB
	ISPLNK

Routine:	Refers to:
XMNAMSPE	
	SCELAB
	ISPLNK
	ISPLNK50
	ISPLNKID

Routine:	Refers to:
XMPRESPE	
	SCELAB
	ISPLNK

Routine:	Refers to:
XMRESULT	
	SCELAB
	ISPLNK
	ISPLNK50
	ISPLNKID
	ISPLNKTV
	ISPLNKTB
	ISPLNKTD

Routine:	Refers to:
XMTYPSPE	
	SCELAB
	ISPLNK

Routine:	Refers to:
XNAMENUM	
	SCELAB
	SCCHRCK

Routine:	Refers to:
XRESULT	
	SCELAB
	XMRESULT
	MALRD
	MAEGTK

APPENDIX I

SCHEMA MANAGER ROUTINES

This appendix provides a listing of each procedure in the Schema Manager Software Package.

The routines are listed in alphabetic order. An index with a brief description of the routine function is provided.

A hierarchy dictionary is provided in Appendix D to show the relationship of the routines.

Routine Index I-2
Routine Dictionary. I-10

ROUTINE INDEX

ADDENUM - DISPLAYS THE ADD ENUMERATION MENU
ADDFIELD - DISPLAYS THE ADDFIELD PANEL
APPROVE - DISPLAYS THE APPROVE UPDATE PANEL
BATDVR - Mainline program for Batch Interface.
BATERR - Error reporting and recovery for Batch Interface.
BATRPT - Batch Interface routine that invokes the necessary routines to generate a specified report.
BCSMAIN - THIS ROUTINE SERVES AS THE MAIN DRIVER FOR THE CONCEPTUAL SCHEMA REPORT.
BLDARRAY - BUILDS THE ARRAY ENTITY.
BLDBPDEF - THIS ROUTINE BUILDS THE BACKPATCH ENTITY.
BLDCLASS - BUILDS THE CLASS ENTITY.
BLDCLS - Batch Interface routine that creates a class entity in the schema model from a class name, kind number, and a list of constituents.
BLDDFTYP - BUILDS THE DEFINED TYPE ENTITY.
BLDEITEM - BUILDS THE ENUMERITEM ENTITY.
BLDENT - BUILDS THE ENTITY ENTITY.
BLDENUMR - BUILDS THE ENUMERATION ENTITY.
BLDFIELD - BUILDS THE FIELD ENTITY.
BLDGBLFD - BUILDS THE GLOBAL FIELD ENTITY.
BLDINT - BUILDS THE INTEGER ENTITY.
BLDLOG - BUILDS THE LOGICAL ENTITY.
BLDPNTR - BUILDS THE POINTER ENTITY.
BLDREAL - BUILDS THE REAL ENTITY.
BLDSSCMA - BUILDS THE SUBSCHEMA ENTITY.
BLDSTRNG - BUILDS THE STRING ENTITY.
BLDSTRUC - BUILDS THE STRUCTURE ENTITY.
BLDSUB - Batch Interface routine that creates a subschema entity in the schema model from a subschema name and list of constituents.
BLDSUPER - THIS ROUTINE BUILDS THE SUPERTYPE ENTITY.
BLDUNRES - THIS ROUTINE BUILDS THE UNRESOLVED ENTITY.
BLEXICAL - LOCATE THE LONGEST POSSIBLE LEXEME FROM WHICH A TOKEN MAY BE DETERMINED, EXCLUDING COMMENTS.
BSCINCLD - THIS ROUTINE GENERATES THE PASCAL INCLUDE FILES AND WRITES THESE DEFINITIONS TO A FILE
BSCTRSPR - THIS ROUTINE BEGINS THE PROCESSING OF THE TRANSACTION STACK.
CLRSTK - Batch Interface routine that clears the transaction processing stack
CRARRAY - DISPLAYS THE CREATE ARRAY MENU
CRCLASS1 - DISPLAYS THE CREATE CLASS1 PANEL
CRCLASS2 - DISPLAYS THE CREATE CLASS PANEL 2 MENU
CRDEFTYP - DISPLAYS THE CREATE DEFINED TYPE MENU
CRENTITY - DISPLAYS THE CREATE ENTITY MENU
CRENUM - DISPLAYS THE CREATE ENUMERATION MENU
CRFIELD - DISPLAYS THE CREATE FIELD PANEL
CRINTGR - DISPLAYS THE CREATE INTEGER MENU
CRLIST - DISPLAYS THE CREATE LIST PANEL

CRPNTR - DISPLAYS THE CREATE POINTER MENU
CRREAL - DISPLAYS THE CREATE REAL PANEL
CRSET - DISPLAYS THE CREATE SET PANEL
CRSTRING - DISPLAYS THE CREATE STRING PANEL
CRSUBSCM - DISPLAYS THE CREATE SUBSCHEMA PANEL
CRSUPTYP - DISPLAYS EITHER THE CREATE/REFERENCE SUPERTYPE MENU OR THE CREATE SUPERTYPE MENU
CRURUL - CREATES THE USER'S RULES. RULES OF CONNECTIVITY USED TO DETERMINE DELETABILITY OF ENTITIES.
CSARYWRT - WRITES OUT AN ARRAY DEFINITION
CSCLSHDG - WRITES OUT A CLASS HEADING ON A NEW PAGE.
CSCLSWRT - WRITES OUT CLASS DEFINITIONS TO A FILE
CSDEFHDG - WRITES OUT A DEFINED TYPE HEADING ON A NEW PAGE.
CSDEFWRT - WRITES OUT A DEFINED TYPE NAME
CSENMWRT - WRITES OUT AN ENUMERATION DEFINITION
CSENTHDG - WRITES OUT AN ENTITY HEADING ON A NEW PAGE.
CSENTWRT - WRITES OUT ENTITY DEFINITIONS TO A FILE
CSGBLHDG - WRITES OUT A GLOBAL FIELD HEADING ON A NEW PAGE
CSGBLWRT - WRITES OUT GLOBAL FIELD DEFINITIONS TO A FILE
CSHDGWRT - CALLS THE APPROPRIATE HEADING ROUTINE.
CSINDWRT - WRITES OUT TO A FILE AN INDEX FOR AN ENTITY, CLASS, OR SUBSCHEMA.
CSINTWRT - WRITES OUT AN INTEGER DEFINITON
CSLOGWRT - WRITES OUT A LOGICAL DEFINITION
CSMAIN - SERVES AS THE MAIN DRIVER FOR THE CONCEPTUAL SCHEMA REPORT.
CSNEWPG - CREATES A NEW PAGE IN THE CONCEPTUAL SCHEMA REPORT.
CSPTRWRT - WRITES OUT A POINTER DEFINITION
CSRELWRT - WRITES OUT A REAL DEFINITION
CSRPTCVR - PRINTS OUT THE REPORT COVER FOR THE CONCEPTUAL SCHEMA.
CSSTGWRT - WRITES OUT A STRING DEFINITON
CSSTRWRT - WRITES OUT A STUCTURE DEFINITION
CSSUBHDG - WRITES OUT A SUBSCHEMA HEADING ON A NEW PAGE.
CSSUBWRT - WRITES OUT SUBSCHEMA DEFINITIONS TO A FILE.
CSSUPHDG - THIS ROUTINE WRITES OUT A SUPERTYPE HEADING ON A NEW PAGE.
CSSUPWRT - THIS ROUTINE WRITES OUT SUPERTYPE DEFINITIONS TO A FILE
CSTYPWRT - WRITES OUT THE DEFINED TYPE DEFINITIONS TO A FILE.
DDABNDS - WRITE THE LOW-BOUND AND UPPER-BOUND FOR THE ARRAY ATTRIBUTE
DDADB - WRITE THE BASIC RECORD OF AN ENTITY TO A SEQUENTIAL FILE
DDARRAY - WRITE THE ARRAY ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE
DDCL - WRITE THE CONSTITUENT REFERENCES OF AN ENTITY TO A SEQUENTIAL FILE
DDCLASS - WRITE THE CLASS KINDS TO A SEQUENTIAL FILE
DDENTITY - WRITE THE ENTITY KINDS TO A SEQUENTIAL FILE
DDENUM - WRITE THE ENUMERATION ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE
DDREPORT - WRITE THE DATA DICTIONARY IN CHARACTER FORM.
DDWRITE - WRITE THE ENTITY DEFINITIONS TO A SEQUENTIAL FILE
DEFADD - Batch Interface routine that adds an unresolved entity reference to the list of backpatch entities.
DEFARR - Batch Interface routine that processes an array definition.
DEFATT - Batch Interface routine that processes an attribute definition.

DEFBAS - Batch Interface routine that processes a primitive data type definition.
DEFCLS - Batch Interface routine that processes a class definition.
DEFDEF - Batch Interface routine that processes a defined type reference.
DEFENM - Batch Interface routine that processes an enumeration definition.
DEFENT - Batch Interface routine that processes an entity definition.
DEFGBL - Batch Interface routine that processes a global attribute definition.
DEFPRE - Batch Interface routine that processes an integer precision, real precision, or string length.
DEFPTR - Batch Interface routine that processes a pointer definition.
DEFQUERY - Batch Interface routine that determines if a newly modeled entity satisfies any unresolved entity references on the backpatch list.
DEFSTC - Batch Interface routine that processes a structure definition
DEFSUB - Batch Interface routine that processes a subschema definition
DEFSUP - Batch Interface routine that processes a supertype definition
DEFTYP - Batch Interface routine that processes a defined type definition.
DISPLIST - DISPLAYS A LIST OF ENTITIES.
ENTCLS - Batch Interface routine that determines if a specified identifier is a modeled entity or class.
ERRMSG - Batch Interface routine that writes appropriate error messages to the report file.
ERRREC - Batch Interface routine that performs the necessary actions to recover from an input error.
GETDD - READ THE DATA DICTIONARY INTO THE APPLICATION PROGRAM.
LEXICAL - LOCATE THE LONGEST POSSIBLE LEXEME FROM WHICH A TOKEN MAY BE DETERMINED.
LMEM23 - DISPLAYS THE LIST MEMBERS (LMEM23) PANEL
MCREATE - DISPLAYS THE CREATE MENU
MFILMOD - DISPLAYS THE FILE/RETRIEVE MENU
MINCLUD - DISPLAYS THE LIST OF SUBSCHEMAS
MMAIN - DISPLAYS THE MAIN MENU
MNEWMOD - DISPLAYS THE FILE/RETRIEVE MENU
MQBHALL - PRINT ALL ENTITIES IN THE MODEL
MQBHATT - PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND
MQBHATTS - PRINT ALL ENTITIES OF A SPECIFIC KIND
MQBHENT - DISPLAY BATCH ENTITY MENU (SELECT TO PRINT ALL ENTITIES OF A SPECIFIC KIND OR TO PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND)
MQBHMAIN - DISPLAY BATCH MAIN MENU (SELECT TO PRINT ALL ENTITIES IN THE MODEL OR TO PRINT ALL ENTITIES OF A SPECIFIC KIND)
MQCLMU - DISPLAY A LIST OF CONSTITUENTS FOR A SPECIFIC KIND
MQGETVAL - CONVERT ATTRIBUTE VALUE TO A STRING VALUE
MQGTDEFN - GET ENTITY DEFINITIONS OF A SPECIFIC KIND
MQIAATT - PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND
MQIAENT - DISPLAY INTERACTIVE ENTITY MENU
MQIAMAIN - DISPLAY MAIN INTERACTIVE MENU
MQNCLMU - DISPLAY A MENU INDICATING NO CONSTITUENTS FOR A SPECIFIC ENTITY.
MQNUSRMU - DISPLAY A MENU INDICATING NO USERS FOR A SPECIFIC ENTITY
MQUDVR - Mainline program for Model Query Utility.

MQUSRMU - DISPLAY LIST OF USERS FOR A SPECIFIC ENTITY
MREPORT - DISPLAYS THE REPORT MENU
MREVIEW - DISPLAYS THE REVIEW MENU
MUPDATE - DISPLAYS THE UPDATE MENU
NVRTVRS - RETRIEVE ENTITY DEFINITIONS FROM THE FILE
PHALFLD - PHYSICALIZE THE ATTRIBUTE OF AN ENTITY (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHBYFPOS - PHYSICALIZE THE ATTRIBUTES THAT SPECIFIED THE FIELD POSITION ORDER (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHDECBYT - TRANSLATE DECIMAL DIGIT PRECISION INTO BYTE PRECISION AND BUILD A LIST OF ATTRIBUTES OF AN ENTITY ACCORDING TO BOUNDARY ALIGNMENT.
PHENTITY - PHYSICALIZE THE ENTITY DEFINITIONS OF THE SUBSCHEMA (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHGLOBAL - PHYSICALIZE THE GLOBAL FIELDS OF THE SCHEMA (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHGTFLD - Determine boundary alignment of different data types in the field
PPHPOSITN - DETERMINE THE FIELD POSITION ORDER
PHSRTFLD - DETERMINE THE LOCATION OF ATTRIBUTES OF AN ENTITY ACCORDING TO BOUNDARY ALIGNMENT.
PHSRTORD - SORT ATTRIBUTES BY THE FIELD POSITION NUMBER
PHSUBTYP - PHYSICALIZE THE SUPER TYPES DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHWOFPOS - PHYSICALIZE THE ATTRIBUTES THAT DID NOT SPECIFIED THE FIELD POSITION NUMBER (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PHYSICAL - PHYSICALIZE THE ENTITY DEFINITIONS OF THE SUBSCHEMA (DETERMINE ATTRIBUTE SIZE AND LOCATION)
PSORDER - DETERMINE THE PHYSICAL SCHEMA ORDER OF ENTITY DEFINITION BY ITS OFFSET.
PSRABNDS - WRITE LOW-BOUND AND UPPER-BOUND FOR THE ARRAY ATTRIBUTE
PSRADB - WRITE THE BASIC RECORD OF AN ENTITY TO A PHYSICAL SCHEMA REPORT FILE
PSRARRAY - WRITE THE ARRAY ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE
PSRCL - WRITE THE CONSTITUENT REFERENCES OF AN ENTITY TO A SEQUENTIAL FILE
PSRENUM - WRITE THE ENUMERATION ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE
PSREPORT - FILE PHYSICAL SCHEMA REPEORT TO SEQUENTIAL FILE
PSRHEAD - WRITE THE PHYSICAL SCHEMA REPORT HEADING
PSRINDEX - WRITE THE TABLE OF CONTENTS FOR THE PHYSICAL SCHEMA REPORT
REARRAY - DISPLAYS THE REVIEW ARRAY PANEL.
RECLASS - DISPLAYS THE REVIEW CLASS PANEL
REDEFTYP - DISPLAYS THE DEFINED TYPE REVIEW PANEL
REENTIT - DISPLAYS THE REVIEW ENTITY MENU
REENUM - DISPLAYS THE REVIEW ENUMERATION MENU
REFIELD - DISPLAYS THE REVIEW FIELD PANEL
REFIELD1 - DISPLAYS THE REVIEW FIELD A MENU OR THE REVIEW FIELD B MENU
REFIELD2 - DISPLAYS THE REVIEW FIELD PANEL
REFSUP - Batch Interface routine that attempts to resolve a reference to a supertype.
REINTGR - DISPLAYS THE REVIEW INTEGER MENU
RELIST - DISPLAYS THE REVIEW LIST PANEL.
REPNTR - DISPLAYS THE REVIEW POINTER MENU

REREAL - DISPLAYS THE REVIEW REAL PANEL
RESET - DISPLAYS THE REVIEW SET PANEL.
RESTRING - DISPLAYS THE REVIEW STRING PANEL
RESTRUC - DISPLAYS THE REVIEW STRUCTURE PANEL
RESUBSCM - DISPLAYS THE REVIEW SUBSCHEMA PANEL
RESUPTYP - DISPLAYS THE REVIEW SUPERTYPE MENU
RSCPAI - COPY THE ARRAY INDEX TABLE INFORMATION INTO THE RUN-TIME SUBSCHEMA.
RSCPAT - COPY THE SIZE AND THE LOWER BOUND OF THE ARRAY INTO THE RUN-TIME SUBSCHEMA.
RSCPCI - COPY THE POINTER INDEX TABLE INFORMATION INTO THE RUN-TIME SUBSCHEMA.
RSCPCT - COPY THE KINDS OF POINTERS INTO THE RUN-TIME SUBSCHEMA.
RSCPEI - COPY THE ENUMERATION INDEX TABLE INFORMATION INTO THE RUN-TIME SUBSCHEMA.
RSCPET - COPY THE ENUMERATION VALUES INTO THE RUN-TIME SUBSCHEMA.
RSFILE - FILE RUN-TIME SUBSCHEMA INTO SEQUENTIAL FILE
RSGTSM - BUILD RUN-TIME SUBSCHEMA FROM SCHEMA MODEL
RSMASKND - INSERT THE MODEL ACCESS SOFTWARE (MAS) ATTRIBUTES (KIND, LENGTH, SYSUSE) INTO A RUN-TIME SUBSCHEMA.
RSTRGF - Translate global fields into a run-time subschema
RSTRSM - TRANSLATE A SCHEMA MODEL ENTRY INTO A RUN-TIME SUBSCHEMA ENTITY, ENUMERATION TABLE AND ARRAY INFO TABLE.
RSTRST - TRANSLATE SUPER TYPE INTO A RUN-TIME SUBSCHEMA
RS1100 - STORE ARRAY_ENTITY(1100) IN THE DATA DICTIONARY AND THE RUN-TIME SUBSCHEMA
SCALFSRT - IS THE ORDER FUNCTION CALLED BY MALSRT
SCARYCR - GATHERS THE DATA NECESSARY TO CREATE THE ARRAY ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCARYUP - GATHERS THE DATA TO UPDATE AN ARRAY.
SCBASIN - WRITES THE ENTITY KIND CONSTANTS TO THE PASCAL INCLUDE FILE.
SCCHRCK - CHECKS THAT THE CHARACTERS IN AN ENTITY NAME ARE VALID
SCCLSCR1 - GATHERS THE NAME AND KIND NUMBER TO BE ASSIGNED TO THE CLASS ENTITY TO BE CREATED.
SCCLSCR2 - GATHERS THE CONSTITUENTS OF THE CLASS ENTITY.
SCCLSUP - GATHERS THE DATA TO UPDATE THE CLASS ENTITY.
SCCOMPAR - THIS ROUTINE COMPARES TWO NAMES FOR THE LENGTH SPECIFIED BY 'UNIQUENESS_LENGTH' IN SCECON. THE VARIABLE 'DIFFERENT' IS SET TO TRUE IF THE TWO NAMES DIFFER.
SCCONIN - WRITES THE ENTITY KIND CONSTANTS TO THE PASCAL INCLUDE FILE.
SCCREATE - THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE CREATE OPTION CHOSEN.
SCDEFCR - GATHERS THE DATA NECESSARY TO CREATE THE DEFINED TYPE ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCDEFUP - GATHERS THE DATA TO UPDATE THE DEFINED TYPE ENTITY.
SCENMUP - GATHERS THE DATA TO UPDATE THE ENUMERATION ENTITY.
SCENTCR - GATHERS THE DATA NECESSARY TO MODEL THE ENTITY ENTITY.
SCENTIN - WRITES THE ENTITY TYPE DECLARATIONS TO THE PASCAL INCLUDE FILE.
SCENTUP - UPDATES THE ENTITY ENTITY.

SCENUCR - GATHERS THE DATA NECESSARY TO CREATE THE ENUMERITEM ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCEXIST - VERIFIES THE EXISTENCE OF AN ENTITY AND RETURNS THE ENTITY KEY IF IT DOES IN FACT EXIST.
SCFLDAD - GATHERS THE DATA TO ADD A FIELD TO AN ENTITY.
SCFLDCR - GATHERS THE DATA NECESSARY TO MODEL THE FIELD ENTITY.
SCFLDIN - WRITES THE FIELD TYPE DECLARATION TO THE PASCAL INCLUDE FILE.
SCFLDSRT - FINDS THE KEY AND RETURNS IT TO THE CALLING PROCEDURE GIVEN A NAME OR USER DEFINED KIND NUMBER AS WELL AS THE ENTITY KIND.
SCFLDST - SORTS THE FIELDS INTO TWO GROUPS AND THEN ORDERS THEM ACCORDING TO OFFSET OR POSITION DEPENDING ON WHETHER OR NOT THEY ARE IN THE LIST OF ADB FIELDS OR THE LIST OF CONSTITUENT FIELDS.
SCFLDUP - GATHERS THE DATA TO UPDATE THE FIELD ENTITY.
SCFNDKEY - FINDS THE KEY AND RETURNS IT TO THE CALLING PROCEDURE GIVEN A NAME OR USER DEFINED KIND NUMBER AS WELL AS THE ENTITY KIND.
SCGENRPT - DETERMINES THE SUBSCHEMA FOR WHICH A REPORT IS TO BE GENERATED AND CALLS THE APPROPRIATE ROUTINE TO PRODUCE THE REPORT.
SCHDVR - THIS IS THE MAINLINE PROGRAM WHICH DRIVES THE SCHEMA EXECUTIVE PACKAGE.
SCINCLD - GENERATES THE PASCAL INCLUDE FILES AND WRITES THESE DEFINITIONS TO A FILE
SCINTCR - GATHERS THE DATA NECESSARY TO CREATE THE INTEGER ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCINTUP - UPDATES THE INTEGER ENTITY.
SCKEFIND - GETS THE KEY TO THE ENTITY TO BE UPDATED OR REVIEWED.
SCKEYIN - WRITES THE KEYBLOCK TYPE DECLARATION TO THE PASCAL INCLUDE FILE.
SCLISTCR - THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE LIST ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCLISTUP - THIS ROUTINE GATHERS THE DATA TO UPDATE A LIST.
SCMASIN - WRITES THE MAS ADB DECLARATION TO THE PASCAL INCLUDE FILE.
SCMEMAD - DISPLAYS A LIST OF MEMBERS FROM WHICH THE USER CAN SELECT. THE ENTITY KEY OF THE MEMBER SELECTED IS RETURNED.
SCMEMLST - LISTS OUT THE CURRENT MEMBERS/CONSTITUENTS OF THE ENTITY, CLASS, SUBSCHEMA, POINTER, STRUCTURE, GLOBAL FIELD AND ENUMERATION ENTITIES.
SCPOPKE - 'POPS' A KEY OFF OF THE KEY STACK.
SCPOPTR - 'POPS' THE TRANSACTION DATA OFF OF THE DYNAMICALLY ALLOCATED STACK.
SCPRMFL - FILLS AN ARRAY WITH THE NAMES OF THE ENTITIES IN THE LIST OF ENTITIES.
SCPRMRE - GATHERS THE DATA TO REVIEW THE ENTITIES AND CALLS THE MENU INTERFACE ROUTINES TO DISPLAY THIS DATA.
SCPTRCR - GATHERS THE DATA NECESSARY TO CREATE THE POINTER ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCPTRUP - GATHERS THE DATA TO UPDATE THE POINTER ENTITY.
SCPUSHKE - 'PUSHES' THE KEY DATA ON TO THE DYNAMICALLY ALLOCATED KEY STACK.
SCPUSHTR - 'PUSHES' THE TRANSACTION DATA ON TO THE DYNAMICALLY ALLOCATED STACK.
SCRELCR - GATHERS THE DATA NECESSARY TO CREATE THE REAL ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCRELUP - UPDATES THE REAL ENTITY.

SCREVIEW - THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE EDIT OPTION CHOSEN.
SCRPTM - DETERMINES THE REPORT OPTION SELECTED AND CALLS THE ROUTINE WHICH GENERATES THE REPORT.
SCSETCR - THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE SET ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCSETUP - THIS ROUTINE GATHERS THE DATA TO UPDATE A SET.
SCSTCUP - GATHERS THE DATA TO UPDATE THE STRUCTURE ENTITY.
SCSTGCR - GATHERS THE DATA NECESSARY TO CREATE THE STRING ENTITY AND PUSHES THE DATA ON THE TRANSACTION STACK.
SCSTGUP - UPDATES THE STRING ENTITY.
SCSUBCR - GATHERS THE DATA NEEDED TO MODEL THE SUBSCHEMA ENTITY.
SCSUBUP - GATHERS THE DATA TO UPDATE THE SUBSCHEMA ENTITY.
SCSUPCR - THIS ROUTINE GATHERS THE DATA NECESSARY TO MODEL THE SUPERTYPE ENTITY.
SCSUPUP - THIS ROUTINE UPDATES THE SUPERTYPE ENTITY.
SCTRSPR - BEGINS THE PROCESSING OF THE TRANSACTION STACK.
SCTYPIN - WRITES THE DEFINED TYPE DECLARATIONS TO THE PASCAL INCLUDE FILE.
SCTYPUP - UPDATES THE DEFINED TYPE, ARRAY, OR FIELD ENTITY'S TYPE.
SCUNIQUE - VERIFIES THE UNIQUENESS OF NAMES WITHIN THE SCHEMA FOR THE SUBSCHEMA, CLASS, ENTITY, GLOBAL FIELD, FIELD, ENUMERITEM, AND DEFINED TYPE ENTITIES AS WELL AS THE USER DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES.
SCUNQEST - CALLS MAKXEQ TO EXECUTE THE PROCEDURE SCUNIQUE WHICH VERIFIES THE UNIQUENESS OF NAMES WITHIN THE SCHEMA FOR THE SUBSCHEMA, CLASS, ENTITY, GLOBAL FIELD, FIELD, ENUMERITEM, AND DEFINED TYPE ENTITIES AS WELL AS THE USER DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES.
SCUNQPND - VERIFIES THE UNIQUENESS OF NAMES AND USER DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES, BY SEARCHING THE TRANSACTION STACK FOR THE CLASS, ENTITY, FIELD, ENUMERITEM, AND DEFINED TYPE ENTITIES.
SCUPDATE - THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE UPDATE OPTION CHOSEN.
SORTKIND - THE ORDER FUNCTION CALLED BY MALSRT
SORTNAME - THE ORDER FUNCTION CALLED BY MALSRT
UPARRAY - DISPLAYS THE UPDATE ARRAY MENU
UPCLASS1 - DISPLAYS THE UPDATE CLASS MENU 1
UPCLASS2 - DISPLAYS THE UPDATE CLASS MENU 2
UPDEFTYP - DISPLAYS THE UPDATE DEFINED TYPE MENU
UPENTY1 - DISPLAYS THE UPDATE ENTITY MENU 1
UPENTY2 - DISPLAYS THE UPDATE ENTITY MENU 2
UPENUM - DISPLAYS THE UPDATE ENUMERATION MENU
UPFIELD - DISPLAYS THE UPDATE FIELD PANEL
UPINT - DISPLAYS THE UPDATE INTEGER MENU
UPLIST - DISPLAYS THE UPDATE LIST MENU
UPPNTR - DISPLAYS THE UPDATE POINTER MENU
UPREAL - DISPLAYS THE UPDATE REAL MENU
UPSET - DISPLAYS THE UPDATE SET MENU
UPSTRING - DISPLAYS THE UPDATE STRING MENU
UPSTRUC - DISPLAYS THE UPDATE STRUCTURE MENU

UPSUB1 - DISPLAYS THE UPDATE SUBSCHEMA PANEL 1
UPSUB2 - DISPLAYS THE REVIEW SUBSCHEMA PANEL 2
UPSUPER - DISPLAYS THE UPDATE SUPERTYPE MENU
XATTDATA - THIS ROUTINE GENERATES A LIST OF ALL "ENTITIES" CONTAINING A PARTICULAR "ENTITY."
XATTNAME - THIS ROUTINE GENERATES A LIST OF ALL ENTITIES HAVING AN ATTRIBUTE WITH A SPECIFIED NAME.
XATTRES - DISPLAYS CROSS REFERENCE REPORT RESULTS
XEXPREC - THIS ROUTINE GENERATES A LIST OF ALL EXISTING PRECISIONS FOR THE INTEGER, REAL, OR STRING DATA TYPE.
XFNDARY - THIS ROUTINE FINDS ALL ARRAY DATA TYPES OF SET, LIST, OR ARRAY AND PUTS THEM ON A LIST.
XFNDKEY - THIS ROUTINE DETERMINES THE KEY FOR A GIVEN ENTITY NAME OR NUMBER
XFNDNAME - THIS ROUTINE FINDS ALL ATTRIBUTES WITH THE GIVEN NAME AND PUTS THEM ON A LIST.
XFNDPREC - THIS ROUTINE FINDS ALL INTEGERS, REALS, OR STRINGS WITH THE SPECIFIED PRECISION AND PUTS THEM ON A LIST.
XLISTENT - THIS ROUTINE GENERATES A LIST OF ALL "ENTITIES" CONTAINING A PARTICULAR ENTITY.
XMAIN - THIS ROUTINE DETERMINES THE CROSS REFERENCE OPTION DESIRED
XMATTRES - DISPLAYS A CROSS REFERENCE MENU
XMMAIN - DISPLAYS A CROSS REFERENCE MENU
XMNAMSPE - DISPLAYS A CROSS REFERENCE REPORT MENU
XMPRESPE - DISPLAYS A CROSS REFERENCE REPORT MENU
XMRESULT - DISPLAYS A CROSS REFERENCE MENU
XMTYPSPE - DISPLAYS A CROSS REFERENCE MENU
XNAMENUM - THIS ROUTINE DETERMINES IF A CHARACTER STRING IS A NAME OR NUMBER
XRESULT - THIS ROUTINE DISPLAYS THE CROSS REFERENCE REPORT RESULTS

ROUTINE DICTIONARY

```
(* %INCLUDE ADDENUM *)
(**)
  PROCEDURE ADDENUM(VAR MESS     : MESSAGE;
                    VAR NAME     : CHAR16;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      THIS FUNCTION:                                                    *)
(*                      DISPLAYS THE ADD ENUMERATION MENU                 *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL   *)
(*        NAME            O   THE NAME OF THE ENUMERATION FROM PANEL     *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT        *)
(*                            OPERATION                                   *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,    *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN     *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE CREATE ENUMERATION PANEL (ADDENUM) BY MAKING        *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN          *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION        *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING           *)
(*        PROCEDURE.                                                      *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE ADDFIELD *)
(**)
  PROCEDURE ADDFIELD(VAR MESS    : MESSAGE;
                     VAR NAME    : T_NAME;
                     VAR POS     : CHAR8;
                     VAR PURP    : CHAR8;
                     VAR REQD    : CHAR8;
                     VAR DEPD    : CHAR8;
                     VAR FTYPE   : ENTITY_TYPE;
                     VAR FLDTYP  : T_FIELDTYPE;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                        DISPLAYS THE ADDFIELD PANEL                *)
(*                                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME            O   THE NAME OF THE FIELD                  *)
(*        PURP            O   THE PURPOSE OF THE FIELD               *)
(*        REQD            O   REQUIREDNESS OF THE FIELD              *)
(*        DEPD            O   DEPENDENCE OF THE FIELD                *)
(*        FTYPE           O   ENTITY TYPE                            *)
(*        FLDTYPE         O   THE FIELD TYPE                         *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                            OPERATION                              *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE ADD FIELD PANEL (ADDFIELD) BY MAKING ISPLNK       *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED    *)
(*        TYPE.                                                         *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*        NONE                                                          *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
```

```
(* %INCLUDE APPROVE *)
(**)

PROCEDURE APPROVE(VAR MESS     : MESSAGE;
                  VAR NAME     : T_NAME;
                  VAR NEXT_OP  : OPERATIONS;
                  VAR RR       : RET_REC);

SUBPROGRAM;

(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS PROCEDURE :                                                *)
(*                         DISPLAYS THE APPROVE UPDATE PANEL              *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL   *)
(*        NAME            I   THE NAME OF THE ENTITY                      *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT         *)
(*                            OPERATION                                   *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,     *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN     *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE APPROVE UPDATE PANEL (APPROVE) BY MAKING ISPLNK     *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED      *)
(*        TYPE.                                                           *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE BATDVR *)
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        THIS IS THE MAINLINE PROGRAM WHICH DRIVES THE BATCH       *)
(*        INTERFACE OF THE SCHEMA MANAGER.                          *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NONE                                                      *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*        NONE                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*        DDNAMES USED WITH STANDARD FILES:                         *)
(*             SOURCE  = INPUT FILE OF SCHEMA DEFINITIONS IN THE    *)
(*                       EXPRESS INFORMATION MODELING LANGUAGE      *)
(*                                                                  *)
(*             REPORT1 = OUTPUT FILE OF ACTIONS TAKEN DURING THE    *)
(*                       BATCH PROCESSING.                          *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*                                                                  *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                  *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*     NARATION ON THE NEXT LINE.                                   *)
(*                                                                  *)
(*    ORIGINATED:                C. H. MOHME               DBMA     *)
(*                                                                  *)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*END---------------------------------------------------------------*)
(* END %INCLUDE BATDVR *)
```

```
(* BEGIN %INCLUDE BATERR ************************************************)
(*                                                                       *)
PROCEDURE BATERR ( Const Expected         : T_Expected;
                   Var   Ent_Kind         : Integer;
                   Var   Token            : T_Token;
                   Var   Token_Value      : T_Token_Value;
                   Var   Token_Location   : Integer;
                   Var   Token_Length     : Integer;
                   Var   Report1          : Text );
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        Error reporting and recovery for Batch Interface.              *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME               I/O  DESCRIPTION                            *)
(*        ====               ===  ===========                            *)
(*        Ent_Kind            I   Kind of entity being constructed       *)
(*        Expected            I   Record containing:                     *)
(*                                   Entries:     Number of entries      *)
(*                                                 in array              *)
(*                                   Token_Value: Array of expected      *)
(*                                                 token values          *)
(*        Report1             O   Error messages                         *)
(*        Token              I/O  Input:  unrecognized token             *)
(*                                Output: first recognized token         *)
(*        Token_Length       I/O  Input:  length of unrecognized token   *)
(*                                Output: length of recognized token     *)
(*        Token_Location     I/O  Input:  location of unrecognized       *)
(*                                       token                           *)
(*                                Output: location of recognized token   *)
(*        Token_Value        I/O  Input:  value of unrecognized token    *)
(*                                Output: value of recognized token      *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL/VS SEGMENT                                *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        Internal routine for Batch Interface of the Schema Manager     *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        Highlight the unrecognized token.                              *)
(*        List the expected tokens.                                      *)
(*        Highlight the tokens ignored while searching for a             *)
(*           recognized token.                                           *)
(*                                                                       *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 7 December 1987, G. A. White                       *)
(*        REVISED:    <date, responsible person, reason/description>     *)
(*                                                                       *)
(* END %INCLUDE BATERR ********************************************)
```

```
(* %INCLUDE BATRPT *)
(**)
  PROCEDURE BATRPT(VAR IRC          :  RET_REC;
                   VAR TOKEN        :  T_TOKEN;
                   VAR TOKEN_VALUE  :  T_TOKEN_VALUE;
                   VAR REPORT1      :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that invokes the necessary routines*)
(*         to generate a specified report.                           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
         TOKEN          I/O  TOKEN FROM BATCH INPUT                *)
(*        TOKEN_VALUE    I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*        TEXT           I/O  OUTPUT FILE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    PERFORM INITIALIZATIONS                                        *)
(*    GET THE REPORT TYPE                                            *)
(*    IF THE REPORT TYPE IS NOT CONCEPTUAL SCHEMA, THEN GET THE      *)
(*      SUBSCHEMA NAME.                                              *)
(*      GET THE SUBSCHEMA KEY AS APPROPRIATE AND GENERATE A REPORT   *)
(*      GET THE SUBSCHEMA KEY                                        *)
(*      PHYSICALIZE THE SUBSCHEMA, IF NECESSARY                      *)
(*      IF REPORT TYPE IS PASCAL INCLUDES, THEN GENERATE THE PASCAL  *)
(*        INCLUDE FILES.                                             *)
(*      IF REPORT TYPE IS DATA DICTIONAY, THEN GENERATE THE DATA     *)
(*        DICTIONARY.                                                *)
(*      IF REPORT TYPE IS PHYSICAL SUBSCHEMA, THEN GENERATE          *)
(*        THE PHYSICAL SUBSCHEMA REPORT.                             *)
(*      PRINT ERROR MESSAGES AS APPROPRIATE                          *)
```

```
(*    IF REPORT TYPE IS CONCEPTUAL SCHEMA, THEN GENERATE THE          *)
(*      CONCEPTUAL SCHEMA REPORT                                      *)
(*    PRINT ERROR MESSAGES AS APPROPRIATE                             *)
(*    GET THE NEXT TOKEN                                              *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 06/09/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE BATRPT *)
```

```
(* %INCLUDE BCSMAIN *)
(**)
  PROCEDURE BCSMAIN(VAR IRC      : RET_REC);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE SERVES AS THE MAIN DRIVER FOR THE CONCEPTUAL  *)
(*        SCHEMA REPORT.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   INITIALIZE THE VARIABLES USED WITHIN THIS ROUTINE.              *)
(*   WRITE OUT TO THE FILE THE REPORT COVER.                         *)
(*   INITIALIZE THE PAGE NUMBER AND PAGE CHAIN.                      *)
(*   MAKE A LIST OF DEFINED TYPES WITHIN THE SCHEMA.                 *)
(*   ALPHABETIZE THE LIST OF DEFINED TYPES.                          *)
(*   WRITE OUT TO THE FILE THE DEFINED TYPES WITHIN THE SCHEMA.      *)
(*   DELETE THE LIST OF DEFINED TYPES WITHIN THE SCHEMA.             *)
(*   MAKE A LIST OF GLOBAL FIELDS WITHIN THE SCHEMA.                 *)
(*   WRITE OUT TO THE FILE THE GLOBAL FIELDS WITHIN THE SCHEMA.      *)
(*   DELETE THE LIST OF GLOBAL FIELDS WITHIN THE SCHEMA.             *)
(*   MAKE A LIST OF ENTITIES WITHIN THE SCHEMA.                      *)
(*   ALPHABETIZE THE LIST OF ENTITIES.                               *)
(*   WRITE OUT TO THE FILE THE ENTITIES WITHIN THE SCHEMA.           *)
(*   MAKE A LIST OF CLASSES WITHIN THE SCHEMA.                       *)
(*   ALPHABETIZE THE LIST OF CLASSES.                                *)
(*   WRITE OUT TO THE FILE THE CLASSES WITHIN THE SCHEMA.            *)
(*   MAKE A LIST OF SUBSCHEMAS WITHIN THE SCHEMA.                    *)
(*   ALPHABETIZE THE LIST OF SUBSCHEMAS.                             *)
(*   WRITE OUT TO THE FILE THE SUBSCHEMAS WITHIN THE SCHEMA.         *)
(*   WRITE OUT TO THE FILE THE ENTITY INDEX.                         *)
```

```
(*   DELETE THE LIST OF ENTITIES.                                          *)
(*   WRITE OUT TO THE FILE THE CLASS INDEX.                                *)
(*   DELETE THE LIST OF CLASSES.                                           *)
(*   WRITE OUT TO THE FILE THE SUBSCHEMA INDEX.                            *)
(*   DELETE THE LIST OF SUBSCHEMAS.                                        *)
(*   IF NO MODEL EXISTS, WRITE APPROPRIATE MESSAGE.                        *)
(*   DISPOSE OF POINTERS USED IN THIS ROUTINE.                             *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*     WITHIN THE INCLUDE FILE 'CSTYPCON', ONE CAN SET THE MAXIMUM         *)
(*   NUMBER OF LINES PER PAGE IN THE REPORT.                               *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID     *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                  *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID     *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE               *)
(*     NARATION ON THE NEXT LINE.                                          *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCXX     I. M. APERSON               GROUP_ID     *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                                   *)
(*                                                                         *)
(*    ORIGINATED: 10/27/86       C. H. MOHME                 DBMA         *)
(*                                                                         *)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE BCSMAIN *)
```

```
(* %INCLUDE BLDARRAY *)
(**)
  PROCEDURE BLDARRAY(VAR IRC : RET_REC;
                     CONST ARRAY_DATA : TRANSACTION;
                     CONST CNST_KEY : ENTKEY;
                     VAR ARRAY_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE ARRAY ENTITY.                            *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                                *)
(*        ====             ===  ===========                                *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        ARRAY_DATA        I   TRANSACTION DATA OF THE ARRAY ENTITY      *)
(*        CNST_KEY          I   KEY TO THE CONSTITUENT                    *)
(*        ARRAY_KEY         O   KEY TO THE ARRAY ENTITY                   *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        A LIST IS MADE OF ALL THE ARRAY ENTITIES.  IF THE LIST IS        *)
(*        NOT EMPTY THEN IT IS SEARCHED FOR A MATCH.  IF THE BOUNDS        *)
(*        ARE IDENTICAL THE CONSTITUENT KEYS ARE COMPARED.  IF THESE       *)
(*        MATCH THE KEY TO THE ARRAY ENTITY THAT IS IDENTICAL TO THE       *)
(*        ONE TO BE CREATED IS PASSED BACK TO THE CALLING PROCEDURE.       *)
(*        THE LIST IS SEARCHED UNTIL A MATCH IS MADE OR THE END OF         *)
(*        THE LIST IS ENCOUNTERED.  IF NO MATCH IS FOUND THE ARRAY         *)
(*        ENTITY IS CREATED USING MAS.                                     *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDBPDEF *)
(**)
  PROCEDURE BLDBPDEF(VAR IRC            : RET_REC;
                     VAR IDENTIFIER     : T_NAME;
                     VAR KIND           : INTEGER;
                     VAR REFERENCE_KEY  : ENTKEY;
                     VAR DEFINITION_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE BACKPATCH ENTITY.                        *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                               *)
(*        ====             ===  ===========                               *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        IDENTIFIER        I   ENTITY NAME                               *)
(*        KIND              I   ENTITY KIND NUMBER                        *)
(*        REFERENCE_KEY     I   THE KEY TO THE ENTITY THAT REFERENCES     *)
(*                              THE UNRESOLVED ENTITY                      *)
(*        DEFINITION_KEY    O   THE KEY TO THE CREATED BACKPATCH          *)
(*                              ENTITY                                     *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THE BACKPATCH ENTITY IS CREATED USING MAS ROUTINES               *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID      *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                  *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID      *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE               *)
(*     NARATION ON THE NEXT LINE.                                          *)
(*                                                                         *)
```

```
(*    REVISED: MM/DD/YY CCXX      I. M. APERSON                GROUP_ID *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                                 *)
(*                                                                       *)
(*    ORIGINATED: 04/22/87        C. H. MOHME                  DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE BLDBPDEF *)
```

```
(* %INCLUDE BLDCLASS *)
(**)
  PROCEDURE BLDCLASS(VAR IRC : RET_REC;
                     CONST CLASS_DATA : TRANSACTION;
                     VAR CLASS_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE CLASS ENTITY.                            *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                                *)
(*        ====             ===  ===========                                *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        CLASS_DATA        I   TRANSACTION DATA OF THE CLASS ENTITY      *)
(*        CLASS_KEY         O   KEY TO THE CLASS ENTITY                   *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      REF                                                                *)
(*        CURRENT_LIST     I/O  LIST CURRENTLY IN USE CONTAINING THE       *)
(*                              CONSTITUENTS OF THE CLASS ENTITY           *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS               *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE CLASS ENTITY.              *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDCLS *)
(**)
  PROCEDURE BLDCLS(VAR IRC          :  RET_REC;
                   VAR TRANS_STACK  :  TRANSPTR;
                   VAR CLASS_FLAG   :  BOOLEAN;
                   VAR CLS_ENT_HEAD :  ENTITY_LIST_PTR);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        Batch Interface routine that creates a class entity in the *)
(*        schema model from a class name, kind number, and a list of *)
(*        constituents.                                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        IRC            O   INTERNAL RETURN CODE                    *)
(*        TRANS_STACK   I/O  TRANSACTION STACK                       *)
(*        CLASS_FLAG    I/O  INDICATES IF ENTITITIES AND CLASSES ARE *)
(*                           DEFINED WITHIN THE CLASS                *)
(*        CLS_ENT_HEAD  I/O  POINTER TO LIST OF CLASS ENTITIES       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    INITIALIZE VARIABLES                                           *)
(*    REMOVE CLASS NAME AND NUMBER FROM LIST AND PUSH THEM ONTO      *)
(*      TRANSACTION STACK                                            *)
(*    REMOVE EACH ENTITY KEY FROM THE LIST AND PUSH THEM ONTO TRANS- *)
(*      ACTION STACK                                                 *)
(*    PUSH FINAL CLASS TRANSACTION ONTO THE STACK AND PROCESS THE    *)
(*      STACK.                                                       *)
(*    REINITIALIZE VARIABLES                                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA   *)
```

```
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE BLDCLS *)
```

```
(* %INCLUDE BLDDFTYP *)
(**)
  PROCEDURE BLDDFTYP(VAR IRC : RET_REC;
                     CONST DEFINED_TYPE_DATA : TRANSACTION;
                     CONST CNST_KEY : ENTKEY;
                     VAR DEFINED_TYPE_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE DEFINED TYPE ENTITY.                     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME                I/O  DESCRIPTION                             *)
(*        ====                ===  ===========                             *)
(*        IRC                  O   INTERNAL RETURN CODE                   *)
(*        DEFINED_TYPE_DATA    I   TRANSACTION DATA OF THE DEFINED        *)
(*                                 TYPE ENTITY                             *)
(*        CNST_KEY             I   KEY TO THE CONSTITUENT                 *)
(*        DEFINED_TYPE_KEY     O   KEY TO THE DEFINED TYPE ENTITY         *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THE ADB DATA IS ASSIGNED AND THEN THE MAS ROUTINE MAECR          *)
(*        IS CALLED TO MODEL THE DEFINED TYPE ENTITY.                      *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDEITEM *)
(**)
  PROCEDURE BLDEITEM(VAR IRC : RET_REC;
                     CONST ENUMERITEM_DATA : TRANSACTION;
                     VAR ENUMERITEM_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE ENUMERITEM ENTITY.                       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                                *)
(*        ====             ===  ===========                                *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        ENUMERITEM_DATA   I   TRANSACTION DATA OF THE ENUMERITEM        *)
(*                              ENTITY                                     *)
(*        ENUMERITEM_KEY    O   KEY TO THE ENUMERITEM ENTITY              *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        A LIST IS MADE OF ALL THE ENUMERITEM ENTITIES.                   *)
(*        IF THE LIST IS NOT EMPTY THEN IT IS SEARCHED FOR A MATCH         *)
(*        UNTIL ONE IS FOUND OR THE END OF THE LIST IS REACHED.            *)
(*        IF A MATCH IS FOUND ITS KEY IS PASSED BACK TO THE CALLING        *)
(*        PROCEDURE OTHERWISE A NEW ENUMERITEM ENTITY IS CREATED.          *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDENT *)
(**)
  PROCEDURE BLDENT(VAR IRC : RET_REC;
                   CONST ENTITY_DATA : TRANSACTION;
                   VAR ENTITY_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE ENTITY ENTITY.                           *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                               *)
(*        ====             ===  ===========                               *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        ENTITY_DATA       I   TRANSACTION DATA OF THE ENTITY ENTITY     *)
(*        ENTITY_KEY        O   KEY TO THE ENTITY ENTITY                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      REF                                                                *)
(*        CURRENT_LIST    I/O  LIST CURRENTLY IN USE CONTAINING THE        *)
(*                             CONSTITUENTS OF THE ENTITY ENTITY           *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS               *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE ENTITY ENTITY.             *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDENUMR *)
(**)
  PROCEDURE BLDENUMR(VAR IRC : RET_REC;
                     VAR ENUMERATION_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE BUILDS THE ENUMERATION ENTITY.               *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME             I/O  DESCRIPTION                         *)
(*         ====             ===  ===========                         *)
(*         IRC               O   INTERNAL RETURN CODE                *)
(*         ENUMERATION_KEY   O   KEY TO THE ENUMERATION ENTITY       *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         CURRENT_LIST     I/O  A KEY TO THE LIST OF CONSTITUENTS   *)
(*                                                                   *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THE ADB DATA IS ASSIGNED, AND THEN THE KEY TO THE COMMON  *)
(*         CURRENT_LIST, WHICH CONTAINS THE ENUMERATION CONSTITUENTS, *)
(*         AS WELL AS THE ADB IS PASSED TO THE MAS ROUTINE MAECR     *)
(*         WHICH MODELS THE ENUMERATION ENTITY.                      *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE BLDFIELD *)
(**)
  PROCEDURE BLDFIELD(VAR IRC : RET_REC;
                     CONST FIELD_DATA : TRANSACTION;
                     CONST CNST_KEY : ENTKEY;
                     VAR FIELD_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE BUILDS THE FIELD ENTITY.                            *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME             I/O  DESCRIPTION                                *)
(*        ====             ===  ===========                                *)
(*        IRC               O   INTERNAL RETURN CODE                      *)
(*        FIELD_DATA        I   TRANSACTION DATA OF THE FIELD ENTITY      *)
(*        CNST_KEY          I   KEY TO THE CONSTITUENT                    *)
(*        FIELD_KEY         O   KEY TO THE FIELD ENTITY                   *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THE ADB DATA IS ASSIGNED AND THEN THE MAS ROUTINE MAECR          *)
(*        IS CALLED TO MODEL THE FIELD ENTITY.                             *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE BLDGBLFD *)
(**)
  PROCEDURE BLDGBLFD(VAR   IRC              : RET_REC;
                     CONST GLOBAL_DATA      : TRANSACTION;
                     VAR   GLOBAL_FIELD_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE BUILDS THE GLOBAL FIELD ENTITY.                    *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME              I/O  DESCRIPTION                              *)
(*        ====              ===  ===========                              *)
(*        IRC                O   INTERNAL RETURN CODE                    *)
(*        GLOBAL_DATA        I   GLOBAL FIELD DATA                       *)
(*        GLOBAL_FIELD_KEY   O   KEY TO THE GLOBAL FIELD ENTITY          *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      REF                                                               *)
(*        CURRENT_LIST      I/O  LIST CURRENTLY IN USE CONTAINING THE     *)
(*                               CONSTITUENTS OF THE GLOBAL FIELD         *)
(*                               ENTITY                                   *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS              *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE GLOBAL FIELD              *)
(*        ENTITY.                                                         *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE BLDINT *)
(**)
  PROCEDURE BLDINT(VAR IRC : RET_REC;
                   CONST INTEGER_DATA : TRANSACTION;
                   VAR INTEGER_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE INTEGER ENTITY.                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
(*        INTEGER_DATA    I   TRANSACTION DATA OF THE INTEGER ENTITY *)
(*        INTEGER_KEY     O   KEY TO THE INTEGER ENTITY             *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        A LIST IS MADE OF ALL THE INTEGER ENTITIES.                *)
(*        IF THE LIST IS NOT EMPTY THEN IT IS SEARCHED FOR A MATCH   *)
(*        UNTIL ONE IS FOUND OR THE END OF THE LIST IS REACHED.      *)
(*        IF A MATCH IS FOUND ITS KEY IS PASSED BACK TO THE CALLING  *)
(*        PROCEDURE OTHERWISE A NEW INTEGER ENTITY IS CREATED.       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE BLDLOG *)
(**)
  PROCEDURE BLDLOG(VAR IRC : RET_REC;
                   VAR LOGICAL_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE LOGICAL ENTITY.                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
(*        LOGICAL_KEY     O   KEY TO THE LOGICAL ENTITY             *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        A LIST IS MADE OF ALL THE LOGICAL ENTITIES.                *)
(*        IF THE LIST IS NOT EMPTY THEN A LOGICAL ENTITY IS CREATED, *)
(*        OTHERWISE THE KEY TO THE EXISTING LOGICAL ENTITY IS        *)
(*        PASSED BACK TO THE CALLING PROCEDURE.                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE BLDPNTR *)
(**)
  PROCEDURE BLDPNTR(VAR IRC : RET_REC;
                    VAR POINTER_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE POINTER ENTITY.                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME             I/O  DESCRIPTION                          *)
(*        ====             ===  ===========                          *)
(*        IRC               O   INTERNAL RETURN CODE                 *)
(*        POINTER_KEY       O   KEY TO THE POINTER ENTITY            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST     I/O  LIST CURRENTLY IN USE CONTAINING THE  *)
(*                              CONSTITUENTS OF THE POINTER ENTITY   *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS         *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE POINTER ENTITY.      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE BLDREAL *)
(**)
  PROCEDURE BLDREAL(VAR IRC : RET_REC;
                    CONST REAL_DATA : TRANSACTION;
                    VAR REAL_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE BUILDS THE REAL ENTITY.                      *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   INTERNAL RETURN CODE                 *)
(*         REAL_DATA       I   TRANSACTION DATA OF THE REAL ENTITY  *)
(*         REAL_KEY        O   KEY TO THE REAL ENTITY               *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         A LIST IS MADE OF ALL THE REAL ENTITIES.                  *)
(*         IF THE LIST IS NOT EMPTY THEN IT IS SEARCHED FOR A MATCH  *)
(*         UNTIL ONE IS FOUND OR THE END OF THE LIST IS REACHED.     *)
(*         IF A MATCH IS FOUND ITS KEY IS PASSED BACK TO THE CALLING *)
(*         PROCEDURE OTHERWISE A NEW REAL ENTITY IS CREATED.         *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE BLDSSCMA *)
(**)
  PROCEDURE BLDSSCMA(VAR IRC : RET_REC;
                     CONST SUBSCHEMA_DATA : TRANSACTION;
                     VAR SUBSCHEMA_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE SUBSCHEMA ENTITY.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME             I/O  DESCRIPTION                          *)
(*        ====             ===  ===========                          *)
(*        IRC               O   INTERNAL RETURN CODE                 *)
(*        SUBSCHEMA_DATA    I   TRANSACTION DATA OF THE SUBSCHEMA    *)
(*                              ENTITY                               *)
(*        SUBSCHEMA_KEY     O   KEY TO THE SUBSCHEMA ENTITY          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST     I/O  LIST CURRENTLY IN USE CONTAINING THE *)
(*                              CONSTITUENTS OF THE SUBSCHEMA ENTITY *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS         *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE SUBSCHEMA ENTITY.    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE BLDSTRNG *)
(**)
  PROCEDURE BLDSTRNG(VAR IRC : RET_REC;
                     CONST STRING_DATA : TRANSACTION;
                     VAR STRING_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE BUILDS THE STRING ENTITY.                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   INTERNAL RETURN CODE                 *)
(*         STRING_DATA     I   TRANSACTION DATA OF THE STRING ENTITY *)
(*         STRING_KEY      O   KEY TO THE STRING ENTITY              *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         A LIST IS MADE OF ALL THE STRING ENTITIES.                *)
(*         IF THE LIST IS NOT EMPTY THEN IT IS SEARCHED FOR A MATCH  *)
(*         UNTIL ONE IS FOUND OR THE END OF THE LIST IS REACHED.     *)
(*         IF A MATCH IS FOUND ITS KEY IS PASSED BACK TO THE CALLING *)
(*         PROCEDURE OTHERWISE A NEW STRING ENTITY IS CREATED.       *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE BLDSTRUC *)
(**)
  PROCEDURE BLDSTRUC(VAR IRC : RET_REC;
                     VAR STRUCTURE_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE STRUCTURE ENTITY.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME             I/O  DESCRIPTION                          *)
(*        ====             ===  ===========                          *)
(*        IRC               O   INTERNAL RETURN CODE                 *)
(*        STRUCTURE_KEY     O   KEY TO THE STRUCTURE ENTITY          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST     I/O  LIST CURRENTLY IN USE CONTAINING THE  *)
(*                              CONSTITUENTS OF THE STRUCTURE ENTITY *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS         *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE STRUCTURE ENTITY.    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE BLDSUB *)
(**)
  PROCEDURE BLDSUB(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*          Batch Interface routine that creates a subschema entity in *)
(*          the schema model from a subschema name and list of       *)
(*          constituents.                                            *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME            I/O  DESCRIPTION                           *)
(*        ====            ===  ===========                           *)
(*        IRC              O   INTERNAL RETURN CODE                  *)
(*        TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*        SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES AND CLASSES ARE *)
(*                             DEFINED WITHIN THE SUBSCHEMA          *)
(*        SUB_ENT_HEAD    I/O  POINTER TO LIST OF SUBSCHEMA ENTITIES *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*                                                                   *)
(*     INITIALIZE VARIABLES                                          *)
(*     REMOVE SUBSCHEMA NAME FROM LIST AND PUSH IT ONTO TRANSACTION  *)
(*       STACK                                                       *)
(*     REMOVE EACH ENTITY KEY FROM THE LIST AND PUSH THEM ONTO TRANS *)
(*       ACTION STACK                                                *)
(*     PUSH FINAL SUBSCHEMA TRANSACTION ONTO THE STACK AND PROCESS   *)
(*       THE STACK                                                   *)
(*     REINITIALIZE THE VARIABLES                                    *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     ORIGINATED: 03/20/87       C. H. MOHME                 DBMA  *)
```

```
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE BLDSUB *)
```

```
(* %INCLUDE BLDSUPER *)
(**)
  PROCEDURE BLDSUPER(VAR   IRC : RET_REC;
                     CONST SUPERTYPE_DATA : TRANSACTION;
                     VAR   SUPERTYPE_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BUILDS THE SUPERTYPE ENTITY.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME             I/O  DESCRIPTION                          *)
(*        ====             ===  ===========                          *)
(*        IRC               O   INTERNAL RETURN CODE                 *)
(*        SUPERTYPE_DATA    I   TRANSACTION DATA OF THE SUPERTYPE    *)
(*                              ENTITY                               *)
(*        SUPERTYPE_KEY     O   KEY TO THE SUPERTYPE ENTITY          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST    I/O  LIST CURRENTLY IN USE CONTAINING THE  *)
(*                             CONSTITUENTS OF THE ENTITY ENTITY     *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE ADB DATA IS ASSIGNED TO VARIABLES THEN THE MAS         *)
(*        ROUTINE MAECR IS CALLED TO CREATE THE ENTITY ENTITY.       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE         *)
(*     NARATION ON THE NEXT LINE.                                    *)
(*                                                                   *)
```

```
(*    REVISED: MM/DD/YY CCXX      I. M. APERSON                GROUP_ID *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                                 *)
(*                                                                       *)
(*    ORIGINATED: 09/28/87        C. H. MOHME                  DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE BLDSUPER *)
```

```
(* %INCLUDE BLDUNRES *)
(**)
  PROCEDURE BLDUNRES(VAR IRC : RET_REC;
                     VAR UNRESOLVED_DATA : TRANSACTION;
                     VAR UNRESOLVED_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE BUILDS THE UNRESOLVED ENTITY.                *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                           *)
(*         ====           ===  ===========                           *)
(*        IRC              O   INTERNAL RETURN CODE                  *)
(*        UNRESOLVED_DATA  O   TRANSACTION DATA OF THE UNRESOLVED    *)
(*                             ENTITY                                *)
(*        UNRESOLVED_KEY   O   KEY TO THE UNRESOLVED ENTITY          *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         CREATE THE UNRESOLVED ENTITY BY CALLING MAS ROUTINES.     *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*      NARATION ON THE NEXT LINE.                                   *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCXX     I. M. APERSON             GROUP_ID *)
(*      DESCRIPTION OF FIRST CHANGE MADE.                            *)
(*                                                                   *)
```

```
(*    ORIGINATED: 04/22/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE BLDUNRES *)
```

```
(* %INCLUDE BLEXICAL *)
(**)
  PROCEDURE BLEXICAL( VAR   TOKEN            : T_TOKEN;
                      VAR   TOKEN_VALUE      : T_TOKEN_VALUE;
                      VAR   TOKEN_LOCATION   : INTEGER;
                      VAR   TOKEN_LENGTH     : INTEGER;
                      VAR   REPORT1          : TEXT);
            SUBPROGRAM;
(*                                                                      *)
(*   $FUNCTION:                                                         *)
(*        LOCATE THE LONGEST POSSIBLE LEXEME FROM WHICH A TOKEN MAY     *)
(*        BE DETERMINED, EXCLUDING COMMENTS.                            *)
(*                                                                      *)
(*   $DESCRIPTION OF ARGUMENTS:                                         *)
(*        NAME           I/O  DESCRIPTION                               *)
(*        ====           ===  ===========                               *)
(*        TOKEN          I/O  INPUT  = TOKEN TYPE FROM PREVIOUS CALL    *)
(*                                     OR INITIALIZATION FLAG           *)
(*                            OUTPUT = CURRENT TOKEN TYPE               *)
(*        TOKEN_VALUE     O   CURRENT TOKEN VALUE                       *)
(*        TOKEN_LOCATION  O   START LOCATION OF TOKEN IN REPORT LINE    *)
(*        TOKEN_LENGTH    O   LENGTH OF TOKEN IN REPORT LINE            *)
(*        REPORT1         O   REPORT FILE FOR ECHOING THE INPUT AND     *)
(*                            REPORTING ERROR MESSAGES                  *)
(*                                                                      *)
(*   $COMMONS:                                                          *)
(*                                                                      *)
(*   $ENVIRONMENT:                                                      *)
(*        LANGUAGE: IBM PASCAL                                          *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*                                                                      *)
(*   $EXECUTION PROCEDURE:                                              *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*                                                                      *)
(*    LOOP UNTIL TOKEN IS NOT A COMMENT                                 *)
(*       GET NEXT TOKEN                                                 *)
(*    END LOOP                                                          *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    ORIGINATED: 03/26/87       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*        REVISED:     7 DEC 87, G. A. WHITE, ADD PARAMETERS FOR       *)
(*                     LOCATION AND LENGTH OF TOKEN IN REPORT LINE.     *)
(*                                                                      *)
```

```
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE BLEXICAL *)
```

```
(* %INCLUDE BSCINCLD *)
(**)
  PROCEDURE BSCINCLD(VAR IRC           :  RET_REC;
                     VAR SUBSCHEMA_KEY :  ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE GENERATES THE PASCAL INCLUDE FILES AND WRITES       *)
(*        THESE DEFINITIONS TO A FILE                                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC            I/O  RETURN CODE                                  *)
(*        MSG            I/O  PANEL MESSAGE                                *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY A PANEL CONTAINING A LIST OF ALL OF THE SUBSCHEMAS       *)
(*                                              WITHIN THE SCHEMA MODEL.   *)
(*        IF RETURN OR EXIT WAS NOT CHOSEN ON THE PANEL THEN               *)
(*          IF A MULTIPLE SELECT WAS MADE ON THE PANEL THEN                *)
(*            DISPLAY AN ERROR MESSAGE                                     *)
(*          ELSE                                                           *)
(*            GET THE KEY OF THE SUBSCHEMA SELECTED                        *)
(*            IF THE SUBSCHEMA HAS NOT BEEN PHYSICALIZED THEN              *)
(*              PHYSICALIZE THE SUBSCHEMA                                  *)
(*            WRITE OUT TO THE FILE THE HEADING FOR THE INCLUDES           *)
(*            MAKE AN INCLUSIVE LIST OF ENTITIES WITHIN THE SUBSCHEMA      *)
(*            DELETE ANY DUPLICATES ON THE LIST                            *)
(*            ALPHABETICALLY SORT THE ENTITIES                             *)
(*            WRITE OUT TO THE FILE THE ENTITY KIND CONSTANTS              *)
(*            WRITE OUT TO THE FILE THE DEFINED TYPE DECLARATIONS          *)
(*            WRITE OUT TO THE FILE THE ENTITY DECLARATIONS                *)
(*            WRITE OUT TO THE FILE THE MAS ENTITY DECLARATIONS            *)
(*            WRITE OUT TO THE FILE THE KEYBLOCK DECLARATIONS              *)
```

```
(*        ELSE                                                          *)
(*          IF RETURN WAS SELECTED THEN                                 *)
(*            RETURN TO THE REPORT MENU                                 *)
(*          ELSE                                                        *)
(*            IF EXIT WAS SELECTED THEN                                 *)
(*              RETURN TO THE MAIN MENU                                 *)
(*            ELSE                                                      *)
(*              DISPLAY AN ERROR MESSAGE FOR AN INVALID OPTION          *)
(*        END;                                                          *)
(*                                                                      *)
(*    $COMMENTS:                                                        *)
(*                                                                      *)
(*    $CHANGE CONTROL:                                                  *)
(*                                                                      *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                      *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*      NARATION ON THE NEXT LINE.                                      *)
(*                                                                      *)
(*     REVISED: MM/DD/YY CCXX     I. M. APERSON              GROUP_ID *)
(*      DESCRIPTION OF FIRST CHANGE MADE.                               *)
(*                                                                      *)
(*     ORIGINATED: 10/23/86       L. J. BEHAN                DBMA     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE BSCINCLD *)
```

```
(* %INCLUDE BSCTRSPR *)
(**)
  PROCEDURE BSCTRSPR(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*         THIS ROUTINE BEGINS THE PROCESSING OF THE TRANSACTION           *)
(*         STACK.                                                          *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME           I/O  DESCRIPTION                                 *)
(*         ====           ===  ===========                                 *)
(*         IRC             O   RETURN CODE                                 *)
(*         TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK             *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*       REF                                                               *)
(*         CURRENT_LIST      I/O    POINTS TO THE LIST OF KEYS             *)
(*                                  CURRENTLY IN USE                       *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*         THIS ROUTINE CALLS SCPOPTR TO POP THE TRANSACTION STACK.        *)
(*         THEN EACH TRANSACTION IS PROCESSED ACCORDING TO ITS TYPE        *)
(*         BY CALLING THE APPROPRIATE ROUTINE TO MODEL THE ENTITIES.       *)
(*                                                                         *)
(*    $COMMENTS:                                                           *)
(*                                                                         *)
(*    $CHANGE CONTROL:                                                     *)
(*                                                                         *)
(*     REVISED: MM/DD/YY CCRR      I. M. THECHANGER            GROUP_ID    *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                                 *)
(*                                                                         *)
(*     REVISED: MM/DD/YY CCZZ      I. M. THEPROGRAMMER         GROUP_ID    *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE              *)
(*      NARATION ON THE NEXT LINE.                                         *)
(*                                                                         *)
(*     REVISED: MM/DD/YY CCXX      I. M. APERSON               GROUP_ID    *)
(*      DESCRIPTION OF FIRST CHANGE MADE.                                  *)
(*                                                                         *)
```

```
(*    ORIGINATED: 02/04/86       L. J. BEHAN                    FRMI     *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE BSCTRSPR *)
```

```
(* %INCLUDE CLRSTK *)
(**)
  PROCEDURE CLRSTK(VAR IRC          :  RET_REC;
                   VAR TRANS_STACK  :  TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that clears the transactio        *)
(*         processing stack                                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ==========                            *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
(*        TRANS_STACK    I/O  TRANSACTION STACK                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    INITIALIZE VARIABLES                                           *)
(*    WHILE THE STACK IS NOT EMPTY, POP TRANSACTION                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE CLRSTK *)
```

```
(* %INCLUDE CRARRAY *)
(**)
  PROCEDURE CRARRAY(VAR MESS     : MESSAGE;
                    VAR LBND     : CHAR8;
                    VAR HBND     : CHAR8;
                    VAR ATYPE    : ENTITY_TYPE;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE ARRAY MENU                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        LBND           O   THE LOWER BOUND OF THE ARRAY            *)
(*        HBND           O   THE UPPER BOUND OF THE ARRAY            *)
(*        ATYPE          O   THE ARRAY TYPE                          *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE ARRAY PANEL (CRARRAY) BY MAKING ISPLNK  *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED     *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRCLASS1 *)
(**)
  PROCEDURE CRCLASS1(VAR MESS     : MESSAGE;
                     VAR NAME     : T_NAME;
                     VAR KNUM     : CHAR8;
                     VAR NEXT_OP  : OPERATIONS;
                     VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE CLASS1 PANEL               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           O   THE CLASS NAME                          *)
(*        KNUM           O   THE CLASS KIND NUMBER                   *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE CLASS PANEL NUMBER ONE (CRCLASS1) BY    *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED      *)
(*        INTO AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER       *)
(*        INFORMATION GATHERED FROM THE PANEL IS PASSED BACK TO THE  *)
(*        CALLING PROCEDURE.                                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRCLASS2 *)
(**)
  PROCEDURE CRCLASS2(VAR MESS    : MESSAGE;
                     VAR KNUM    : CHAR8;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      THIS FUNCTION:                                                    *)
(*                      DISPLAYS THE CREATE CLASS PANEL 2 MENU            *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        KNUM           O   THE MEMBER KIND NUMBER                       *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                    *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISLAY THE CREATE CLASS PANEL NUMBER TWO (CRCLASS2) BY          *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO      *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION     *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-      *)
(*        CEDURE.                                                         *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE CRDEFTYP *)
(**)
  PROCEDURE CRDEFTYP(VAR MESS    : MESSAGE;
                     VAR NAME    : CHAR16;
                     VAR FTYPE   : ENTITY_TYPE;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE DEFINED TYPE MENU          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           O   THE NAME OF THE DEFINED TYPE ENTERED    *)
(*        FTYPE          O   TYPES INTEGER,STRING,REAL...ETC.        *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE DEFINED TYPE PANEL (CRDEFTYP) BY MAKING *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION   *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO- *)
(*        CEDURE.                                                    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRENTITY *)
(**)
  PROCEDURE CRENTITY(VAR MESS    : MESSAGE;
                     VAR NAME    : T_NAME;
                     VAR KNUM    : CHAR8;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*             THIS PROCEDURE :                                      *)
(*                  DISPLAYS THE CREATE ENTITY MENU                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           O   THE ENTITY NAME                         *)
(*        KNUM           O   THE ENTITY KIND NUMBER                  *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE ENTITY PANEL (CRENTITY) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED FROM *)
(*        THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRENUM *)
(**)
  PROCEDURE CRENUM(VAR MESS    : MESSAGE;
                   VAR NAME    : CHAR16;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE ENUMERATION MENU           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           O   THE NAME OF THE ENUMERATION FROM PANEL   *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        XRC            O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE ENUMERATION PANEL (CRENUM) BY MAKING    *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION   *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING      *)
(*        PROCEDURE.                                                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRFIELD *)
(**)

  PROCEDURE CRFIELD(VAR MESS    : MESSAGE;
                    VAR NAME    : T_NAME;
                    VAR POS     : CHAR8;
                    VAR PURP    : CHAR8;
                    VAR REQD    : CHAR8;
                    VAR DEPD    : CHAR8;
                    VAR FTYPE   : ENTITY_TYPE;
                    VAR FLDTYP  : T_FIELDTYPE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR      : RET_REC);

      SUBPROGRAM;

(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS PROCEDURE :                                                 *)
(*                          DISPLAYS THE CREATE FIELD PANEL                *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL    *)
(*        NAME            O   THE NAME OF THE FIELD                       *)
(*        PURP            O   THE PURPOSE OF THE FIELD                    *)
(*        REQD            O   REQUIREDNESS, OPTIONAL OR NOT, OF FIELD     *)
(*        DEPD            O   DEPENDENCE OF THE FIELD                     *)
(*        FTYPE           O   ENTITY TYPE                                 *)
(*        FLDTYPE         O   THE FIELD TYPE                              *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT         *)
(*                            OPERATION                                    *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,     *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE CREATE FIELD PANEL (CRFIELD) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED   *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED FROM  *)
(*        THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.           *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*        NONE                                                         *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
```

```
(* %INCLUDE CRINTGR *)
(**)
  PROCEDURE CRINTGR(VAR MESS     : MESSAGE;
                    VAR PREC     : CHAR8;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE INTEGER MENU               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        PREC           O   THE PRECISION OF THE INTEGER ENTERED    *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE INTEGER PANEL (CRINTGR) BY MAKING       *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION   *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING      *)
(*        PROCEDURE.                                                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRLIST *)
(**)
  PROCEDURE CRLIST(VAR MESS     : MESSAGE;
                   VAR MIN      : CHAR8;
                   VAR MAX      : CHAR8;
                   VAR ATYPE    : ENTITY_TYPE;
                   VAR NEXT_OP  : OPERATIONS;
                   VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE CREATE LIST PANEL                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MIN            O   THE MINIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        MAX            O   THE MAXIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        ATYPE          O   THE LIST TYPE                                *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE CREATE LIST PANEL (CRLIST) BY MAKING ISPLINK         *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED           *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.          *)
(*                                                                         *)
```

```
(*   $COMMENTS:                                                          *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                       *)
(*    REVISED: MM/DD/YY          I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF THE CHANGE MADE.  IF LENGTHY, CONTINUE THE       *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                       *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                 DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE CRLIST *)
```

```
(* %INCLUDE CRPNTR *)
(**)
  PROCEDURE CRPNTR(VAR MESS    : MESSAGE;
                   VAR KNUM    : CHAR8;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE CREATE POINTER MENU                    *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        KNUM           O   THE MEMBER KIND NUMBER OF THE POINTER        *)
(*                           INTO AN INTEGER                               *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE CREATE POINTER PANEL (CRPNTR) BY MAKING ISPLNK       *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED FROM      *)
(*        THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.               *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE CRREAL *)
(**)
  PROCEDURE CRREAL(VAR MESS    : MESSAGE;
                   VAR SIZE    : CHAR8;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*       THIS FUNCTION:                                              *)
(*                      DISPLAYS THE CREATE REAL PANEL               *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                            *)
(*         ====          ===  ===========                            *)
(*         MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*         SIZE           O   THE PRECISION OF THE REAL ENTERED       *)
(*         NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                            OPERATION                              *)
(*         RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*         DDNAMES USED WITH STANDARD FILES:                         *)
(*              NONE                                                 *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         DISPLAY THE CREATE REAL PANEL (CRREAL) BY MAKING ISPLNK   *)
(*         CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*         TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED FROM *)
(*         THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.        *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE CRSET *)
(**)
  PROCEDURE CRSET(VAR MESS      : MESSAGE;
                  VAR MIN       : CHAR8;
                  VAR MAX       : CHAR8;
                  VAR ATYPE     : ENTITY_TYPE;
                  VAR NEXT_OP   : OPERATIONS;
                  VAR RR        : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*       THIS FUNCTION:                                              *)
(*                     DISPLAYS THE CREATE SET PANEL                 *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*         MIN            O   THE MINIMUM NUMBER OF OCCURRENCES IN THE *)
(*                            SET                                    *)
(*         MAX            O   THE MAXIMUM NUMBER OF OCCURRENCES IN THE *)
(*                            SET                                    *)
(*         ATYPE          O   THE SET TYPE                           *)
(*         NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                            OPERATION                              *)
(*         RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*         DDNAMES USED WITH STANDARD FILES:                         *)
(*              NONE                                                 *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         DISPLAY THE CREATE SET PANEL (CRSET) BY MAKING ISPLINK    *)
(*         CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*         TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED    *)
(*         FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.   *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                       *)
(*    REVISED: MM/DD/YY          I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF THE CHANGE MADE.  IF LENGTHY, CONTINUE THE       *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                       *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END-----------------------------------------------------------------------*)
(* END %INCLUDE CRSET *)
```

```
(* %INCLUDE CRSTRING *)
(**)
  PROCEDURE CRSTRING(VAR MESS    : MESSAGE;
                     VAR SLEN    : CHAR8;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE CREATE STRING PANEL               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        SLEN           O   THE LENGTH OF THE STRING ENTERED        *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION.                              *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE STRING PANEL (CRSTRING) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED FROM *)
(*        THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CRSUBSCM *)
(**)
  PROCEDURE CRSUBSCM(VAR MESS    : MESSAGE;
                     VAR NAME    : T_NAME;
                     VAR KNUM    : CHAR8;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                      DISPLAYS THE CREATE SUBSCHEMA PANEL                *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NAME           O   THE SUBSCHEMA NAME                           *)
(*        KNUM           O   THE SUBSCHEMA MEMBER KIND NUMBER             *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE CREATE SUBSCHEMA PANEL (CRSUBSCM) BY MAKING          *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN           *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION         *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-       *)
(*        CEDURE.                                                          *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE CRSUPTYP *)
(**)
  PROCEDURE CRSUPTYP(VAR MESS        : MESSAGE;
                     VAR NAME        : T_NAME;
                     VAR REFR        : CHAR8;
                     VAR CREATE_ONLY : BOOLEAN;
                     VAR NEXT_OP     : OPERATIONS;
                     VAR RR          : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*       THIS PROCEDURE:                                                   *)
(*             DISPLAYS EITHER THE CREATE/REFERENCE SUPERTYPE MENU          *)
(*             OR THE CREATE SUPERTYPE MENU                                *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME          I/O  DESCRIPTION                                  *)
(*         ====          ===  ===========                                  *)
(*         MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*         NAME           O   THE ENTITY NAME                              *)
(*         REFR           O   INDICATES IF THE SUPERTYPE REFERENCES        *)
(*                            ANOTHER SUPERTYPE                            *)
(*         CREATE_ONLY    I   INDICATES IF A SUPERTYPE CAN BE CREATED/     *)
(*                            REFERENCED OR ONLY CREATED.                  *)
(*         NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                            OPERATION                                    *)
(*         RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*         NONE                                                            *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*         DDNAMES USED WITH STANDARD FILES:                               *)
(*              NONE                                                       *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*         DISPLAY THE CREATE SUPERTYPE PANEL (CRSUPTYPE) BY MAKING        *)
(*         ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN          *)
(*         ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION        *)
(*         GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING           *)
(*         PROCEDURE.                                                      *)
(*                                                                         *)
```

```
(*   $COMMENTS:                                                          *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY          I. M. APROGRAMMER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                *)
(*                                                                       *)
(*    ORIGINATED: 09/28/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE CRSUPTYP *)
```

```
(* %INCLUDE CRURUL. *)
(**)
  PROCEDURE CRURUL(CONST ENTITY_TYPE:ORD_KIND;VAR GROUP:T_GROUP_ARRAY;
      VAR NUM_GROUP:LISTPSTN; VAR MIN_CNST:LISTPSTN);EXTERNAL;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*        CREATES THE USER'S RULES.  RULES OF CONNECTIVITY USED TO         *)
(*        DETERMINE DELETABILITY OF ENTITIES.                              *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME           I/O  DESCRIPTION                                 *)
(*         ====           ===  ===========                                 *)
(*         ENTITY_TYPE     I   ENTITY KIND VALUE WHICH WILL HAVE THE       *)
(*                             DELETE RULE                                 *)
(*         GROUP           O   ARRAY THAT WILL BE FILLED WITH THE RULES    *)
(*                             AND NUMBER OF CONSTITUENTS OF EACH          *)
(*                             DIFFERENT RELATIONSHIP THAT THIS ENTITY     *)
(*                             KIND CAN HAVE WITH ITS CONSTITUENTS         *)
(*         NUM_GROUP       O   NUMBER OF DIFFERENT RELATIONSHIPS THIS      *)
(*                             ENTITY CAN HAVE WITH ITS CONSTITUENTS       *)
(*         MIN_CNST        O   MINIMUM NUMBER OF CONSTITUENTS THAT THIS    *)
(*                             ENTITY CAN HAVE WHEN IT HAS A GROUP OF      *)
(*                             CONSTITUENTS THAT ARE "SECONDARY"           *)
(*         RC              O   EXTERNAL RETURN CODE                        *)
(*                             = 0  OK RETURN CODE                         *)
(*                             > 0  CRITICAL ERROR                         *)
(*                             < 0  WARNING                                *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE MODEL ACCESS SOFTWARE                *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*        ??????ARE SET TO INDICATE IF THE RELATIONSHIP BETWEEN THE        *)
(*        USER AND ITS CONSTITUTES IS DEPENDENT OR INDEPENDENT AND         *)
(*        STRONG OR WEAK.                                                  *)
(*        DEFAULT RULE IS DEPENDENT/STRONG.                                *)
(*                                                                         *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: 09/28/87          C. H. MOHME                 DBMA      *)
(*     TO INCORPORATE SUPERTYPE DATA TYPE                               *)
(*                                                                       *)
(*    REVISED: 04/09/87          C. H. MOHME                 DBMA      *)
(*     TO INCORPORATE LIST DATA TYPE                                    *)
(*                                                                       *)
(*    REVISED: 09/29/86          L. J. BEHAN                 DBMA      *)
(*     ENTERED THE NEW RULES FOR THE SCHEMA EXECUTIVE ENTITIES          *)
(*                                                                       *)
(*    REVISED: 06/19/86          B. A. ULMER                 FRMI      *)
(*     REDO LOGIC OF HOW CRURUL WORKS BASED ON THE NEW DELETE RULES     *)
(*                                                                       *)
(*    REVISED: 09/  /85          B. A. ULMER                 FRMI      *)
(*     ADD ENTITY KINDS SO AS TO TEST THE NEW DELETE RULES (2070,       *)
(*     2080, 2090)                                                      *)
(*                                                                       *)
(*    REVISED: 09/  /85          B. A. ULMER                 FRMI      *)
(*     ADD PARAMETERS TO HANDLE THE TWO NEW DELETE RULES                *)
(*                                                                       *)
(*    REVISED: 09/18/84          D. J. KERCHNER              FRMI      *)
(*     ADDED I/S RULE FOR THE PICK ENTITY                               *)
(*                                                                       *)
(*    ORIGINATED: MM/DD/YY CCWW  I. M. THEORIGINATOR         GROUP_ID  *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
%PAGE                                                                   *)
(*-----------------------------------------------------------------------*)
(*    DATA STRUCTURES/MAJOR VARIABLES:                                   *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END-----------------------------------------------------------------*)
(**)
(* END %INCLUDE CRURUL *)
```

```
(* %INCLUDE CSARYWRT *)
(**)
  PROCEDURE CSARYWRT(VAR IRC         : RET_REC;
                     VAR ARRAY_KEY   : ENTKEY;
                     VAR CSRFILE     : TEXT;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER;
                     VAR INDENT      : INTEGER;
                     VAR PAGE_TYPE   : PAGES);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT AN ARRAY DEFINITION                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   INTERNAL RETURN CODE                        *)
(*        ARRAY_KEY       I   ARRAY KEY                                   *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                          *)
(*        INDENT         I/O  NUMBER OF SPACES TO INDENT                  *)
(*        PAGE_TYPE      I/O  TYPE OF REPORT PAGE                         *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   CREATE A NEW PAGE WITH HEADING, IF NECESSARY.                         *)
(*   GET THE ARRAY'S ADB.                                                  *)
(*   WRITE OUT TO THE FILE THE ARRAY BOUNDS.                               *)
(*   SET THE LIST OF CONSTITUENTS OF THE ARRAY TO BE READ IN THE           *)
(*     FORWARD DIRECTION.                                                  *)
(*   READ THE ARRAY CONSTITUENT FROM THE LIST.                             *)
(*   GET THE ARRAY CONSTITUENT'S ADB.                                      *)
```

```
(*   CASE CONSTITUENT_ADB.ENT_KIND OF                                    *)
(*     INTEGER      :  WRITE OUT INTEGER DEFINITION                      *)
(*     REAL         :  WRITE OUT REAL DEFINITION                         *)
(*     STRING       :  WRITE OUT STRING DEFINITION                       *)
(*     LOGICAL      :  WRITE OUT LOGICAL DEFINITION                      *)
(*     ARRAY        :  WRITE OUT ARRAY DEFINITION                        *)
(*     DEFINED TYPE :  WRITE OUT DEFINED TYPE DEFINITION                 *)
(*     POINTER      :  WRITE OUT POINTER DEFINITION                      *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
```

```
(* %INCLUDE CSCLSHDG *)
(**)
  PROCEDURE CSCLSHDG(VAR CSRFILE     : TEXT;
                     VAR HEADING     : HEADING_TYPE;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES OUT A CLASS HEADING ON A NEW PAGE.     *)
(*                                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        CSRFILE       I/O  OUTPUT FILE                           *)
(*        HEADING        I   DEFINITION OR INDEX HEADING           *)
(*        PAGE_NUMBER   I/O  CURRENT PAGE NUMBER                   *)
(*        LINE_COUNT     O   LINE COUNT ON THE CURRENT PAGE        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   CREATE A NEW PAGE.                                              *)
(*   WRITE OUT TO THE FILE THE APPROPRIATE CLASS HEADING (DEFINITION *)
(*     OR INDEX).                                                    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CSCLSWRT *)
(**)
  PROCEDURE CSCLSWRT(VAR IRC          : RET_REC;
                     VAR CLASS_LIST   : LISTKEY;
                     VAR CSRFILE      : TEXT;
                     VAR PAGE_NUMBER  : INTEGER;
                     VAR CURRENT_PAGE : PAGE_PTR);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES OUT CLASS DEFINITIONS TO A FILE        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC            O   INTERNAL RETURN CODE                  *)
(*        CLASS_LIST     I   ALPHABETIZED LIST OF CLASSES          *)
(*        CSRFILE       I/O  THE OUTPUT FILE                       *)
(*        PAGE_NUMBER   I/O  THE CURRENT PAGE NUMBER               *)
(*        CURRENT_PAGE  I/O  POINTS TO THE CURRENT PAGE RECORD IN  *)
(*                           THE CHAIN                             *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   SET THE LIST OF CLASSES TO BE READ IN THE FORWARD DIRECTION.    *)
(*   COUNT THE NUMBER OF CLASSES IN THE LIST.                        *)
(*   FOR X = 1 TO THE NUMBER OF CLASSES IN THE LIST                  *)
(*     WRITE OUT TO THE FILE THE CLASS HEADING ON A NEW PAGE.        *)
(*     UPDATE THE PAGE RECORD CHAIN.                                 *)
(*     READ A CLASS FROM THE LIST OF CLASSES.                        *)
(*     GET THE CLASS' ADB.                                           *)
(*     WRITE OUT TO THE FILE THE CLASS NAME AND NUMBER.              *)
(*     SET THE LIST OF CONSTITUENTS OF THE CLASS TO BE READ IN THE   *)
(*       FORWARD DIRECTION.                                          *)
(*     COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                 *)
```

```
(*      FOR Y = 1 TO THE NUMBER OF CLASS CONSTITUENTS                    *)
(*        READ A CONSTITUENT FROM THE LIST OF CONSTITUENTS.              *)
(*        GET THE CONSTITUENT'S ADB.                                     *)
(*        CREATE A NEW PAGE, IF NECESSARY.                               *)
(*        WRITE OUT TO THE FILE THE CONSTITUENT (CLASS OR ENTITY)        *)
(*        INCREMENT THE LINE COUNTER.                                    *)
(*      WRITE OUT TO THE FILE 'END;'.                                    *)
(*                                                                       *)
(*    $COMMENTS:                                                         *)
(*                                                                       *)
(*    $CHANGE CONTROL:                                                   *)
(*                                                                       *)
```

```
(* %INCLUDE CSDEFHDG *)
(**)
  PROCEDURE CSDEFHDG(VAR CSRFILE     : TEXT;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A DEFINED TYPE HEADING ON A NEW         *)
(*        PAGE.                                                           *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE              *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   CREATE A NEW PAGE.                                                   *)
(*   WRITE OUT TO THE FILE THE DEFINED TYPE DEFINITION HEADING.           *)
(*   INCREMENT THE LINE COUNTER.                                          *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE CSDEFWRT *)
(**)
  PROCEDURE CSDEFWRT(VAR IRC              : RET_REC;
                     VAR DEFINED_TYPE_KEY : ENTKEY;
                     VAR CSRFILE          : TEXT;
                     VAR LINE_COUNT       : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT A DEFINED TYPE NAME                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*    IRC                 O   INTERNAL RETURN CODE                       *)
(*    DEFINED_TYPE_KEY    I   DEFINED TYPE KEY                           *)
(*    CSRFILE            I/O  THE OUTPUT FILE                            *)
(*    LINE_COUNT         I/O  CURRENT LINE COUNT                         *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   GET THE DEFINED TYPE'S ADB.                                           *)
(*   WRITE OUT TO THE FILE THE DEFINED TYPE NAME.                          *)
(*   INCREMENT THE LINE COUNTER.                                           *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE CSENMWRT *)
(**)
  PROCEDURE CSENMWRT(VAR IRC              : RET_REC;
                     VAR ENUMERATION_KEY : ENTKEY;
                     VAR CSRFILE          : TEXT;
                     VAR PAGE_NUMBER      : INTEGER;
                     VAR LINE_COUNT       : INTEGER;
                     VAR PAGE_TYPE        : PAGES);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT AN ENUMERATION DEFINITION                *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*     IRC                0   INTERNAL RETURN CODE                       *)
(*     ENUMERATION_KEY    I   ENUMERATION KEY                            *)
(*     CSRFILE           I/O  THE OUTPUT FILE                            *)
(*     PAGE_NUMBER       I/O  CURRENT PAGE NUMBER                        *)
(*     LINE_COUNT        I/O  CURRENT LINE COUNT                         *)
(*     PAGE_TYPE         I/O  TYPE OF REPORT PAGE                        *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL                                           *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT            *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*                                                                       *)
(*   GET THE ENUMERATION'S ADB.                                          *)
(*   WRITE OUT TO THE FILE 'ENUMERATION OF ('.                           *)
(*   INCREMENT THE LINE COUNTER.                                         *)
(*   SET THE LIST OF CONSTITUENTS OF THE ENUMERATION TO BE READ IN       *)
(*     THE FORWARD DIRECTION.                                            *)
(*   COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                       *)
(*   FOR Y = 1 TO THE NUMBER OF CONSTITUENTS                             *)
(*     READ A CONSTITUENT FROM THE LIST OF CONSTITUENTS.                 *)
(*     GET THE CONSTITUENT'S ADB.                                        *)
(*     CREATE A NEW PAGE, IF NECESSARY.                                  *)
(*     WRITE OUT TO THE FILE THE CONSTITUENT.                            *)
(*     INCREMENT THE LINE COUNTER.                                       *)
(*   WRITE OUT TO THE FILE ')'.                                          *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
```

```
(* %INCLUDE CSENTHDG *)
(**)
  PROCEDURE CSENTHDG(VAR CSRFILE     : TEXT;
                     VAR HEADING     : HEADING_TYPE;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT AN ENTITY HEADING ON A NEW PAGE.         *)
(*                                                                         *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        CSRFILE       I/O  OUTPUT FILE                                   *)
(*        HEADING        I   DEFINITION OR INDEX HEADING                   *)
(*        PAGE_NUMBER   I/O  CURRENT PAGE NUMBER                           *)
(*        LINE_COUNT     O   LINE COUNT ON THE CURRENT PAGE                *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   CREATE A NEW PAGE.                                                    *)
(*   WRITE OUT TO THE FILE THE APPROPRIATE ENTITY HEADING (DEFINITION      *)
(*     OR INDEX).                                                          *)
(*   INCREMENT THE LINE COUNTER.                                           *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE CSENTWRT *)
(**)
  PROCEDURE CSENTWRT(VAR IRC          : RET_REC;
                     VAR ENTITY_LIST  : LISTKEY;
                     VAR CSRFILE      : TEXT;
                     VAR PAGE_NUMBER  : INTEGER;
                     VAR CURRENT_PAGE : PAGE_PTR);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT ENTITY DEFINITIONS TO A FILE             *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ==========                                    *)
(*        IRC            O   INTERNAL RETURN CODE                         *)
(*        ENTITY_LIST    I   ALPHABETIZED LIST OF ENTITIES                *)
(*        CSRFILE       I/O  THE OUTPUT FILE                              *)
(*        PAGE_NUMBER   I/O  THE CURRENT PAGE NUMBER                      *)
(*        CURRENT_PAGE  I/O  POINTS TO THE CURRENT PAGE RECORD IN         *)
(*                           THE CHAIN.                                    *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   SET THE LIST OF ENTITIES TO BE READ IN THE FORWARD DIRECTION.         *)
(*   COUNT THE NUMBER OF ENTITIES IN THE LIST.                             *)
(*   FOR X = 1 TO THE NUMBER OF ENTITIES IN THE LIST                       *)
(*     WRITE OUT TO THE FILE THE ENTITY HEADING.                           *)
(*     UPDATE THE PAGE RECORD CHAIN.                                       *)
(*     READ AN ENTITY FROM THE LIST OF ENTITIES.                           *)
(*     GET THE ENTITY'S ADB.                                               *)
(*     WRITE OUT TO THE FILE THE ENTITY NAME AND NUMBER.                   *)
(*     SET THE LIST OF CONSTITUENTS OF THE ENTITY TO BE READ IN THE        *)
(*       FORWARD DIRECTION.                                                *)
(*     COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                       *)
```

```
(*     FOR Y = 1 TO THE NUMBER OF CONSTITUENTS IN THE LIST          *)
(*       READ A CONSTITUENT FROM THE LIST.                          *)
(*       GET THE CONSTITUENT'S ADB.                                 *)
(*       CREATE A NEW PAGE, IF NECESSARY.                           *)
(*       WRITE OUT TO THE FILE A FIELD NAME.                        *)
(*       SET THE TYPE OF FIELD TO BE READ IN THE FORWARD DIRECTION. *)
(*       GET THE TYPE'S KEY.                                        *)
(*       GET THE TYPE'S ADB.                                        *)
(*       CASE TYPE_ADB.ENT_KIND OF                                  *)
(*         INTEGER      :  WRITE OUT THE INTEGER DEFINITION         *)
(*         REAL         :  WRITE OUT THE REAL DEFINITION            *)
(*         STRING       :  WRITE OUT THE STRING DEFINITION          *)
(*         LOGICAL      :  WRITE OUT THE LOGICAL DEFINITION         *)
(*         ARRAY        :  WRITE OUT THE ARRAY DEFINITION           *)
(*         DEFINED TYPE :  WRITE OUT THE DEFINED TYPE DEFINITION    *)
(*         POINTER      :  WRITE OUT THE POINTER DEFINITION         *)
(*     WRITE OUT TO THE FILE 'END;'.                                *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
```

```
(* %INCLUDE CSGBLHDG *)
(**)
  PROCEDURE CSGBLHDG(VAR CSRFILE     : TEXT;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A GLOBAL FIELD HEADING ON A NEW         *)
(*        PAGE.                                                           *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE              *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   CREATE A NEW PAGE.                                                   *)
(*   WRITE OUT TO THE FILE THE GLOBAL FIELD DEFINITION HEADING.           *)
(*   INCREMENT THE LINE COUNTER.                                          *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE CSGBLWRT *)
(**)
  PROCEDURE CSGBLWRT(VAR IRC                : RET_REC;
                     VAR GLOBAL_FIELD_LIST : LISTKEY;
                     VAR CSRFILE           : TEXT;
                     VAR PAGE_NUMBER       : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT GLOBAL FIELD DEFINITIONS TO A FILE       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*    IRC                 O   INTERNAL RETURN CODE                        *)
(*    GLOBAL_FIELD_LIST   I   LIST OF GLOBAL FIELDS                       *)
(*    CSRFILE            I/O  THE OUTPUT FILE                             *)
(*    PAGE_NUMBER        I/O  THE CURRENT PAGE NUMBER                     *)
(*    PAGE_MARK          I/O  ARRAY OF BOOLEAN:  TRUE IF PAGE MARKS       *)
(*                            THE BEGINNING OF A NEW ENTITY, FALSE       *)
(*                            OTHERWISE.                                 *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL                                           *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT            *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*                                                                       *)
(*   SET THE LIST OF GLOBAL FIELDS TO BE READ IN THE FORWARD             *)
(*     DIRECTION.                                                        *)
(*   COUNT THE NUMBER OF GLOBAL FIELDS IN THE LIST.                      *)
(*   CREATE A LIST FOR THE ALPHABETIZED GLOBAL FIELDS.                   *)
(*   FOR X = 1 TO THE NUMBER OF GLOBAL FIELDS IN THE LIST                *)
(*     READ A GLOBAL FIELD FROM THE LIST OF GLOBAL FIELDS.               *)
(*     SET THE CONSTITUENT OF THE GLOBAL FIELD TO BE READ IN THE         *)
(*       FORWARD DIRECTION.                                              *)
(*     READ A FIELD KEY FROM THE CONSTITUENT LIST OF THE GLOBAL FIELD.   *)
(*     ATTACH THE FIELD KEY TO THE LIST FOR THE ALPHABETIZED GLOBAL      *)
(*       FIELDS.                                                         *)
```

```
(*   SORT THE LIST OF GLOBAL FIELDS INTO ALPHABETICAL ORDER.          *)
(*   SET THE LIST OF GLOBAL FIELDS TO BE READ IN THE FORWARD          *)
(*     DIRECTION.                                                     *)
(*   WRITE OUT TO THE FILE THE GLOBAL FIELD DEFINITION HEADING.       *)
(*   FOR X = 1 TO THE NUMBER OF GLOBAL FIELDS                         *)
(*     READ A GLOBAL FIELD FROM THE LIST OF GLOBAL FIELDS.            *)
(*     GET THE FIELD'S ADB (THE GLOBAL FIELD'S CONSTITUENT).          *)
(*     CREATE A NEW PAGE, IF NECESSARY.                               *)
(*     WRITE OUT TO THE FILE THE GLOBAL FIELD NAME.                   *)
(*     SET THE TYPE OF THE FIELD TO BE READ IN THE FORWARD DIRECTION. *)
(*     GET THE TYPE KEY.                                              *)
(*     GET THE TYPE'S ADB.                                            *)
(*     CASE TYPE_ADB.ENT_KIND OF                                      *)
(*       INTEGER      :  WRITE OUT THE INTEGER DEFINITION             *)
(*       REAL         :  WRITE OUT THE REAL DEFINITION                *)
(*       STRING       :  WRITE OUT THE STRING DEFINITION              *)
(*       LOGICAL      :  WRITE OUT THE LOGICAL DEFINITION             *)
(*       ARRAY        :  WRITE OUT THE ARRAY DEFINITION               *)
(*       DEFINED TYPE :  WRITE OUT THE DEFINED TYPE DEFINITION        *)
(*       POINTER      :  WRITE OUT THE POINTER DEFINITION             *)
(*     WRITE OUT TO THE FILE A BLANK LINE.                            *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE CSHDGWRT *)
(**)
  PROCEDURE CSHDGWRT(VAR CSRFILE     : TEXT;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER;
                     VAR HEADING     : HEADING_TYPE;
                     VAR PAGE_TYPE   : PAGES);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE CALLS THE APPROPRIATE HEADING ROUTINE.              *)
(*                                                                         *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        CSRFILE        I/O  OUTPUT FILE                                  *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                          *)
(*        LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE               *)
(*        HEADING         I   DEFINITION OR INDEX HEADING                  *)
(*        PAGE_TYPE       I   TYPE OF HEADING TO PRINT                     *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   CASE TYPE OF PAGE OF                                                  *)
(*     ENTITY PAGE       :  WRITE OUT TO THE FILE THE ENTITY HEADING.      *)
(*     CLASS PAGE        :  WRITE OUT TO THE FILE THE CLASS HEADING.       *)
(*     DEFINED TYPE PAGE :  WRITE OUT TO THE FILE THE DEFINED TYPE         *)
(*                          HEADING.                                       *)
```

```
(*     SUBSCHEMA PAGE     .  WRITE OUT TO THE FILE THE SUBSCHEMA        *)
(*                           HEADING.                                   *)
(*                                                                      *)
(*  $COMMENTS:                                                          *)
(*                                                                      *)
(*  $CHANGE CONTROL:                                                    *)
(*                                                                      *)
```

```
(* %INCLUDE CSINDWRT *)
(**)
  PROCEDURE CSINDWRT(VAR IRC          : RET_REC;
                     VAR INDEX_TYPE   : PAGES;
                     VAR LIST         : LISTKEY;
                     VAR CSRFILE      : TEXT;
                     VAR PAGE_NUMBER  : INTEGER;
                     VAR CURRENT_PAGE : PAGE_PTR);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE WRITES OUT TO A FILE AN INDEX FOR AN ENTITY, *)
(*         CLASS, OR SUBSCHEMA.                                      *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         IRC            O   INTERNAL RETURN CODE                  *)
(*         INDEX_TYPE     I   TYPE OF INDEX TO BE PRINTED           *)
(*         LIST           I   ALPHABETIZED LIST OF ENTITIES         *)
(*         CSRFILE       I/O  THE OUTPUT FILE                       *)
(*         PAGE_NUMBER   I/O  THE CURRENT PAGE NUMBER               *)
(*         CURRENT_PAGE  I/O  POINTS TO A PAGE RECORD WHICH CONTAINS*)
(*                            THE PAGE NUMBER IN THE REPORT FOR THE *)
(*                            ENTITY.                               *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT       *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*                                                                   *)
(*    SET THE LIST OF ENTITIES TO BE READ IN THE FORWARD DIRECTION.  *)
(*    COUNT THE NUMBER OF ENTITIES IN THE LIST.                      *)
(*    WRITE OUT THE PROPER PAGE HEADING.                             *)
(*    FOR X = 1 TO THE NUMBER OF ENTITIES                            *)
(*      WRITE OUT THE PROPER PAGE HEADING, IF NEW PAGE IS NEEDED.    *)
(*      READ AN ENTITY FROM THE LIST OF ENTITIES.                    *)
(*      GET THE ENTITY'S ADB.                                        *)
```

```
(*     WRITE OUT TO THE FILE THE ENTITY NAME (AND NUMBER).          *)
(*     WRITE OUT TO THE FILE THE PAGE NUMBER THAT THE ENTITY BEGINS  *)
(*       ON.                                                        *)
(*     INCREMENT THE LINE COUNT.                                    *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
```

```
(* %INCLUDE CSINTWRT *)
(**)
  PROCEDURE CSINTWRT(VAR IRC         : RET_REC;
                     VAR INT_KEY     : ENTKEY;
                     VAR CSRFILE     : TEXT;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES OUT AN INTEGER DEFINITON               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC            O   INTERNAL RETURN CODE                  *)
(*        INT_KEY        I   INTEGER KEY                           *)
(*        CSRFILE       I/O  THE OUTPUT FILE                       *)
(*        LINE_COUNT    I/O  CURRENT LINE COUNT                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   GET THE INTEGER'S ADB.                                          *)
(*   WRITE OUT TO THE FILE THE INTEGER DEFINITION.                   *)
(*   INCREMENT THE LINE COUNTER.                                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CSLOGWRT *)
(**)
  PROCEDURE CSLOGWRT(VAR IRC         : RET_REC;
                     VAR CSRFILE     : TEXT;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------*)
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        THIS ROUTINE WRITES OUT A LOGICAL DEFINITION              *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   RETURN RECORD                        *)
(*        CSRFILE        I/O  THE OUTPUT FILE                      *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                   *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*        NONE                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT       *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*   WRITE OUT TO THE FILE 'LOGICAL;'.                              *)
(*   INCREMENT THE LINE COUNTER.                                    *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
```

```
(* %INCLUDE CSMAIN *)
(**)
  PROCEDURE CSMAIN(VAR IRC      : RET_REC;
                   VAR MSG      : MESSAGE);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE SERVES AS THE MAIN DRIVER FOR THE CONCEPTUAL  *)
(*        SCHEMA REPORT.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC            O   INTERNAL RETURN CODE                  *)
(*        MSG            O   MESSAGE RETURNED INDICATING IF A REPORT *)
(*                           HAS BEEN PRODUCED                     *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   INITIALIZE THE VARIABLES USED WITHIN THIS ROUTINE.              *)
(*   WRITE OUT TO THE FILE THE REPORT COVER.                         *)
(*   INITIALIZE THE PAGE NUMBER AND PAGE CHAIN.                      *)
(*   MAKE A LIST OF DEFINED TYPES WITHIN THE SCHEMA.                 *)
(*   ALPHABETIZE THE LIST OF DEFINED TYPES.                          *)
(*   WRITE OUT TO THE FILE THE DEFINED TYPES WITHIN THE SCHEMA.      *)
(*   DELETE THE LIST OF DEFINED TYPES WITHIN THE SCHEMA.             *)
(*   MAKE A LIST OF GLOBAL FIELDS WITHIN THE SCHEMA.                 *)
(*   WRITE OUT TO THE FILE THE GLOBAL FIELDS WITHIN THE SCHEMA.      *)
(*   DELETE THE LIST OF GLOBAL FIELDS WITHIN THE SCHEMA.             *)
(*   MAKE A LIST OF ENTITIES WITHIN THE SCHEMA.                      *)
(*   ALPHABETIZE THE LIST OF ENTITIES.                               *)
(*   WRITE OUT TO THE FILE THE ENTITIES WITHIN THE SCHEMA.           *)
(*   MAKE A LIST OF CLASSES WITHIN THE SCHEMA.                       *)
(*   ALPHABETIZE THE LIST OF CLASSES.                                *)
(*   WRITE OUT TO THE FILE THE CLASSES WITHIN THE SCHEMA.            *)
```

```
(*   MAKE A LIST OF SUBSCHEMAS WITHIN THE SCHEMA.                    *)
(*   ALPHABETIZE THE LIST OF SUBSCHEMAS.                             *)
(*   WRITE OUT TO THE FILE THE SUBSCHEMAS WITHIN THE SCHEMA.         *)
(*   WRITE OUT TO THE FILE THE ENTITY INDEX.                         *)
(*   DELETE THE LIST OF ENTITIES.                                    *)
(*   WRITE OUT TO THE FILE THE CLASS INDEX.                          *)
(*   DELETE THE LIST OF CLASSES.                                     *)
(*   WRITE OUT TO THE FILE THE SUBSCHEMA INDEX.                      *)
(*   DELETE THE LIST OF SUBSCHEMAS.                                  *)
(*   IF NO MODEL EXISTS, WRITE APPROPRIATE MESSAGE.                  *)
(*   DISPOSE OF POINTERS USED IN THIS ROUTINE.                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*     WITHIN THE INCLUDE FILE 'CSTYPCON', ONE CAN SET THE MAXIMUM   *)
(*   NUMBER OF LINES PER PAGE IN THE REPORT.                         *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CSNEWPG *)
(**)
  PROCEDURE CSNEWPG(VAR CSRFILE     : TEXT;
                    VAR PAGE_NUMBER : INTEGER;
                    VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    $FUNCTION:                                                          *)
(*         THIS ROUTINE CREATES A NEW PAGE IN THE CONCEPTUAL SCHEMA       *)
(*         REPORT.                                                        *)
(*                                                                        *)
(*    $DESCRIPTION OF ARGUMENTS:                                          *)
(*         NAME           I/O  DESCRIPTION                               *)
(*         ====           ===  ===========                               *)
(*         CSRFILE        I/O  OUTPUT FILE                               *)
(*         PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                       *)
(*         LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE            *)
(*                                                                        *)
(*    $COMMONS:                                                           *)
(*         NONE                                                           *)
(*                                                                        *)
(*    $ENVIRONMENT:                                                       *)
(*         LANGUAGE: IBM PASCAL                                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                        *)
(*    $EXECUTION PROCEDURE:                                               *)
(*         INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT            *)
(*                                                                        *)
(*    $PROCESSING DESCRIPTION:                                            *)
(*                                                                        *)
(*    INCREMENT THE PAGE NUMBER.                                          *)
(*    CREATE A NEW PAGE IN THE FILE.                                      *)
(*    WRITE OUT TO THE FILE THE CURRENT PAGE NUMBER.                      *)
(*    INCREMENT THE LINE COUNTER.                                         *)
(*                                                                        *)
(*    $COMMENTS:                                                          *)
(*                                                                        *)
(*    $CHANGE CONTROL:                                                    *)
(*                                                                        *)
```

```
(* %INCLUDE CSPTRWRT *)
(**)
  PROCEDURE CSPTRWRT(VAR IRC         : RET_REC;
                     VAR POINTER_KEY : ENTKEY;
                     VAR CSRFILE     : TEXT;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER;
                     VAR INDENT      : INTEGER;
                     VAR PAGE_TYPE   : PAGES);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A POINTER DEFINITION                    *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC             O   INTERNAL RETURN CODE                        *)
(*        POINTER_KEY     I   POINTER KEY                                 *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                          *)
(*        INDENT         I/O  NUMBER OF SPACES TO INDENT                  *)
(*        PAGE_TYPE      I/O  TYPE OF REPORT PAGE                         *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   GET THE POINTER'S ADB.                                               *)
(*   WRITE OUT TO THE FILE 'POINTER TO ('.                                *)
(*   CREATE A NEW PAGE, IF NECESSARY.                                     *)
(*   SET THE LIST OF CONSTITUENTS OF THE POINTER TO BE READ IN THE        *)
(*     FORWARD DIRECTION.                                                 *)
(*   COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                        *)
```

```
(*   FOR Y = 1 TO THE NUMBER OF CONSTITUENTS IN THE LIST                *)
(*     CREATE A NEW PAGE, IF NECESSARY.                                 *)
(*     READ A CONSTITUENT FROM THE LIST OF CONSTITUENTS.                *)
(*     GET THE CONSTITUENT'S ADB.                                       *)
(*     WRITE OUT TO THE FILE A CONSTITUENT (ENTITY OR CLASS)            *)
(*     INCREMENT THE LINE COUNTER.                                      *)
(*   WRITE OUT TO THE FILE ')'.                                         *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
```

```
(* %INCLUDE CSRELWRT *)
(**)
  PROCEDURE CSRELWRT(VAR IRC         : RET_REC;
                     VAR REAL_KEY    : ENTKEY;
                     VAR CSRFILE     : TEXT;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A REAL DEFINITION                       *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC             O   INTERNAL RETURN CODE                       *)
(*        REAL_KEY        I   REAL KEY                                   *)
(*        CSRFILE        I/O  THE OUTPUT FILE                            *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                         *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   GET THE INTEGER'S ADB.                                               *)
(*   WRITE OUT TO THE FILE THE REAL DEFINITION.                           *)
(*   INCREMENT THE LINE COUNTER.                                          *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE CSRPTCVR *)
(**)
  PROCEDURE CSRPTCVR(VAR CSRFILE : TEXT);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE PRINTS OUT THE REPORT COVER FOR THE CONCEPTUAL *)
(*        SCHEMA.                                                    *)
(*                                                                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        CSRFILE       I/O  OUTPUT FILE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   CREATE A NEW PAGE.                                              *)
(*   WRITE 'CONCEPTUAL SCHEMA REPORT'.                               *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE CSSTGWRT *)
(**)
  PROCEDURE CSSTGWRT(VAR IRC         : RET_REC;
                     VAR STRING_KEY  : ENTKEY;
                     VAR CSRFILE     : TEXT;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT A STRING DEFINITON                       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   INTERNAL RETURN CODE                        *)
(*        STRING_KEY      I   STRING KEY                                  *)
(*        CSRFILE        I/O  THE OUTPUT FILE                             *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                          *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   GET THE STRING'S ADB.                                                 *)
(*   WRITE OUT TO THE FILE THE STRING DEFINITION.                          *)
(*   INCREMENT THE LINE COUNTER.                                           *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE CSSTRWRT *)
(**)
  PROCEDURE CSSTRWRT(VAR IRC           : RET_REC;
                     VAR STRUCTURE_KEY : ENTKEY;
                     VAR CSRFILE       : TEXT;
                     VAR PAGE_NUMBER   : INTEGER;
                     VAR LINE_COUNT    : INTEGER;
                     VAR PAGE_TYPE     : PAGES);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT A STUCTURE DEFINITION                    *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   INTERNAL RETURN CODE                        *)
(*        STRUCTURE_KEY   I   STRUCTURE KEY                               *)
(*        CSRFILE        I/O  THE OUTPUT FILE                             *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT     I/O  CURRENT LINE COUNT                          *)
(*        PAGE_TYPE      I/O  TYPE OF REPORT PAGE                         *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   GET THE STRUCTURE'S ADB.                                              *)
(*   WRITE OUT TO THE FILE 'STRUCTURE'.                                    *)
(*   INCREMENT THE LINE COUNTER.                                           *)
(*   SET THE LIST OF CONSTITUENTS TO BE READ IN THE FORWARD DIRECTION.     *)
(*   COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                         *)
(*   FOR X = 1 TO THE NUMBER OF CONSTITUENTS IN THE LIST                   *)
(*     READ A CONSTITUENT FROM THE LIST.                                   *)
(*     GET THE CONSTITUENT'S ADB.                                          *)
(*     CREATE A NEW PAGE, IF NECESSARY.                                    *)
(*     WRITE OUT TO THE FILE THE FIELD NAME.                               *)
(*     SET THE TYPE LIST TO BE READ IN THE FORWARD DIRECTION.              *)
```

```
(*     GET THE TYPE'S KEY.                                                  *)
(*     GET THE TYPE'S ADB.                                                  *)
(*     CASE TYPE_ADB.ENT_KIND OF                                            *)
(*       INTEGER      :  WRITE OUT THE INTEGER DEFINITION.                  *)
(*       REAL         :  WRITE OUT THE REAL DEFINITION.                     *)
(*       STRING       :  WRITE OUT THE STRING DEFINITION.                   *)
(*       LOGICAL      :  WRITE OUT THE LOGICAL DEFINITION.                  *)
(*       ARRAY        :  WRITE OUT THE ARRAY DEFINITION.                    *)
(*       DEFINED TYPE :  WRITE OUT THE DEFINED TYPE DEFINITION.             *)
(*       POINTER      :  WRITE OUT THE POINTER DEFINITION.                  *)
(*     CREATE A NEW PAGE, IF NECESSARY.                                     *)
(*   WRITE OUT TO THE FILE 'END;'.                                          *)
(*                                                                          *)
(*   $COMMENTS:                                                             *)
(*                                                                          *)
(*   $CHANGE CONTROL:                                                       *)
(*                                                                          *)
```

```
(* %INCLUDE CSSUBHDG *)
(**)
  PROCEDURE CSSUBHDG(VAR CSRFILE     : TEXT;
                     VAR HEADING     : HEADING_TYPE;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A SUBSCHEMA HEADING ON A NEW PAGE.      *)
(*                                                                        *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        HEADING         I   DEFINITION OR INDEX HEADING                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE              *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   CREATE A NEW PAGE.                                                   *)
(*   WRITE OUT TO THE FILE THE APPROPRIATE HEADING (DEFINITION OR         *)
(*     INDEX).                                                            *)
(*   INCREMENT THE LINE COUNTER.                                          *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE CSSUBWRT *)
(**)
  PROCEDURE CSSUBWRT(VAR IRC            : RET_REC;
                     VAR SUBSCHEMA_LIST : LISTKEY;
                     VAR CSRFILE        : TEXT;
                     VAR PAGE_NUMBER    : INTEGER;
                     VAR CURRENT_PAGE   : PAGE_PTR);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES OUT SUBSCHEMA DEFINITIONS TO A FILE.   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC            O   INTERNAL RETURN CODE                  *)
(*        SUBSCHEMA_LIST I   ALPHABETIZED LIST OF CLASSES          *)
(*        CSRFILE       I/O  THE OUTPUT FILE                       *)
(*        PAGE_NUMBER   I/O  THE CURRENT PAGE NUMBER               *)
(*        CURRENT_PAGE  I/O  POINTS TO THE CURRENT PAGE RECORD IN  *)
(*                           THE CHAIN                             *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT        *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   SET THE LIST OF SUBSCHEMAS TO BE READ IN THE FORWARD DIRECTION. *)
(*   COUNT THE NUMBER OF SUBSCHEMAS IN THE LIST.                     *)
(*   FOR X = 1 TO THE NUMBER OF SUBSCHEMAS IN THE LIST               *)
(*     WRITE OUT TO THE FILE THE SUBSCHEMA HEADING.                  *)
(*     UPDATE THE PAGE RECORD CHAIN.                                 *)
(*     READ A SUBSCHEMA FROM THE LIST OF SUBSCHEMAS.                 *)
(*     GET THE SUBSCHEMA'S ADB.                                      *)
(*     WRITE OUT TO THE FILE THE SUBSCHEMA NAME.                     *)
(*     SET THE LIST OF CONSTITUENTS TO BE READ IN THE FORWARD        *)
(*       DIRECTION.                                                  *)
(*     COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                 *)
```

```
(*     FOR Y = 1 TO THE NUMBER OF CONSTITUENTS IN THE LIST          *)
(*       READ A CONSTITUENT FROM THE LIST OF CONSTITUENTS           *)
(*       GET THE CONSTITUENT'S ADB.                                 *)
(*       WRITE OUT TO THE FILE THE HEADING ON A NEW PAGE, IF        *)
(*         NECESSARY.                                               *)
(*       WRITE OUT TO THE FILE THE CONSTITUENT (ENTITY OR CLASS).   *)
(*     WRITE OUT TO THE FILE 'END;'.                                *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
```

```
(* %INCLUDE CSSUPHDG *)
(**)
  PROCEDURE CSSUPHDG(VAR CSRFILE     : TEXT;
                     VAR HEADING     : HEADING_TYPE;
                     VAR PAGE_NUMBER : INTEGER;
                     VAR LINE_COUNT  : INTEGER);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES OUT A SUPERTYPE HEADING ON A NEW PAGE.      *)
(*                                                                        *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        CSRFILE        I/O  OUTPUT FILE                                 *)
(*        HEADING         I   DEFINITION OR INDEX HEADING                 *)
(*        PAGE_NUMBER    I/O  CURRENT PAGE NUMBER                         *)
(*        LINE_COUNT      O   LINE COUNT ON THE CURRENT PAGE              *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*   CREATE A NEW PAGE.                                                   *)
(*   WRITE OUT TO THE FILE THE APPROPRIATE SUPERTYPE HEADING              *)
(*     DEFINITION OR INDEX.                                               *)
(*   INCREMENT THE LINE COUNTER.                                          *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID     *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                 *)
(*                                                                        *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER          GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                     *)
(*    REVISED: MM/DD/YY          I. M. THECHANGER             DBMA     *)
(*                                                                     *)
(*    ORIGINATED: 09/28/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE CSSUPHDG *)
```

```
(* %INCLUDE CSSUPWRT *)
(**)
  PROCEDURE CSSUPWRT(VAR IRC             : RET_REC;
                     VAR SUPERTYPE_LIST  : LISTKEY;
                     VAR CSRFILE         : TEXT;
                     VAR PAGE_NUMBER     : INTEGER;
                     VAR CURRENT_PAGE    : PAGE_PTR);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*         THIS ROUTINE WRITES OUT SUPERTYPE DEFINITIONS TO A FILE         *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME           I/O  DESCRIPTION                                 *)
(*         ====           ===  ===========                                 *)
(*         IRC             O   INTERNAL RETURN CODE                        *)
(*         SUPERTYPE_LIST  I   ALPHABETIZED LIST OF SUPERTYPES             *)
(*         CSRFILE        I/O  THE OUTPUT FILE                             *)
(*         PAGE_NUMBER    I/O  THE CURRENT PAGE NUMBER                     *)
(*         CURRENT_PAGE   I/O  POINTS TO THE CURRENT PAGE RECORD IN        *)
(*                             THE CHAIN.                                  *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*         NONE                                                            *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT             *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*                                                                         *)
(*    SET THE LIST OF ENTITIES TO BE READ IN THE FORWARD DIRECTION.        *)
(*    COUNT THE NUMBER OF ENTITIES IN THE LIST.                            *)
(*    FOR X = 1 TO THE NUMBER OF ENTITIES IN THE LIST                      *)
(*      WRITE OUT TO THE FILE THE ENTITY HEADING.                          *)
(*      UPDATE THE PAGE RECORD CHAIN.                                      *)
(*      READ AN ENTITY FROM THE LIST OF ENTITIES.                          *)
(*      GET THE ENTITY'S ADB.                                              *)
(*      WRITE OUT TO THE FILE THE ENTITY NAME AND NUMBER.                  *)
(*      SET THE LIST OF CONSTITUENTS OF THE ENTITY TO BE READ IN THE       *)
(*        FORWARD DIRECTION.                                               *)
(*      COUNT THE NUMBER OF CONSTITUENTS IN THE LIST.                      *)
```

```
(*     FOR Y = 1 TO THE NUMBER OF CONSTITUENTS IN THE LIST            *)
(*       READ A CONSTITUENT FROM THE LIST.                            *)
(*       GET THE CONSTITUENT'S ADB.                                   *)
(*       CREATE A NEW PAGE, IF NECESSARY.                             *)
(*       WRITE OUT TO THE FILE A FIELD NAME.                          *)
(*       SET THE TYPE OF FIELD TO BE READ IN THE FORWARD DIRECTION.   *)
(*       GET THE TYPE'S KEY.                                          *)
(*       GET THE TYPE'S ADB.                                          *)
(*       CASE TYPE_ADB.ENT_KIND OF                                    *)
(*         INTEGER      :  WRITE OUT THE INTEGER DEFINITION           *)
(*         REAL         :  WRITE OUT THE REAL DEFINITION              *)
(*         STRING       :  WRITE OUT THE STRING DEFINITION            *)
(*         LOGICAL      :  WRITE OUT THE LOGICAL DEFINITION           *)
(*         ARRAY        :  WRITE OUT THE ARRAY DEFINITION             *)
(*         DEFINED TYPE :  WRITE OUT THE DEFINED TYPE DEFINITION      *)
(*         POINTER      :  WRITE OUT THE POINTER DEFINITION           *)
(*     WRITE OUT TO THE FILE 'END;'.                                  *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                    *)
(*     ORIGINATED: 09/28/87       C. H. MOHME               DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE CSSUPWRT *)
```

```
(* %INCLUDE CSTYPWRT *)
(**)
  PROCEDURE CSTYPWRT(VAR IRC          : RET_REC;
                     VAR DEF_TYP_LIST : LISTKEY;
                     VAR CSRFILE      : TEXT;
                     VAR PAGE_NUMBER  : INTEGER);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES OUT THE DEFINED TYPE DEFINITIONS TO          *)
(*        A FILE.                                                          *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   INTERNAL RETURN CODE                         *)
(*        DEF_TYP_LIST    I   ALPHABETIZED LIST OF DEFINED TYPES           *)
(*        CSRFILE        I/O  THE OUTPUT FILE                             *)
(*        PAGE_NUMBER    I/O  THE CURRENT PAGE NUMBER                     *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE CONCEPTUAL SCHEMA REPORT              *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*   SET THE LIST OF DEFINED TYPES TO BE READ IN THE FORWARD               *)
(*     DIRECTION.                                                          *)
(*   COUNT THE NUMBER OF DEFINED TYPES IN THE LIST.                        *)
(*   WRITE OUT TO THE FILE THE DEFINED TYPE DEFINITION HEADING.            *)
(*   FOR X = 1 TO THE NUMBER OF DEFINED TYPES IN THE LIST                  *)
(*     READ A DEFINED TYPE FROM THE LIST.                                  *)
(*     GET THE DEFINED TYPE'S ADB.                                         *)
(*     WRITE OUT THE DEFINED TYPE HEADING, IF NECESSARY.                   *)
(*     WRITE OUT TO THE FILE THE DEFINED TYPE NAME.                        *)
(*     SET THE LIST OF CONSTITUENTS TO BE READ IN THE FORWARD              *)
(*       DIRECTION.                                                        *)
(*     READ THE CONSTITUENT FROM THE LIST OF CONSTITUENTS.                 *)
(*     GET THE CONSTITUENT'S ADB.                                          *)
```

```
(*     CASE CONSTITUENT_ADB.ENT_KIND OF                                  *)
(*       INTEGER      :  WRITE OUT THE INTEGER DEFINITION.               *)
(*       REAL         :  WRITE OUT THE REAL DEFINITION.                  *)
(*       STRING       :  WRITE OUT THE STRING DEFINITION.                *)
(*       LOGICAL      :  WRITE OUT THE LOGICAL DEFINITION.               *)
(*       ARRAY        :  WRITE OUT THE ARRAY DEFINITION.                 *)
(*       DEFINED TYPE :  WRITE OUT THE DEFINED TYPE DEFINITION.          *)
(*       POINTER      :  WRITE OUT THE POINTER DEFINITION.               *)
(*       ENUMERATION  :  WRITE OUT THE ENUMERATION DEFINITION.           *)
(*       STRUCTURE    :  WRITE OUT THE STRUCTURE DEFINITION.             *)
(*     INCREMENT THE LINE COUNTER.                                       *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
```

```
(* BEGIN %INCLUDE DDABNDS ********************************************)
(*                                                                    *)
PROCEDURE DDABNDS ( VAR   DDFILE          : T_FILE_VARIANT;
                    CONST NO_OF_DIMEN     : INTEGER;
                    CONST STARTING_ARRAY_POSITION : INTEGER;
                    VAR   POINTER         : T_VARIANT_POINTER;
                    VAR   ENTITY_SIZE     : INTEGER;
                    VAR   ENTITY_POSITION : INTEGER;
                    CONST ENUM_INDEX      : INTEGER );
          SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        WRITE THE LOW-BOUND AND UPPER-BOUND FOR THE ARRAY ATTRIBUTE *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        DDFILE             O   DATA DICTIONARY SEQUENTIAL FILE      *)
(*        NO_OF_DIMEN        I   NUMBER OF ARRAY DIMENSION            *)
(*        STARTING_ARRAY_PO  I   STARTING POSITION IN THE ARRAY TABLE *)
(*        POINTER            I   POINTER TO ARRAY TABLE               *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION  *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION       *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH THE NUMBER OF DIMENSIONS                       *)
(*           WRITE LOW-BOUND AND UPPER-BOUND                          *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 23 MARCH 1987, M. H. CHOI, DBMA                 *)
(*                                                                    *)
(* END %INCLUDE DDABNDS  **********************************************)
```

```
(* BEGIN %INCLUDE DDADB ************************************************)
(*                                                                      *)
PROCEDURE DDADB ( VAR   DDFILE          : T_FILE_VARIANT;
                  CONST RUNTIME         : T_RUN_TIME;
                  CONST ENTRY           : INTEGER;
                  CONST PS_ORDER        : T_PS_ORDER;
                  VAR   ENTITY_SIZE     : INTEGER;
                  VAR   ENTITY_POSITION: INTEGER );
        SUBPROGRAM;
(*                                                                      *)
(*   $FUNCTION:                                                         *)
(*        WRITE THE BASIC RECORD OF AN ENTITY TO A SEQUENTIAL FILE      *)
(*                                                                      *)
(*   $DESCRIPTION OF ARGUMENTS:                                         *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        DDFILE             I   DATA DICTIONARY SEQUENTIAL FILE       *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION        *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION         *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER         *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION   *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION        *)
(*                                                                      *)
(*   $COMMONS:                                                          *)
(*                                                                      *)
(*   $ENVIRONMENT:                                                      *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*                                                                      *)
(*   $EXECUTION PROCEDURE:                                              *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM      *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*        WRITE BASIC RECORD                                            *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*        ORIGINATED: 17 MARCH 1987, M. H. CHOI, DBMA                   *)
(*                                                                      *)
(* END %INCLUDE DDADB ***************************************************)
```

```
(* BEGIN %INCLUDE DDARRAY ********************************************)
(*                                                                   *)
PROCEDURE DDARRAY ( VAR   DDFILE         : T_FILE_VARIANT;
                    CONST RUNTIME        : T_RUN_TIME;
                    CONST ENTRY          : INTEGER;
                    CONST PS_ORDER       : T_PS_ORDER;
                    VAR   ENTITY_SIZE    : INTEGER;
                    VAR   ENTITY_POSITION : INTEGER );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        WRITE THE ARRAY ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        DDFILE             O   DATA DICTIONARY SEQUENTIAL FILE     *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION       *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER       *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION      *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        OBTAIN THE NUMBER OF DIMENSIONS                            *)
(*        OBTAIN THE STARTING POSITION OF ARRAY TABLE                *)
(*        CASE DATA TYPE OF                                          *)
(*          IN-ADB : WRITE BASIC DEFINITION                          *)
(*                   DDALIST ( EXTERNAL SUBPROGRAM FOR LOW-BOUND AND *)
(*                             UPPER-BOUND )                         *)
(*          IN-CL  : WRITE BASIC DEFINITION                          *)
(*                   DDCL ( EXTERNAL SUBPROGRAM FOR CONSTITUENT LIST) *)
(*                   DDALIST ( EXTERNAL SUBPROGRAM FOR LOW-BOUND AND *)
(*                             UPPER-BOUND )                         *)
(*        END CASE                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 17 MARCH 1987, M. H. CHOI, DBMA                *)
(*                                                                   *)
(* END %INCLUDE DDARRAY  *********************************************)
```

```
(* BEGIN %INCLUDE DDCL ************************************************)
(*                                                                     *)
PROCEDURE DDCL ( VAR   DDFILE          : T_FILE_VARIANT;
                 CONST RUNTIME         : T_RUN_TIME;
                 CONST ENTRY           : INTEGER;
                 VAR   ENTITY_SIZE     : INTEGER;
                 VAR   ENTITY_POSITION: INTEGER );
        SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        WRITE THE CONSTITUENT REFERENCES OF AN ENTITY TO A           *)
(*        SEQUENTIAL FILE                                              *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        DDFILE             O   DATA DICTIONARY SEQUENTIAL FILE       *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION        *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION         *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION   *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION        *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM     *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        IF NOT ARRAY ATTRIBUTE THEN                                  *)
(*           WRITE BASIC DEFINITION                                    *)
(*        END IF                                                       *)
(*        OBTAIN THE NUMBER OF ELIGIBLE KINDS                          *)
(*        OBTAIN STARTING POSITION OF CONSTITUENT LIST TABLE           *)
(*        LOOP THROUGH THE NUMBER OF ELIGIBLE KINDS                    *)
(*           WRITE ELIGIBLE KIND IN THE CONSTITUENT LIST TABLE         *)
(*        END LOOP                                                     *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 17 MARCH 1987, M. H. CHOI, DBMA                  *)
(*                                                                     *)
(* END %INCLUDE DDCL **************************************************)
```

```
(* BEGIN %INCLUDE DDCLASS ********************************************)
(*                                                                   *)
PROCEDURE DDCLASS ( CONST SUBSCHEMA_KEY : ENTKEY;
                    VAR   DDFILE        : T_FILE_VARIANT;
                    VAR   DDINX         : T_INX_FILE;
                    VAR   ENTITY_POSITION: INTEGER;
                    VAR   XRC           : INTEGER );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        WRITE THE CLASS KINDS TO A SEQUENTIAL FILE                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        DDFILE             I   DATA DICTIONARY SEQUENTIAL FILE     *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION       *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER       *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION      *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        WRITE BASIC RECORD                                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 08 MAY 1987, M. H. CHOI, DBMA                  *)
(*                                                                   *)
(* END %INCLUDE DDCLASS **********************************************)
```

```
(* BEGIN %INCLUDE DDENTITY ********************************************)
(*                                                                     *)
PROCEDURE DDENTITY ( CONST SUBSCHEMA_KEY  : ENTKEY;
                     VAR   DDFILE         : T_FILE_VARIANT;
                     VAR   DDINX          : T_INX_FILE;
                     VAR   ENTITY_POSITION: INTEGER;
                     VAR   XRC            : INTEGER );
        SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        WRITE THE ENTITY KINDS TO A SEQUENTIAL FILE                  *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        DDFILE             I   DATA DICTIONARY SEQUENTIAL FILE       *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION        *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION         *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER         *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION   *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION        *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM     *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        WRITE BASIC RECORD                                           *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 08 MAY 1987, M. H. CHOI, DBMA                    *)
(*                                                                     *)
(* END %INCLUDE DDENTITY **********************************************)
```

```
(* BEGIN %INCLUDE DDENUM ********************************************)
(*                                                                   *)
PROCEDURE DDENUM ( VAR   DDFILE          : T_FILE_VARIANT;
                   CONST RUNTIME         : T_RUN_TIME;
                   CONST ENTRY           : INTEGER;
                   CONST PS_ORDER        : T_PS_ORDER;
                   VAR   ENTITY_SIZE     : INTEGER;
                   VAR   ENTITY_POSITION: INTEGER;
                   CONST ENUM_TABLE_INDEX : INTEGER;
                   CONST NO_OF_ENUM      : INTEGER );
         SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        WRITE THE ENUMERATION ATTRIBUTE OF AN ENTITY TO A          *)
(*        SEQUENTIAL FILE                                            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        DDFILE             O   DATA DICTIONARY SEQUENTIAL FILE     *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION       *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER       *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION      *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        WRITE BASIC DEFINITION                                     *)
(*        OBTAIN THE NUMBER OF ENUMERATION VALUES                    *)
(*        OBTAIN THE STARTING POSITION OF ENUMERATION VALUE TABLE    *)
(*        LOOP THROUGH THE NUMBER OF ENUMERATION VALUES              *)
(*           WRITE THE ENUMERATION VALUE FROM THE TABLE              *)
(*        END LOOP                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 17 MARCH 1987, M. H. CHOI, DBMA                *)
(*                                                                   *)
(* END %INCLUDE DDENUM  *********************************************)
```

```
(* BEGIN %INCLUDE DDREPORT ********************************************)
(*                                                                     *)
PROCEDURE DDREPORT ( VAR   SUBSCHEMA_KEY     : ENTKEY;
                     VAR    RETURN_CODE      : INTEGER );
         EXTERNAL;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        WRITE THE DATA DICTIONARY IN CHARACTER FORM.                 *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME                  I/O  DESCRIPTION                     *)
(*        ====                  ===  ===========                     *)
(*        SUBSCHEMA_KEY          I   INPUT VALUE OF ARBITRARY SIZE    *)
(*        RETURN_CODE            O   RETURN CODE                      *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM     *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        SORT ENTITIES BY KIND NUMBER                                 *)
(*        LOOP THROUGH THE LIST OF ENTITIES                            *)
(*           REQUEST ENTITY DEFINITION                                 *)
(*           LOOP THROUGH THE NUMBER OF ATTRIBUTES IN THE DEFINITION   *)
(*              CASE DATA TYPE OF                                      *)
(*                 IN-ADB   : DDADB ( EXTERNAL SUBPROGRAM )            *)
(*                 IN-ENUM  : DDENUM ( EXTERNAL SUBPROGRAM )           *)
(*                 IN-CL    : DDCL ( EXTERNAL SUBPROGRAM )             *)
(*                 IN-ARRAY : DDARRAY ( EXTERNAL SUBPROGRAM )          *)
(*              END CASE                                               *)
(*           END LOOP                                                  *)
(*        END LOOP                                                     *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 23 MARCH 1987, M. H. CHOI, DBMA                  *)
(*                                                                     *)
(* END %INCLUDE DDREPORT **********************************************)
```

```
(* BEGIN %INCLUDE DDWRITE  ********************************************)
(*                                                                     *)
PROCEDURE DDWRITE ( CONST  RUNTIME         : T_RUN_TIME;
                    VAR    DDFILE          : T_FILE_VARIANT;
                    VAR    DDINX           : T_INX_FILE;
                    VAR    ENTITY_POSITION: INTEGER );
        SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        WRITE THE ENTITY DEFINITIONS TO A SEQUENTIAL FILE            *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        DDFILE             I   DATA DICTIONARY SEQUENTIAL FILE       *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION        *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION         *)
(*        PS_ORDER           I   LIST OF PHYSICAL SCHEMA ORDER         *)
(*        ENTITY_SIZE        O   NUMBER OF RECORDS IN THE DEFINITION   *)
(*        ENTITY_POSITION    O   FIRST RECORD OF THE DEFINITION        *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM     *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        WRITE BASIC RECORD                                           *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 08 MAY 1987, M. H. CHOI, DBMA                    *)
(*                                                                     *)
(* END %INCLUDE DDWRITE  **********************************************)
```

```
(* %INCLUDE DEFADD *)
(**)
  PROCEDURE DEFADD(VAR IRC             :  RET_REC;
                   VAR IDENTIFIER      :  T_NAME;
                   VAR KIND            :  INTEGER;
                   VAR REFERENCE_KEY   :  ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that adds an unresolved entity    *)
(*         reference to the list of backpatch entities.              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME            I/O  DESCRIPTION                            *)
(*       ====            ===  ===========                            *)
(*       IRC              O   INTERNAL RETURN CODE                   *)
(*       IDENTIFIER       I   NAME OF ENTITY OR DEFINED TYPE         *)
(*       KIND             I   KIND NUMBER OF ENTITY                  *)
(*       REFERENCE_KEY    I   KEY OF "ENTITY" WITH UNRESOLVED        *)
(*                            REFERENCE                              *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    PERFORM INITIALIZATIONS.                                       *)
(*    DETERMINE IF ANY BACKPATCH ENTITIES EXIST IN THE MODEL.        *)
(*    IF NO BACKPATCH ENTITIES EXIST, CREATE ONE.                    *)
(*    IF BACKPATCH ENTITIES EXIST, DETERMINE IF ONE HAS THE SAME     *)
(*      NAME OR KIND AS THE ENTITY INPUT.                            *)
(*    IF A BACKPATCH ENTITY HAS THE SAME NAME OR KIND, THEN ADD A    *)
(*      REFERENCE AS A CONSTITUENT OF THAT BACKPATCH ENTITY.         *)
(*    IF NO EXISTING BACKPATCH ENTITY HAS THE SAME NAME OR KIND,     *)
(*      THEN CREATE A NEW BACKPATCH ENTITY.                          *)
(*                                                                   *)
(*                                                                   *)
(*                                                                   *)
(*                                                                   *)
(*                                                                   *)
(*                                                                   *)
```

```
(*                                                                    *)
(*                                                                    *)
(*                                                                    *)
(*                                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 04/22/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFADD *)
```

```
(* %INCLUDE DEFARR *)
(**)
  PROCEDURE DEFARR(VAR IRC             :  RET_REC;
                   VAR ENT_KIND        :  INTEGER;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*         Batch Interface routine that processes an array definition.*)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*       IRC              O   INTERNAL RETURN CODE                         *)
(*       ENT_KIND         I   THE KIND OF ENTITY BEING CONSTRUCTED         *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                            *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                       *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                 *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE              *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                              *)
(*       TEXT            I/O  OUTPUT FILE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*    INITIALIZE VARIABLES                                                 *)
(*    GET ARRAY LOWER BOUND AND VERIFY RANGE                               *)
(*    GET ARRAY UPPER BOUND AND VERIFY RANGE                               *)
(*    PUSH ARRAY TRANSACTION ONTO STACK                                    *)
(*    GET THE ARRAY TYPE                                                   *)
(*    IF THE ARRAY TYPE IS BASIC TYPE THEN                                 *)
(*      CREATE BASIC CONSTITUENT                                           *)
```

```
(*    ELSE                                                              *)
(*      IF THE ARRAY TYPE IS DEFINED TYPE THEN                          *)
(*        CREATE DEFINED TYPE CONSTITUENT                               *)
(*    PRINT ERRORS AS APPROPRIATE                                       *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA      *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFARR *)
```

```
(* %INCLUDE DEFATT *)
(**)
  PROCEDURE DEFATT(VAR IRC            :  RET_REC;
                   VAR TRANS_STACK    :  TRANSPTR;
                   VAR TOKEN          :  T_TOKEN;
                   VAR TOKEN_VALUE    :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION :  INTEGER;
                   VAR TOKEN_LENGTH   :  INTEGER;
                   VAR FIELDTYPE      :  T_FIELDTYPE;
                   VAR SUBSCHEMA_FLAG :  BOOLEAN;
                   VAR SUB_ENT_HEAD   :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST   :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG     :  BOOLEAN;
                   VAR CLS_ENT_HEAD   :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST   :  ENTITY_LIST_PTR;
                   VAR REPORT1        :  TEXT);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*          Batch Interface routine that processes an attribute            *)
(*          definition.                                                    *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME            I/O  DESCRIPTION                                 *)
(*        ====            ===  ==========                                  *)
(*        IRC              O   INTERNAL RETURN CODE                        *)
(*        TRANS_STACK     I/O  TRANSACTION STACK                           *)
(*        TOKEN           I/O  TOKEN FROM BATCH INPUT                      *)
(*        TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                *)
(*        TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE             *)
(*        TOKEN_LENGTH    I/O  LENGTH OF TOKEN                             *)
(*        FIELDTYPE       I/O  TYPE OF ATTRIBUTE (ENTITY OR GLOBAL)        *)
(*        SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                             THE SUBSCHEMA                               *)
(*        SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES        *)
(*        SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN THE LIST OF     *)
(*                             SUBSCHEMA ENTITIES                          *)
(*        CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                             THE CLASS                                   *)
(*        CLS_ENT_HEAD    I/O  POINTS TO THE LIST OF CLASS ENTITIES        *)
(*        CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN THE LIST  OF    *)
(*                             CLASS ENTITIES                              *)
(*        REPORT1         I/O  OUTPUT FILE                                 *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*                                                                         *)
```

```
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*    INITIALIZE VARIABLES                                                *)
(*    STORE ATTRIBUTE NAME                                                *)
(*    DETERMINE IF THE ATTRIBUTE NAME IS UNIQUE                           *)
(*    IF THE ATTRIBUTE TYPE IS GLOBAL OR STRUCTURE, THEN WE WILL WANT     *)
(*      A LIST OF ALL OF THE ATTRIBUTES SO THAT WE CAN VERIFY THAT        *)
(*      THE ATTRIBUTE NAME IS UNIQUE AMONG ALL MODELED ATTRIBUTES.        *)
(*      IF THE ATTRIBUTE TYPE IS ENTITY THEN WE WILL WANT A LIST OF       *)
(*      ALL OF THE GLOBAL AND STRUCTURE ATTRIBUTES SO THAT WE CAN         *)
(*      VERIFY THAT THE ATTRIBUTE NAME IS UNIQUE AMONG ALL MODELED        *)
(*      GLOBAL AND STRUCTURE ATTRIBUTES.                                  *)
(*    VERIFY THAT THE ATTRIBUTE NAME IS UNIQUE AMONG THOSE ATTRIBUTES     *)
(*      ON THE STACK.                                                     *)
(*    VERIFY THAT THE ATTRIBUTE NAME IS UNIQUE AMONG THOSE ATTRIBUTES     *)
(*      ALREADY MODELED.                                                  *)
(*    IF THE ATTRIBUTE NAME IS UNIQUE THEN PUT THE ATTRIBUTE DATA ON-     *)
(*      TO THE STACK AND GET NEXT TOKEN.                                  *)
(*    DETERMINE IF ANOTHER ATTRIBUTE FOLLOWS THE CURRENT ONE              *)
(*      (SEPARATED BY A COMMA) OR IF THE TYPE DEFINITION FOR              *)
(*      THE ATTRIBUTE(S) IS NEXT.                                         *)
(*    IF ANOTHER ATTRIBUTE FOLLOWS THEN                                   *)
(*      DEFINE THE ATTRIBUTE                                              *)
(*    ELSE                                                                *)
(*      IF ATTRIBUTE TYPE IS BASIC TYPE THEN                              *)
(*        CREATE BASIC TYPE                                               *)
(*      ELSE                                                              *)
(*        IF ATTRIBUTE TYPE IS DEFINED TYPE THEN                          *)
(*          CREATE DEFINED TYPE CONSTITUENT                               *)
(*    PRINT ERRORS AS APPROPRIATE                                         *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA        *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE DEFATT *)
```

```
(* %INCLUDE DEFBAS *)
(**)
  PROCEDURE DEFBAS(VAR IRC            :  RET_REC;
                   VAR ENT_KIND       :  INTEGER;
                   VAR TRANS_STACK    :  TRANSPTR;
                   VAR TOKEN          :  T_TOKEN;
                   VAR TOKEN_VALUE    :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION :  INTEGER;
                   VAR TOKEN_LENGTH   :  INTEGER;
                   VAR REPORT1        :  TEXT);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that processes a primitive data   *)
(*         type definition.                                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*       IRC              O   INTERNAL RETURN CODE                  *)
(*       ENT_KIND         O   THE KIND OF ENTITY BEING CONSTRUCTED  *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       TOKEN_LOCATION  I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       TOKEN_LENGTH    I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       REPORT1         I/O  OUTPUT FILE                           *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT   *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*                                                                  *)
(*    INITIALIZE VARIABLES                                          *)
(*    IF BASIC TYPE IS INTEGER THEN                                 *)
(*      GET INTEGER PRECISION                                       *)
(*      PUSH INTEGER DATA ONTO TRANSACTION STACK                    *)
(*    IF BASIC TYPE IS REAL THEN                                    *)
(*      GET REAL PRECISION                                          *)
(*      PUSH REAL DATA ONTO TRANSACTION STACK                       *)
```

```
(*    IF BASIC TYPE IS STRING THEN                                    *)
(*      GET STRING PRECISION                                          *)
(*      PUSH STRING DATA ONTO TRANSACTION STACK                       *)
(*    IF BASIC TYPE IS LOGICAL THEN                                   *)
(*      PUSH LOGICAL DATA ONTO TRANSACTION STACK                      *)
(*    IF BASIC TYPE IS ARRAY THEN                                     *)
(*      DEFINE ARRAY                                                  *)
(*    IF BASIC TYPE IS POINTER THEN                                   *)
(*      DEFINE POINTER                                                *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFBAS *)
```

```
(* %INCLUDE DEFCLS *)
(**)
  PROCEDURE DEFCLS(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*         Batch Interface routine that processes a class definition.      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*       NAME            I/O  DESCRIPTION                                  *)
(*       ====            ===  ===========                                  *)
(*       IRC              O   INTERNAL RETURN CODE                         *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                            *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                       *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                 *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE              *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                              *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES OR CLASSES ARE         *)
(*                            DEFINED WITHIN A SUBSCHEMA                   *)
(*       SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES         *)
(*       SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF          *)
(*                            SUBSCHEMA ENTITIES                           *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES OR CLASSES ARE         *)
(*                            DEFINED WITHIN A CLASS                       *)
(*       CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES             *)
(*       CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF          *)
(*                            CLASS ENTITIES                               *)
(*       REPORT1         I/O  OUTPUT FILE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
```

```
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*    INITIALIZE VARIABLES                                                *)
(*    GET CLASS NAME AND KIND NUMBER                                      *)
(*    VERIFY UNIQUENESS OF CLASS NAME AND KIND NUMBER                     *)
(*    IF THE NEXT TOKEN IS AN IDENTIFIER OR AN INTEGER, THEN WE KNOW      *)
(*      THAT WE WILL BE GETTING A LIST OF ENTITIES AND CLASSES THAT       *)
(*      ARE TO BE MEMBERS OF THE CLASS.  IF THE SUBSCHEMA FLAG OR         *)
(*      CLASS FLAG IS SET, THEN ADD THE CLASS NAME TO THE APPROPRIATE     *)
(*      LIST(S).  PUSH THE CLASS NAME AND KIND NUMBER ONTO THE PRO-       *)
(*      CESSING STACK.                                                    *)
(*      DETERMINE IF THE ENTITY OR CLASS EXISTS                           *)
(*      IF WE HAVE AN ENTITY OR CLASS KEY THEN WE VERIFY THAT THE         *)
(*        ENTITY OR CLASS IS NOT ALREADY A MEMBER OF THE CLASS.  IF       *)
(*        IT IS NOT A MEMBER, WE ATTACH THE KEY TO THE LIST AND PUSH      *)
(*        A TRANSACTION ONTO THE STACK.                                   *)
(*      IF THE ENTITY OR CLASS CONSTITUENT DOES NOT ALREADY EXIST,        *)
(*        CREATE AN UNRESOLVED ENTITY AND MAKE IT A CONSTITUENT OF        *)
(*        THE CLASS.  IF THE ENTITY OR CLASS CONSTITUENT IS LATER         *)
(*        CREATED, THEN IT WILL REPLACE THE UNRESOLVED ENTITY IN THE      *)
(*        CLASS CONSTITUENT LIST.                                         *)
(*      IF WE HAVE READ IN ALL OF THE ENTITY AND CLASS NAMES, WE PUSH     *)
(*        THE FINAL CLASS TRANSACTION ONTO THE STACK AND PROCESS THE      *)
(*        TRANSACTION STACK.                                              *)
(*    IF WE HAVE ENCOUNTERED AN ENTITY OR CLASS DEFINITION, THEN WE       *)
(*      SET A FLAG TO INDICATE THAT AFTER MODELING THESE ENTITIES         *)
(*      AND CLASSES, WE MUST THEN MODEL THE CLASS.                        *)
(*    ADD THE CLASS TO THE SUBSCHEMA AND CLASS LIST, AS NECESSARY         *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                    *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA        *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE DEFCLS *)
```

```
(* %INCLUDE DEFDEF *)
(**)
  PROCEDURE DEFDEF(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR REPORT1         :  TEXT);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*          Batch Interface routine that processes a defined type    *)
(*          reference.                                               *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*        IRC              O   INTERNAL RETURN CODE                  *)
(*        TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*        TOKEN           I/O  TOKEN FROM BATCH INPUT                *)
(*        TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*        TOKEN_LOCATION   I   LOCATION OF TOKEN IN INPUT LINE       *)
(*        TOKEN_LENGTH     I   LENGTH OF TOKEN                       *)
(*        REPORT1         I/O  OUTPUT FILE                          *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*                                                                   *)
(*     INITIALIZE THE VARIABLES                                      *)
(*     DETERMINE IF THE DEFINED TYPE EXISTS                          *)
(*     IF DEFINED TYPE EXISTS THEN PUT DEFINED TYPE TRANSACTION ONTO *)
(*       THE TRANSACTION STACK                                       *)
(*     BUILD POINTER REFERENCING EXISTING ENTITY OR CLASS OR BUILD   *)
(*       UNRESOLVED ENTITY.                                          *)
(*     WRITE APPROPRIATE ERROR MESSAGES                              *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA       *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE DEFDEF *)
```

```
(* %INCLUDE DEFENM *)
(**)
  PROCEDURE DEFENM(VAR IRC            :  RET_REC;
                   VAR ENT_KIND       :  INTEGER;
                   VAR TRANS_STACK    :  TRANSPTR;
                   VAR TOKEN          :  T_TOKEN;
                   VAR TOKEN_VALUE    :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION :  INTEGER;
                   VAR TOKEN_LENGTH   :  INTEGER;
                   VAR REPORT1        :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*          Batch Interface routine that processes an enumeration    *)
(*          definition.                                              *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*        IRC              O   INTERNAL RETURN CODE                  *)
(*        ENT_KIND         I   THE KIND OF ENTITY BEING CONSTRUCTED  *)
(*        TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*        TOKEN           I/O  TOKEN FROM BATCH INPUT                *)
(*        TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*        TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE       *)
(*        TOKEN_LENGTH    I/O  LENGTH OF TOKEN                       *)
(*        REPORT1         I/O  OUTPUT FILE                           *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*                                                                   *)
(*     INITIALIZE VARIABLES                                          *)
(*     PUSH ENUMERATION TRANSACTION ONTO THE STACK                   *)
(*     GET EACH ENUMERATION ITEM                                     *)
(*     VERIFY THE UNIQUENESS OF EACH ENUMERATION ITEM NAME AMONG THE *)
(*       ENTITIES CURRENTLY MODELED AND THOSE ON THE STACK           *)
(*     IF THE ENUMERATION ITEM NAME IS UNIQUE THEN PUSH A TRANSACTION *)
(*       ONTO THE STACK                                              *)
(*     WRITE APPROPRIATE ERROR MESSAGES                              *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA      *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE DEFENM *)
```

```
(* %INCLUDE DEFENT *)
(**)
  PROCEDURE DEFENT(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*         Batch Interface routine that processes an entity definition     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*       NAME            I/O  DESCRIPTION                                  *)
(*       ====            ===  ===========                                  *)
(*       IRC              O   INTERNAL RETURN CODE                         *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                            *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                       *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                 *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE              *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                              *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN     *)
(*                            THE SUBSCHEMA                                *)
(*       SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES         *)
(*       SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF          *)
(*                            SUBSCHEMA ENTITIES                           *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN     *)
(*                            THE CLASS                                    *)
(*       CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES             *)
(*       CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF          *)
(*                            CLASS ENTITIES                               *)
(*       REPORT1         I/O  OUTPUT FILE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
```

```
(*   $EXECUTION PROCEDURE:                                              *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*                                                                      *)
(*    INITIALIZE VARIABLES                                              *)
(*    GET THE ENTITY NAME AND KIND NUMBER                               *)
(*    VERIFY THE UNIQUENESS OF THE ENTITY NAME AND KIND NUMBER AMONG    *)
(*      MODELED ENTITIES AND THOSE ON THE STACK                         *)
(*    IF THE ENTITY NAME AND KIND NUMBER ARE UNIQUE THEN CHECK IF THE   *)
(*      SUBSCHEMA FLAG OR CLASS FLAG ARE SET.  IF SO, ADD THE ENTITY    *)
(*      NAME TO THE APPROPRIATE LIST(S).  PUSH ENTITY TRANSACTION ON-   *)
(*      TO THE STACK.                                                   *)
(*    DEFINE EACH ATTRIBUTE OF THE ENTITY                               *)
(*    WHEN THE END OF THE ENTITY IS ENCOUNTERED, THEN PUSH FINAL        *)
(*      ENTITY TRANSACTION ONTO THE TRANSACTION STACK AND PROCESS       *)
(*      THE STACK.                                                      *)
(*    WRITE ERROR MESSAGES AS APPROPRIATE                               *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA      *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFENT *)
```

```
(* %INCLUDE DEFGBL *)
(**)
  PROCEDURE DEFGBL(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that processes a global attribute *)
(*         definition.                                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME            I/O  DESCRIPTION                            *)
(*       ====            ===  ===========                            *)
(*       IRC              O   INTERNAL RETURN CODE                   *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                      *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                 *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT           *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE        *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                        *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE SUBSCHEMA                          *)
(*       SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES   *)
(*       SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF    *)
(*                            SUBSCHEMA ENTITIES                     *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE CLASS                              *)
(*       CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES       *)
(*       CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF    *)
(*                            CLASS ENTITIES                         *)
(*       REPORT1         I/O  OUTPUT FILE                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
```

```
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*                                                                        *)
(*    PERFORM INITIALIZATIONS                                             *)
(*    GET GLOBAL ATTRIBUTE NAME                                           *)
(*    PUSH INITIAL GLOBAL ATTRIBUTE TRANSACTION ONTO THE STACK            *)
(*    DEFINE THE GLOBAL ATTRIBUTE                                         *)
(*    IF GLOBAL ATTRIBUTE ACCEPTED, THEN PUSH FINAL GLOBAL ATTRIBUTE      *)
(*      TRANSACTION ONTO THE STACK AND PROCESS.  IF THE GLOBAL ATTRI-      *)
(*      BUTE NOT ACCEPTED THEN RECOVER.                                   *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                    *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    ORIGINATED: 05/15/87       C. H. MOHME                  DBMA       *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE DEFGBL *)
```

```
(* %INCLUDE DEFPRE *)
(**)
  PROCEDURE DEFPRE(VAR IRC             :  RET_REC;
                   VAR ENT_KIND        :  INTEGER;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR DATA            :  TRANSACTION;
                   VAR PRECISION       :  INTEGER;
                   VAR REPORT1         :  TEXT);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*        Batch Interface routine that processes an integer precision,     *)
(*        real precision, or string length.                                *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME           I/O  DESCRIPTION                                 *)
(*         ====           ===  ===========                                 *)
(*         IRC             O   INTERNAL RETURN CODE                        *)
(*         ENT_KIND        I   THE KIND OF ENTITY BEING CONSTRUCTED        *)
(*         TRANS_STACK    I/O  TRANSACTION STACK                           *)
(*         TOKEN          I/O  TOKEN FROM BATCH INPUT                      *)
(*         TOKEN_VALUE    I/O  TOKEN VALUE FROM BATCH INPUT                *)
(*         TOKEN_LOCATIONI/O   LOCATION OF TOKEN IN INPUT LINE             *)
(*         TOKEN_LENGTH   I/O  LENGTH OF TOKEN                             *)
(*         DATA           I/O  CONTAINS TRANSACTION INFORMATION            *)
(*         PRECISION       O   INTEGER PRECISION, REAL PRECISION, OR        *)
(*                             STRING LENGTH                               *)
(*         REPORT1        I/O  OUTPUT FILE                                 *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT         *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*                                                                         *)
(*     INITIALIZE VARIABLES                                                *)
(*     GET THE PRECISION, IF PRESENT                                       *)
```

```
(*    IF THE PRECISION IS SPECIFIED, THEN VERIFY THAT IT IS IN THE    *)
(*      PROPER RANGE.                                                 *)
(*    IF THE PRECISION IS NOT SPECIFIED, THEN ASSIGN THE DEFAULT      *)
(*      PRECISION.                                                    *)
(*     WRITE APPROPRIATE ERROR MESSAGES                               *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFPRE *)
```

```
(* %INCLUDE DEFPTR *)
(**)
  PROCEDURE DEFPTR(VAR IRC             :  RET_REC;
                   VAR ENT_KIND        :  INTEGER;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that processes a pointer definition*)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*       IRC              O   INTERNAL RETURN CODE                  *)
(*       ENT_KIND         I   THE KIND OF ENTITY BEING CONSTRUCTED  *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       TOKEN_LOCATION  I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       TOKEN_LENGTH    I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       REPORT1         I/O  OUTPUT FILE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    INITIALIZE VARIABLES                                           *)
(*    PUSH INITIAL POINTER TRANSACTION ONTO THE TRANSACTION STACK    *)
(*    GET EACH POINTER CONSTITUENT NAME OR KIND NUMBER.              *)
(*      DETERMINE IF THE ENTITY OR CLASS EXISTS                      *)
(*      VERIFY THAT THE ENTITY OR CLASS IS NOT ALREADY A CONSTITUENT. *)
(*        IF IT IS NOT, THEN PUSH ENTITY KEY TRANSACTION ONTO THE    *)
(*        TRANSACTION STACK.                                         *)
(*      IF THE POINTER REFERENCES AN ENTITY THAT IS NOT DEFINED, THEN *)
(*        CREATE AN UNRESOLVED ENTITY.  IF THE UNDEFINED ENTITY IS   *)
(*        LATER DEFINED, IT WILL REPLACE THE UNRESOLVED ENTITY IN THE *)
(*        POINTER'S CONSTITUENT LIST.                                *)
```

```
(*    PUSH THE FINAL POINTER TRANSACTION ONTO THE TRANSACTION STACK   *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFPTR *)
```

```
(* %INCLUDE DEFQUERY *)
(**)
  PROCEDURE DEFQUERY(VAR IRC            :  RET_REC;
                     VAR IDENTIFIER     :  T_NAME;
                     VAR KIND           :  INTEGER;
                     VAR ENT_KIND       :  INTEGER;
                     VAR DEFINITION_KEY :  ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that determines if a newly modeled *)
(*         entity satisfies any unresolved entity references on the  *)
(*         backpatch list.                                           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME            I/O  DESCRIPTION                            *)
(*       ====            ===  ===========                            *)
(*       IRC              O   INTERNAL RETURN CODE                   *)
(*       IDENTIFIER       I   NAME OF ENTITY, CLASS, OR DEFINED TYPE *)
(*       KIND             I   KIND NUMBER OF ENTITY OR CLASS         *)
(*       ENT_KIND         I   THE KIND OF ENTITY (DEFINED TYPE, ENTITY *)
(*                            OR CLASS)                              *)
(*       DEFINITION_KEY   I   KEY OF "ENTITY" WITH UNRESOLVED        *)
(*                            REFERENCE                              *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*     INITIALIZE VARIABLES                                          *)
(*     DETERMINE IF ANY BACKPATCH ENTITIES EXIST IN THE MODEL        *)
(*     IF BACKPATCH ENTITIES EXIST, DETERMINE IF ONE MATCHES THE NAME *)
(*       OR KIND OF THE NEWLY CREATED ENTITY.                        *)
(*     IF A MATCH HAS BEEN FOUND, THEN WE CAN RESOLVE EACH REFERENCE *)
(*       OF THE BACKPATCH ENTITY.                                    *)
(*     IF THE NEWLY CREATED ENTITY IS A DEFINED TYPE, THEN WE REPLACE *)
(*       THE CONSTITUENT OF THE REFERENCE KEY WITH THE KEY OF THE    *)
(*       NEWLY DEFINED ENTITY.                                       *)
(*     IF THE REFERENCE KEY IS A CONSTITUENT OF A CLASS, THEN WE PUT *)
(*       THE KEY OF THE NEWLY DEFINED ENTITY IN THE CLASS' CONSTITU- *)
(*       ENT LIST IN PLACE OF THE REFERENCE KEY.                     *)
```

```
(*      IF THE REFERENCE KEY IS A CONSTITUENT OF A SUBSCHEMA, THEN WE  *)
(*        PUT THE KEY OF THE NEWLY DEFINED ENTITY IN THE SUBSCHEMA'S   *)
(*        CONSTITUENT LIST IN PLACE OF THE REFERENCE KEY.              *)
(*      IF THE REFERENCE KEY IS A CONSTITUENT OF A POINTER, THEN WE    *)
(*        PUT THE KEY OF THE NEWLY DEFINED ENTITY IN THE POINTER'S     *)
(*        CONSTITUENT LIST IN PLACE OF THE REFERENCE KEY.              *)
(*      IF THE NEWLY DEFINED ENTITY IS NOT A DEFINED TYPE AND IF THE   *)
(*        REFERENCE KEY IS NOT A CONSTITUENT OF A SUBSCHEMA, CLASS OR  *)
(*        POINTER THEN WE BUILD A POINTER (WHICH REFERENCES THE        *)
(*        NEWLY DEFINED ENTITY) AND MAKE IT A CONSTITUENT OF THE       *)
(*        REFERENCE KEY.                                               *)
(*      DELETE THE BACKPATCH ENTITY AND READ THE NEXT ONE              *)
(*                                                                     *)
(*                                                                     *)
(*                                                                     
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED: 04/22/87       C. H. MOHME                 DBMA       *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE DEFQUERY *)
```

```
(* %INCLUDE DEFSTC *)
(**)
  PROCEDURE DEFSTC(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*         Batch Interface routine that processes a structure              *)
(*         definition.                                                     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME            I/O  DESCRIPTION                                 *)
(*        ====            ===  ===========                                 *)
(*        IRC              O   INTERNAL RETURN CODE                        *)
(*        TRANS_STACK     I/O  TRANSACTION STACK                           *)
(*        TOKEN           I/O  TOKEN FROM BATCH INPUT                      *)
(*        TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                *)
(*        TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE             *)
(*        TOKEN_LENGTH    I/O  LENGTH OF TOKEN                             *)
(*        SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                             THE SUBSCHEMA                               *)
(*        SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES        *)
(*        SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF         *)
(*                             SUBSCHEMA ENTITIES                          *)
(*        CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                             THE CLASS                                   *)
(*        CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES            *)
(*        CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF         *)
(*                             CLASS ENTITIES                              *)
(*        REPORT1         I/O  OUTPUT FILE                                 *)
(*                                                                         *)
```

```
(*   $COMMONS:                                                          *)
(*                                                                      *)
(*   $ENVIRONMENT:                                                      *)
(*        LANGUAGE: IBM PASCAL                                          *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*                                                                      *)
(*   $EXECUTION PROCEDURE:                                              *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*                                                                      *)
(*    PERFORM INITIALIZATIONS                                           *)
(*    PUSH INITIAL STRUCTURE TRANSACTION ONTO THE STACK                 *)
(*    DEFINE STRUCTURE ATTRIBUTES                                       *)
(*    RECOVER FROM ERROR, AS NECESSARY                                  *)
(*    PUSH FINAL STRUCTURE TRANSACTION ONTO THE STACK AND GET NEXT      *)
(*      TOKEN                                                           *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                  *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    ORIGINATED: 05/18/87       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFSTC *)
```

```
(* %INCLUDE DEFSUB *)
(**)
  PROCEDURE DEFSUB(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      Batch Interface routine that processes a subschema definition.     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*       NAME            I/O  DESCRIPTION                                  *)
(*       ====            ===  ===========                                  *)
(*       IRC              O   INTERNAL RETURN CODE                         *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                            *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                       *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                 *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE              *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                              *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN     *)
(*                            THE SUBSCHEMA                                *)
(*       SUB_ENT_HEAD    I/O  POINTS TO THE LIST OF SUBSCHEMA ENTITIES     *)
(*       SUB_ENT_LIST    I/O  POINTS TO THE CURRENT ENTITY IN THE LIST     *)
(*                            OF SUBSCHEMA ENTITIES                        *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN     *)
(*                            THE CLASS                                    *)
(*       CLS_ENT_HEAD    I/O  POINTS TO THE LIST OF CLASS ENTITIES         *)
(*       CLS_ENT_LIST    I/O  POINTS TO THE CURRENT ENTITY IN THE LIST     *)
(*                            OF CLASS ENTITIES                            *)
(*       REPORT1         I/O  OUTPUT FILE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
```

```
(*   $EXECUTION PROCEDURE:                                              *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*                                                                      *)
(*    INITIALIZE VARIABLES                                              *)
(*    GET SUBSCHEMA NAME AND VERIFY ITS UNIQUENESS                      *)
(*    IF SUBSCHEMA NAME IS UNIQUE THEN GET NEXT MEANINGFUL TOKEN        *)
(*    IF THE TOKEN IS AN IDENTIFIER OR AN INTEGER, THEN WE KNOW THAT    *)
(*      WE WILL BE GETTING A LIST OF ENTITIES AND CLASSES THAT ARE TO   *)
(*      BE MEMBERS OF THE SUBSCHEMA.  PUSH THE SUBSCHEMA NAME ONTO      *)
(*      THE PROCESSING STACK.                                           *)
(*      DETERMINE IF THE IDENTIFIER IS AN ENTITY OR A CLASS             *)
(*      IF WE HAVE AN ENTITY OR CLASS KEY, THEN WE VERIFY THAT IT IS    *)
(*        NOT ALREADY A MEMBER OF THE SUBSCHEMA.  IF IT IS NOT, THEN    *)
(*        WE ATTACH THE ENTITY KEY TO THE LIST AND PUSH A TRANS-        *)
(*        ACTION ONTO THE TRANSACTION STACK.                            *)
(*      IF THE SUBSCHEMA'S CONSTITUENT IS NOT YET DEFINED, CREATE AN    *)
(*        UNRESOLVED ENTITY.  WHEN THE CONSTITUENT IS LATER DEFINED,    *)
(*        IF WILL REPLACE THE UNRESOLVED ENTITY IN THE CONSTITUENT      *)
(*        LIST OF THE SUBSCHEMA.                                        *)
(*      IF WE HAVE READ IN ALL OF THE ENTITY AND CLASS NAMES, WE PUSH   *)
(*        THE FINAL SUBSCHEMA TRANSACTION ONTO THE STACK AND PROCESS    *)
(*        THE TRANSACTION STACK.                                        *)
(*    IF WE HAVE ENCOUNTERED AN ENTITY OR CLASS DEFINITION, THEN WE     *)
(*      SET A FLAG TO INDICATE THAT AFTER MODELING THESE ENTITIES       *)
(*      AND CLASSES, WE MUST THEN MODEL THE SUBSCHEMA.                  *)
(*    WRITE APPROPRIATE ERROR MESSAGES.                                 *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA      *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFSUB *)
```

```
(* %INCLUDE DEFSUP *)
(**)
  PROCEDURE DEFSUP(VAR IRC             :  RET_REC;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR TOKEN           :  T_TOKEN;
                   VAR TOKEN_VALUE     :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION  :  INTEGER;
                   VAR TOKEN_LENGTH    :  INTEGER;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR REPORT1         :  TEXT);
       SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      Batch Interface routine that processes a supertype definition.*)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*       NAME            I/O  DESCRIPTION                                 *)
(*       ====            ===  ===========                                 *)
(*       IRC              O   INTERNAL RETURN CODE                        *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                           *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                      *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT                *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE             *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                             *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                            THE SUBSCHEMA                               *)
(*       SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES        *)
(*       SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF         *)
(*                            SUBSCHEMA ENTITIES                          *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN    *)
(*                            THE CLASS                                   *)
(*       CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES            *)
(*       CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF         *)
(*                            CLASS ENTITIES                              *)
(*       REPORT1         I/O  OUTPUT FILE                                 *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT         *)
(*                                                                        *)
```

```
(*   $PROCESSING DESCRIPTION:                                              *)
(*                                                                         *)
(*    INITIALIZE VARIABLES                                                 *)
(*    GET THE ENTITY NAME AND KIND NUMBER                                  *)
(*    VERIFY THE UNIQUENESS OF THE ENTITY NAME AND KIND NUMBER AMONG       *)
(*      MODELED ENTITIES AND THOSE ON THE STACK                            *)
(*    IF THE ENTITY NAME AND KIND NUMBER ARE UNIQUE THEN CHECK IF THE      *)
(*      SUBSCHEMA FLAG OR CLASS FLAG ARE SET.  IF SO, ADD THE ENTITY       *)
(*      NAME TO THE APPROPRIATE LIST(S).  PUSH ENTITY TRANSACTION ON-      *)
(*      TO THE STACK.                                                      *)
(*    DEFINE EACH ATTRIBUTE OF THE ENTITY                                  *)
(*    WHEN THE END OF THE ENTITY IS ENCOUNTERED, THEN PUSH FINAL           *)
(*      ENTITY TRANSACTION ONTO THE TRANSACTION STACK AND PROCESS          *)
(*      THE STACK.                                                         *)
(*    WRITE ERROR MESSAGES AS APPROPRIATE                                  *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    ORIGINATED: 09/29/87       C. H. MOHME                 DBMA         *)
(*                                                                         *)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*END----------------------------------------------------------------------*)
(* END %INCLUDE DEFSUP *)
```

```
(* %INCLUDE DEFTYP *)
(**)
  PROCEDURE DEFTYP(VAR IRC            :  RET_REC;
                   VAR TRANS_STACK    :  TRANSPTR;
                   VAR TOKEN          :  T_TOKEN;
                   VAR TOKEN_VALUE    :  T_TOKEN_VALUE;
                   VAR TOKEN_LOCATION :  INTEGER;
                   VAR TOKEN_LENGTH   :  INTEGER;
                   VAR SUBSCHEMA_FLAG :  BOOLEAN;
                   VAR SUB_ENT_HEAD   :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST   :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG     :  BOOLEAN;
                   VAR CLS_ENT_HEAD   :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST   :  ENTITY_LIST_PTR;
                   VAR REPORT1        :  TEXT);
      SUBPROGRAM;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*         Batch Interface routine that processes a defined type      *)
(*         definition.                                                *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*       NAME            I/O  DESCRIPTION                           *)
(*       ====            ===  ===========                           *)
(*       IRC              O   INTERNAL RETURN CODE                  *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                     *)
(*       TOKEN           I/O  TOKEN FROM BATCH INPUT                *)
(*       TOKEN_VALUE     I/O  TOKEN VALUE FROM BATCH INPUT          *)
(*       TOKEN_LOCATION  I/O  LOCATION OF TOKEN IN INPUT LINE       *)
(*       TOKEN_LENGTH    I/O  LENGTH OF TOKEN                       *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE SUBSCHEMA                         *)
(*       SUB_ENT_HEAD    I/O  POINTS TO LIST OF SUBSCHEMA ENTITIES  *)
(*       SUB_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF   *)
(*                            SUBSCHEMA ENTITIES                    *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE CLASS                             *)
(*       CLS_ENT_HEAD    I/O  POINTS TO LIST OF CLASS ENTITIES      *)
(*       CLS_ENT_LIST    I/O  POINTS TO CURRENT ENTITY IN LIST OF   *)
(*                            CLASS ENTITIES                        *)
(*       REPORT1         I/O  OUTPUT FILE                           *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
```

```
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT     *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*                                                                    *)
(*    INITIALIZE VARIABLES                                            *)
(*    GET THE DEFINED TYPE NAME AND VERIFY ITS UNIQUENESS AMONG       *)
(*      ENTITIES IN THE MODEL AND ON THE STACK                        *)
(*    IF THE DEFINED TYPE NAME IS UNIQUE, THEN PUSH DEFINED TYPE      *)
(*      TRANSACTION ONTO THE TRANSACTION STACK.                       *)
(*    GET THE TYPE DEFINITION.  THE TYPE MAY BE AN ENUMERATION, A     *)
(*      BASIC TYPE, OR A DEFINED TYPE.                                *)
(*    IF THE TYPE DEFINITION IS ENUMERATION THEN                      *)
(*      DEFINE ENUMERATION                                            *)
(*    ELSE                                                            *)
(*      IF THE TYPE DEFINITION IS STRUCTURE THEN                      *)
(*        DEFINE STRUCTURE                                            *)
(*      ELSE                                                          *)
(*        IF THE TYPE DEFINITION IS A BASICA TYPE THEN                *)
(*          DEFINE BASIC TYPE                                         *)
(*        ELSE                                                        *)
(*          IF THE TYPE DEFINITION IS A DEFINED TYPE THEN             *)
(*            DEFINED DEFINED TYPE                                    *)
(*    PROCESS THE TRANSACTION STACK                                   *)
(*    WRITE APPROPRIATE ERROR MESSAGES                                *)
(*    GET NEXT DEFINED TYPE NAME IF PRESENT                           *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE DEFTYP *)
```

```
(* %INCLUDE DISPLIST *)
(**)
PROCEDURE DISPLIST(VAR MESS          : MESSAGE;
                   VAR MEMBERLIST    : T_ARRAY23;
                   VAR MAX_ARRAYSIZE : INTEGER;
                   VAR NAME          : T_NAME;
                   VAR NEXT_OP       : OPERATIONS;
                   VAR RR            : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DISPLAYS A LIST OF ENTITIES.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        MESS           I   THE ERROR MESSAGE RECEIVED FROM MAINLINE *)
(*        MEMBERLIST     I   THE ARRAY OF MEMBERS TO SELECT FROM    *)
(*        MAX_ARRAYSIZE  I   THE SIZE OF THE ARRAY OF MEMBERS       *)
(*        NAME           O   THE MEMBER SELECTED                    *)
(*        NEXT_OP        O   TELLS THE MAINLINE WHAT PANEL TO CALL  *)
(*                           NEXT                                    *)
(*        RR             O   TELLS THE MAINLINE IF THERE IS AN ERROR *)
(*                           AND IN WHAT ROUTINE IT OCCURS           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE DISPLAY LIST PANEL (DISPLIST) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.  THE MEMBER SELECTED IS RETURNED TO THE CALLING PRO- *)
(*        CEDURE.                                                    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE ENTCLS *)
(**)
  PROCEDURE ENTCLS(VAR IRC          :  RET_REC;
                   VAR DATA_REC     :  BLKDATA);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that determines if a specified    *)
(*         identifier is a modeled entity or class.                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   INTERNAL RETURN CODE                   *)
(*        DATA_REC       I/O  CONTAINS ENTITY DATA                   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    INITIALIZE VARIABLES                                           *)
(*    DETERMINE IF THE NAME IS AN ENTITY.  IF IT IS, GET THE KEY.    *)
(*    IF THE NAME IS NOT AN ENTITY, DETERMINE IF IT IS A CLASS.  IF  *)
(*      IT IS, GET THE KEY.                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE ENTCLS *)
```

```
(* %INCLUDE ERRMSG *)
(**)
  PROCEDURE ERRMSG(VAR ENT_KIND       :  INTEGER;
                   VAR REPORT1        :  TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*         Batch Interface routine that writes appropriate error     *)
(*         messages to the report file.                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME            I/O  DESCRIPTION                            *)
(*       ====            ===  ===========                            *)
(*       ENT_KIND         I   THE KIND OF ENTITY BEING CREATED       *)
(*       REPORT1         I/O  THE BATCH OUTPUT FILE                  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*     WRITE APPROPRIATE ERROR MESSAGES FOR THE SUBSCHEMA, CLASS,    *)
(*       ENTITY, SUPERTYPE, DEFINED TYPE, STRUCTURE, AND GLOBAL      *)
(*       ATTRIBUTE KINDS.                                            *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 12/12/87       C. H. MOHME                 DBMA    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE ERRMSG *)
```

```
(* %INCLUDE ERRREC *)
(**)
  PROCEDURE ERRREC(VAR IRC             :  RET_REC;
                   VAR ENT_KIND        :  INTEGER;
                   VAR TRANS_STACK     :  TRANSPTR;
                   VAR SUBSCHEMA_FLAG  :  BOOLEAN;
                   VAR SUB_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR SUB_ENT_LIST    :  ENTITY_LIST_PTR;
                   VAR CLASS_FLAG      :  BOOLEAN;
                   VAR CLS_ENT_HEAD    :  ENTITY_LIST_PTR;
                   VAR CLS_ENT_LIST    :  ENTITY_LIST_PTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        Batch Interface routine that performs the necessary actions *)
(*        to recover from an input error.                            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*       NAME            I/O  DESCRIPTION                            *)
(*       ====            ===  ===========                            *)
(*       IRC             I/O  INTERNAL RETURN CODE                   *)
(*       ENT_KIND         I   KIND OF ENTITY                         *)
(*       TRANS_STACK     I/O  TRANSACTION STACK                      *)
(*       SUBSCHEMA_FLAG  I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE SUBSCHEMA                          *)
(*       SUB_ENT_HEAD    I/O  POINTS TO THE LIST OF SUBSCHEMA ENTITIES *)
(*       SUB_ENT_LIST    I/O  POINTS TO THE CURRENT ENTITY IN THE LIST *)
(*                            OF SUBSCHEMA ENTITIES                  *)
(*       CLASS_FLAG      I/O  INDICATES IF ENTITIES ARE DEFINED WITHIN *)
(*                            THE CLASS                              *)
(*       CLS_ENT_HEAD    I/O  POINTS TO THE LIST OF CLASS ENTITIES   *)
(*       CLS_ENT_LIST    I/O  POINTS TO THE CURRENT ENTITY IN THE LIST *)
(*                            OF SUBSCHEMA ENTITIES                  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*                                                                     *)
(*    INITIALIZE VARIABLES                                             *)
(*    CLEAR TRANSACTION STACK                                          *)
(*    CASE TYPE OF RECOVERY OF                                         *)
(*      SUBSCHEMA :  DISCARD TOKENS UNTIL END_SCHEMA ENCOUNTERED       *)
(*      CLASS     :  DISCARD TOKENS UNTIL END_CLASS ENCOUNTERED        *)
(*                   IF CLASS IS IN ANOTHER CLASS, DISCARD TOKENS     *)
(*                     UNTIL END_CLASS IS ENCOUNTERED                 *)
(*                   IF CLASS IS IN A SUBSCHEMA, DISCARD TOKENS UNTIL *)
(*                     END_SCHEMA ENCOUNTERED                         *)
(*      ENTITY    :  DISCARD TOKENS UNTIL END_ENTITY ENCOUNTERED       *)
(*                   IF ENTITY IS IN A CLASS, DISCARD TOKENS UNTIL     *)
(*                     END_CLASS ENCOUNTERED                          *)
(*                   IF ENTITY IS IN A SUBSCHEMA, DISCARD TOKENS      *)
(*                     UNTIL END_SCHEMA ENCOUNTERED                   *)
(*      DEFINED                                                        *)
(*      TYPE      :  DISCARD TOKENS UNTIL SEMICOLON ENCOUNTERED        *)
(*                   IF DEFINED TYPE IS IN SUBSCHEMA DISCARD TOKENS    *)
(*                     UNTIL END_SCHEMA ENCOUNTERED                   *)
(*    SET FLAGS TO FALSE                                               *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
(*    ORIGINATED: 03/20/87       C. H. MOHME                 DBMA     *)
(*                                                                     *)
(*-------------------------------------------------------------------*)
(*                                                                     *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE ERRREC *)
```

```
(* BEGIN %INCLUDE GETDD ************************************************)
(*                                                                      *)
PROCEDURE GETDD ( CONST  KIND             : INTEGER;
                  CONST  MAX_AVAIL        : INTEGER;
                  CONST  ATTRIBUTE_ORDER  : CHAR;
                  VAR    USER_ARRAY       : T_USER_ARRAY;
                  VAR    MAX_ACTUAL       : INTEGER;
                  VAR    RETURN_CODE      : INTEGER );
        EXTERNAL;
(*                                                                      *)
(*   $FUNCTION:                                                         *)
(*        READ THE DATA DICTIONARY INTO THE APPLICATION PROGRAM.        *)
(*                                                                      *)
(*   $DESCRIPTION OF ARGUMENTS:                                         *)
(*        NAME                 I/O  DESCRIPTION                       *)
(*        ====                 ===  ===========                       *)
(*        KIND                  I   A KIND NUMBER OF ENTITY           *)
(*        MAX_ACTUAL            O   AN ACTUAL NUMBER OF RECORDS IN    *)
(*                                  ENTITY DEFINITION                 *)
(*        MAX_AVAIL             I   A NUMBER OF 80 CHARACTER RECORDS  *)
(*                                  AVAILABLE IN CALLER TO HOLE       *)
(*                                  ENTITY DEFINITION                 *)
(*        USER_ARRAY            O   AN ENTITY DEFINITION              *)
(*        RETURN_CODE           O   RETURN CODE                       *)
(*                                 -1 = ACTUAL SIZE GREATER THAN      *)
(*                                      SPACE AVAILABLE               *)
(*                                  0 = SUCCESS                       *)
(*                                  1 = KIND NOT IN DATA DICTIONARY   *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH DATA DICTIONARY INDEX FILE                     *)
(*          IF KIND IN DATA DICTIONARY THEN                           *)
(*             GET ENTITY DEFINITION FROM DDFILE                      *)
(*             FILL UP THE ARRAY OF ENTITY DEFINITIONS UP TO NUMBER   *)
(*             OF RECORDS AVAILABLE IN CALLER                         *)
(*          END IF                                                    *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 23 MARCH 1987, M. H. CHOI, DBMA                 *)
(*                                                                    *)
(* END %INCLUDE GETDD  ************************************************)
```

```
(* BEGIN %INCLUDE LEXICAL ********************************************)
(*                                                                   *)
PROCEDURE LEXICAL ( VAR   TOKEN           : T_TOKEN;
                    VAR   TOKEN_VALUE     : T_TOKEN_VALUE;
                    VAR   TOKEN_LOCATION  : INTEGER;
                    VAR   TOKEN_LENGTH    : INTEGER;
                    VAR   REPORT1         : TEXT);
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        LOCATE THE LONGEST POSSIBLE LEXEME FROM WHICH A TOKEN MAY  *)
(*        BE DETERMINED.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        TOKEN         I/O  INPUT  = TOKEN TYPE FROM PREVIOUS CALL  *)
(*                                    OR INITIALIZATION FLAG         *)
(*                           OUTPUT = CURRENT TOKEN TYPE             *)
(*        TOKEN_VALUE    O   CURRENT TOKEN VALUE                     *)
(*        TOKEN_LOCATION O   START LOCATION OF TOKEN IN REPORT LINE  *)
(*        TOKEN_LENGTH   O   LENGTH OF TOKEN IN REPORT LINE          *)
(*        REPORT1       I/O                                          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL SEGMENT SUBPROGRAM                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             SOURCE  : THE INPUT STREAM OF CHARACTERS             *)
(*             REPORT1 : FORMATTED REPORT OF THE INPUT              *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        ?                                                          *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*        ORIGINATED: 18 MAR 87, G. A. WHITE                         *)
(*                                                                   *)
(*        REVISED:     7 DEC 87, G. A. WHITE, ADD PARAMETERS FOR     *)
(*                     LOCATION AND LENGTH OF TOKEN IN REPORT LINE.  *)
(*                                                                   *)
(* END %INCLUDE LEXICAL **********************************************)
```

```
(* %INCLUDE LMEM23 *)
(**)
PROCEDURE LMEM23(VAR MESS     : MESSAGE;
                 VAR MEMBERS  : T_ARRAY23;
                 VAR SIZE     : INTEGER;
                 VAR NEXT_OP  : OPERATIONS;
                 VAR RR       : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE LIST MEMBERS (LMEM23) PANEL       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO DISPLAY          *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE LIST MEMBERS PANEL (LMEM23) BY MAKING ISPLNK   *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE MCREATE *)
(**)
  PROCEDURE MCREATE(VAR MESS    : MESSAGE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS PROCEDURE:                                              *)
(*                      DISPLAYS THE CREATE MENU                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE CREATE PANEL (MCREATE) BY MAKING ISPLNK CALLS. *)
(*        THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID  *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: 09/28/87          C. H. MOHME               DBMA      *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                         *)
(*                                                                   *)
```

```
(*    ORIGINATED: 08/13/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MCREATE *)
```

```
(* %INCLUDE MFILMOD *)
(**)
  PROCEDURE MFILMOD(VAR MESS    : MESSAGE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR      : RET_REC);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS PROCEDURE:                                            *)
(*                          DISPLAYS THE FILE/RETRIEVE MENU          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE FILE MODEL PANEL (MFILMOD) BY MAKING ISPLNK    *)
(*        CALLS. THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*        TYPE.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE MINCLUD *)
(**)
PROCEDURE MINCLUD(VAR MESS        : MESSAGE;
                  VAR REPORT_TYPE : OPERATIONS;
                  VAR MEMBERS     : T_ARRAY23;
                  VAR SIZE        : INTEGER;
                  VAR MEMBER      : T_NAME;
                  VAR NEXT_OP     : OPERATIONS;
                  VAR RR          : RET_REC);
SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      THIS FUNCTION:                                                    *)
(*                    DISPLAYS THE LIST OF SUBSCHEMAS                     *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        REPORT_TYPE    I   THE TYPE OF REPORT TO BE GENERATED           *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM          *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS             *)
(*        MEMBER         O   THE MEMBER SELECTED                          *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                    *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE SUBSCHEMA PANEL (MINCLUD) (FROM WHICH A SUB-        *)
(*        SCHEMA MAY BE SELECTED) BY MAKING ISPLNK CALLS.  THE OPTION     *)
(*        CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.                   *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE MMAIN *)
(**)
  PROCEDURE MMAIN(VAR MESS    : MESSAGE;
                  VAR NEXT_OP : OPERATIONS;
                  VAR RR      : RET_REC);

      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS PROCEDURE:                                            *)
(*                          DISPLAYS THE MAIN MENU                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                            OPERATION                              *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE MAIN MENU PANEL (MMAIN) BY MAKING ISPLNK CALLS. *)
(*        THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE MNEWMOD *)
(**)
  PROCEDURE MNEWMOD(VAR MESS     : MESSAGE;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);

      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS PROCEDURE:                                                 *)
(*                          DISPLAYS THE FILE/RETRIEVE MENU               *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL   *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT        *)
(*                            OPERATION                                   *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,    *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN     *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE NEW MODEL PANEL (MNEWMOD) BY MAKING ISPLNK          *)
(*        CALLS. THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.                                                           *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* BEGIN %INCLUDE MQBHALL ***********************************************)
(*                                                                       *)
PROCEDURE MQBHALL ( CONST NO_OF_KINDS  : INTEGER;
                    VAR   NO_OF_ENTRY  : INTEGER );
          SUBPROGRAM;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        PRINT ALL ENTITIES IN THE MODEL                                *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME               I/O  DESCRIPTION                            *)
(*        ====               ===  ===========                            *)
(*        NO_OF_KINDS         I   NUMBER OF KINDS IN THE MODEL           *)
(*        NO_OF_ENTRY        I/O  NUMBER OF ENTITIES PRINTED             *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM       *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        LOOP THROUGH LIST OF ENTITIES IN THE MODEL                     *)
(*           PRINT ENTITY                                                *)
(*        END LOOP                                                       *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)               *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                     *)
(*                                                                       *)
(* END %INCLUDE MQBHALL *************************************************)
```

```
(* BEGIN %INCLUDE MQBHATT  ********************************************)
(*                                                                    *)
PROCEDURE MQBHATT ( CONST KIND         : INTEGER;
                    CONST INST_NO      : INTEGER;
                    VAR   NO_OF_ENTRY  : INTEGER );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND               *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        KIND                I   A KIND NUMBER OF ENTITY             *)
(*        INST_NO             I   INSTANCE NUMBER                     *)
(*        NO_OF_ENTRY        I/O  NUMBER OF ENTITIES PRINTED          *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        PRINT ADB, CONSTITUENT LIST, AND USER LIST.                 *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                  *)
(*                                                                    *)
(* END %INCLUDE MQBHATT ***********************************************)
```

```
(* BEGIN %INCLUDE MQBHATTS ********************************************)
(*                                                                    *)
PROCEDURE MQBHATTS ( CONST KIND         : INTEGER;
                     VAR   NO_OF_ENTRY  : INTEGER );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PRINT ALL ENTITIES OF A SPECIFIC KIND                       *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        KIND               I   A KIND NUMBER OF ENTITY              *)
(*        NO_OF_ENTRY       I/O  NUMBER OF ENTITIES PRINTED           *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH LIST OF ENTITIES OF A SPECIFIC KIND            *)
(*           PRINT ADB, CONSTITUENT LIST, AND USER LIST               *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                  *)
(*                                                                    *)
(* END %INCLUDE MQBHATTS **********************************************)
```

```
(* BEGIN %INCLUDE MQBHENT  ********************************************)
(*                                                                    *)
PROCEDURE MQBHENT ( CONST MEMBER       : T_CHAR56;
                    VAR   NO_OF_ENTRY  : INTEGER;
                    VAR   ACTION       : OPERATIONS );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        DISPLAY BATCH ENTITY MENU ( SELECT TO PRINT ALL ENTITIES OF *)
(*           A SPECIFIC KIND OR TO PRINT INDIVIDUAL INSTANCES OF A    *)
(*           SPECIFIC KIND )                                          *)
(*                                                                    *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        MEMBER             I   ENTITY OF A SPCIFIC KIND             *)
(*        NO_OF_ENTRY       I/O  NUMBER OF ENTITIES PRINTED           *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT  *)
(*                               OPERATION                            *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        NAME/VALUE INTERFACE                                        *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        DISPLAY BATCH ENTITY MENU                                   *)
(*        SELECT TO PRINT ALL ENTITIES OF A SPECIFIC KIND OR          *)
(*        TO PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND            *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                  *)
(*                                                                    *)
(* END %INCLUDE MQBHENT ***********************************************)
```

```
(* BEGIN %INCLUDE MQBHMAIN ********************************************)
(*                                                                    *)
PROCEDURE MQBHMAIN ( CONST MEMBERLIST   : T_MEMBER;
                     CONST NO_OF_KINDS  : INTEGER;
                     VAR   ACTION       : OPERATIONS );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        DISPLAY BATCH MAIN MENU ( SELECT TO PRINT ALL ENTITIES IN   *)
(*           THE MODEL OR TO PRINT ALL ENTITIES OF A SPECIFIC KIND )  *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        MEMBERLIST         I   LIST OF ENTITIES IN THE MODEL        *)
(*        NO_OF_KINDS        I   NUMBER OF KINDS IN THE MODEL         *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT  *)
(*                               OPERATION                            *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        NAME/VALUE INTERFACE                                        *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        DISPLAY BATCH MAIN MENU                                     *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                  *)
(*                                                                    *)
(* END %INCLUDE MQBHMAIN **********************************************)
```

```
(* BEGIN %INCLUDE MQCLMU  ********************************************)
(*                                                                   *)
PROCEDURE MQCLMU ( CONST LIST_OF_CNSTS : LISTKEY;
                   CONST CL_NAME       : T_CL_NAME;
                   CONST NO_OF_CL      : INTEGER;
                   VAR   MEMBER        : T_CHAR56;
                   VAR   ACTION        : OPERATIONS );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DISPLAY A LIST OF CONSTITUENTS FOR A SPECIFIC KIND         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        LIST_OF_CNSTS      I   POINTER TO CONSTITUENT LIST         *)
(*        CL_NAME            I   POINTER ATTRIBUTE NAME              *)
(*        NO_OF_CL           I   NUMBER OF CONSTITUENT LIST          *)
(*        MEMBER             O   SELECTED CONSTITUENT FOR DEFINITIONS *)
(*        ACTION             O   ENUMERATED TYPE TO INDICATE THE NEXT *)
(*                               OPERATION                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                 *)
(*                                                                   *)
(* END %INCLUDE MQCLMU ***********************************************)
```

```
(* BEGIN %INCLUDE MQGETVAL ********************************************)
(*                                                                    *)
PROCEDURE MQGETVAL ( CONST DATA_TYPE       : INTEGER;
                     CONST SIZE            : INTEGER;
                     CONST ATTRIBUTE_VALUE : T_ATTRIBUTE_VALUE;
                     VAR   VAL             : STRING(16) );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        CONVERT ATTRIBUTE VALUE TO A STRING VALUE                   *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                        *)
(*        ====              ===  ===========                        *)
(*        DATA_TYPE          I   TYPE OF THE VALUE                  *)
(*        SIZE               I   SIZE OF THE VALUE                  *)
(*        ATTRIBUTE_VALUE    I   DEPENDS ON THE TYPE                *)
(*        VAL                O   STRING VALUE                       *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        NAME/VALUE INTERFACE                                        *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        ?                                                           *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED:  15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                    *)
(* END %INCLUDE MQGETVAL **********************************************)
```

```
(* BEGIN %INCLUDE MQGTDEFN ************************************************)
(*                                                                         *)
PROCEDURE MQGTDEFN ( CONST KIND              : INTEGER;
                     CONST ENTITY_KEY        : ENTKEY;
                     VAR   MEMBERLIST        : T_MEMBERLIST;
                     VAR   TOTAL_MEMBER      : INTEGER;
                     VAR   CL_NAME           : T_CL_NAME;
                     VAR   NO_OF_CL          : INTEGER );
        SUBPROGRAM;
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        GET ENTITY DEFINITIONS OF A SPECIFIC KIND                        *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME              I/O  DESCRIPTION                             *)
(*        ====              ===  ===========                             *)
(*        KIND               I   A KIND NUMBER OF ENTITY                 *)
(*        ENTITY_KEY         I   POINTER TO ENTITY INSTANCE              *)
(*        MEMBERLIST         O   LIST OF ATTRIBUTE NAMES WITH A VALUE    *)
(*        TOTAL_MEMBER       O   TOTAL NUMBER OF MEMBERS IN THE LIST     *)
(*        CL_NAME            O   LIST OF ATTRIBUTE NAMES IN THE          *)
(*                               CONSTITUENT LIST                        *)
(*        NO_OF_CL           O   TOTAL NUMBER OF CONSTITUENT LIST        *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM         *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        OBTAIN ENTITY DEFINITIONS FROM THE DATA DICTIONARY               *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)                 *)
(*        ORIGINATED:  10 DECEMBER, M. H. CHOI, DBMA                       *)
(*                                                                         *)
(* END %INCLUDE MQGTDEFN **************************************************)
```

```
(* BEGIN %INCLUDE MQIAATT ********************************************)
(*                                                                   *)
PROCEDURE MQIAATT ( CONST KIND        : T_CHAR48;
                    CONST INST_NO     : INTEGER;
                    VAR   TEMP_MEMBER : T_CHAR56;
                    VAR   ACTION      : OPERATIONS );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        PRINT INDIVIDUAL INSTANCES OF A SPECIFIC KIND              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        KIND               I   A KIND NUMBER OF ENTITY             *)
(*        INST_NO            I   INSTANCE NUMBER                     *)
(*        TEMP_MEMBER        O   SELECTED ENTITY                     *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT *)
(*                               OPERATION                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED:  15 DECEMBER 1987, M. H. CHOI, DBMA            *)
(*                                                                   *)
(* END %INCLUDE MQIAATT ***********************************************)
```

```
(* BEGIN %INCLUDE MQIAENT ********************************************)
(*                                                                   *)
PROCEDURE MQIAENT ( CONST MEMBER   : T_CHAR56;
                    VAR   ACTION   : OPERATIONS );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DISPLAY INTERACTIVE ENTITY MENU                            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME               I/O  DESCRIPTION                        *)
(*        ====               ===  ===========                        *)
(*        MEMBER              I   ENTITY OF A SPECIFIC KIND          *)
(*        ACTION              O   ENUMERATED TYPE INDICATING THE NEXT*)
(*                                OPERATION                          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED: 15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE MQIAENT **********************************************)
```

```
(* BEGIN %INCLUDE MQIAMAIN ******************************************)
(*                                                                   *)
PROCEDURE MQIAMAIN ( CONST MEMBERLIST  : T_MEMBER;
                     CONST NO_OF_KINDS : INTEGER;
                     VAR   ACTION      : OPERATIONS );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DISPLAY MAIN INTERACTIVE MENU                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        MEMBERLIST         I   LIST OF ENTITIES IN THE MODEL       *)
(*        NO_OF_KINDS        I   NUMBER OF KINDS IN THE MODEL        *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT *)
(*                               OPERATION                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY INTERACTIVE MAIN MENU                              *)
(*        SELECT AN ENTITY FOR THE DEFINITIONS                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED: 15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE MQIAMAIN ********************************************)
```

```
(* BEGIN %INCLUDE MQNCLMU ********************************************)
(*                                                                   *)
PROCEDURE MQNCLMU ( VAR ACTION   : OPERATIONS );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DISPLAY A MENU INDICATING NO CONSTITUENTS FOR A SPECIFIC   *)
(*        ENTITY.                                                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT *)
(*                               OPERATION                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED: 15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE MQNCLMU **********************************************)
```

```
(* BEGIN %INCLUDE MQNUSRMU  ******************************************)
(*                                                                    *)
PROCEDURE MQNUSRMU ( VAR ACTION  : OPERATIONS );
          SUBPROGRAM;
(*                                                                    *)
(*    $FUNCTION:                                                      *)
(*         DISPLAY A MENU INDICATING NO USERS FOR A SPECIFIC ENTITY    *)
(*                                                                    *)
(*    $DESCRIPTION OF ARGUMENTS:                                      *)
(*         NAME               I/O  DESCRIPTION                        *)
(*         ====               ===  ===========                        *)
(*         ACTION              O   ENUMERATED TYPE INDICATING THE NEXT *)
(*                                 OPERATION                          *)
(*                                                                    *)
(*    $COMMONS:                                                       *)
(*                                                                    *)
(*    $ENVIRONMENT:                                                   *)
(*         LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                    *)
(*    $EXECUTION PROCEDURE:                                           *)
(*         NAME/VALUE INTERFACE                                       *)
(*         CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*    $PROCESSING DESCRIPTION:                                        *)
(*         ?                                                          *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*         REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*         ORIGINATED: 15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                    *)
(* END %INCLUDE MQNUSRMU **********************************************)
```

```
(* %INCLUDE MQUDVR *)
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        THIS IS THE MAINLINE PROGRAM WHICH DRIVES THE MODEL QUERY   *)
(*        UTILITY OF THE SCHEMA MANAGER.                              *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*        DDNAMES USED WITH STANDARD FILES:                           *)
(*             DDFILE   = THE DATA DICTIONARY FILE DEFINITIONS.       *)
(*                                                                    *)
(*             DDINX    = THE DATA DICTIONARY FILE INDEX.             *)
(*                                                                    *)
(*             FT08F001 = PART MODEL MASTER FILE (PID).               *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*                                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                             *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED:                M. H. CHOI                 DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE MQUDVR *)
```

```
(* BEGIN %INCLUDE MQUSRMU ********************************************)
(*                                                                   *)
PROCEDURE MQUSRMU ( CONST LIST_OF_USERS : LISTKEY;
                    VAR   TEMP_MEMBER   : T_CHAR56;
                    VAR   ACTION        : OPERATIONS );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DISPLAY LIST OF USERS FOR A SPECIFIC ENTITY                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        LIST_OF_USERS      I   LIST OF USERS FOR A SPECIFIC ENTITY  *)
(*        TEMP_MEMBER        O   SELECTED USER FOR THE DEFINITIONS    *)
(*        ACTION             O   ENUMERATED TYPE INDICATING THE NEXT  *)
(*                               OPERATION                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        ?                                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)           *)
(*        ORIGINATED: 15 DECEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE MQUSRMU ***********************************************)
```

```
(* %INCLUDE MREPORT *)
(**)
  PROCEDURE MREPORT(VAR MESS    : MESSAGE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS PROCEDURE:                                                    *)
(*                        DISPLAYS THE REPORT MENU                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                            OPERATION                                    *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE REPORT MENU (MREPORT) BY MAKING ISPLNK CALLS.        *)
(*        THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.         *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE MREVIEW *)
(**)
  PROCEDURE MREVIEW(VAR MESS    : MESSAGE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS PROCEDURE:                                              *)
(*                        DISPLAYS THE REVIEW MENU                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                            OPERATION                              *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW PANEL (MREVIEW) BY MAKING ISPLNK CALLS.   *)
(*        THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR      I. M. THECHANGER          GROUP_ID   *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: 09/28/87           C. H. MOHME               DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                         *)
(*                                                                   *)
```

```
(*    ORIGINATED: 08/14/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE MREVIEW *)
```

```
(* %INCLUDE MUPDATE *)
(**)
  PROCEDURE MUPDATE(VAR MESS     : MESSAGE;
                    VAR NEXT_OP : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*       THIS PROCEDURE:                                             *)
(*                        DISPLAYS THE UPDATE MENU                   *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                           *)
(*         ====           ===  ===========                           *)
(*         MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*         NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                             OPERATION                             *)
(*         RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                             IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*         DDNAMES USED WITH STANDARD FILES:                         *)
(*              NONE                                                 *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                   *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         DISPLAY THE UPDATE PANEL (MUPDATE) BY MAKING ISPLNK CALLS.   *)
(*         THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED TYPE.  *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR      I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: 09/28/87           C. H. MOHME               DBMA    *)
(*      INCORPORATED THE SUPERTYPE DATA TYPE.                        *)
(*                                                                   *)
```

```
(*    ORIGINATED: 08/13/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE MUPDATE *)
```

```
(* BEGIN %INCLUDE NVRTVRS ********************************************)
(*                                                                   *)
PROCEDURE NVRTVRS ( CONST KIND             : INTEGER;
                    VAR   RUNTIME          : T_RUN_TIME;
                    VAR   RUNTIME_SIZE     : INTEGER;
                    VAR   RETURN_CODE      : INTEGER );
          SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        RETRIEVE ENTITY DEFINITIONS FROM THE FILE                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        KIND               I   THE KIND NUMBER OF THE ENTITY       *)
(*                               DEFINITION TO BE READ               *)
(*        RUNTIME            O   RUN-TIME SUBSCHEMA WHICH CONTAINS   *)
(*                               THE ENTITY DEFINITION, ALONG WITH   *)
(*                               ANY ENUMERATION VALUES, ANY ARRAY   *)
(*                               INFORMATIONS, AND CONSTITUENT LIST  *)
(*                               INFORMATIONS, IN A COMPACTED FORM.  *)
(*        RUNTIME_SIZE       O   THE NUMBER OF BYTES ACTUALLY REQUIRED *)
(*                               FOR THE COMPACTED RUN-TIME SUBSCHEMA. *)
(*        RETURN_CODE        O   RETURN CODE                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        NAME/VALUE INTERFACE                                       *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        LOOP THROUGH INXFILE                                       *)
(*           IF KIND FOUND IN INXFILE THEN                           *)
(*              LOOP THROUGH DATAFILE                                *)
(*                 IF KIND FOUND IN DATAFILE THEN                    *)
(*                    STORE ENTITY DEFINITION IN TEMPORARY WORK AREA *)
(*                 END IF                                            *)
(*           END IF                                                  *)
(*        END LOOP                                                   *)
(*        STORE ENTITY DEFINITION INTO RUN-TIME SUBSCHEMA            *)
(*        STORE SIZE OF ENTITY DEFINITION INTO RUN-TIME SUBSCHEMA    *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*        ORIGINATED: 21 OCTOBER 1986, M. H. CHOI, DBMA            *)
(*                                                                 *)
(* END %INCLUDE NVRTVRS ********************************************)
```

```
(* BEGIN %INCLUDE PHALFLD ********************************************)
(*                                                                   *)
PROCEDURE PHALFLD ( VAR   FIELD_KEY         : ENTKEY;
                    CONST OFFSET_LIST       : T_OFFSET_LIST;
                    CONST OFFSET_LIST_COUNT : INTEGER;
                    VAR   ARRAY_TABLE_POSITION : INTEGER;
                    VAR   CL_POSITION       : INTEGER;
                    VAR   ENUM_TABLE_POSITION : INTEGER;
                    VAR   IRC               : RET_REC );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        PHYSICALIZE THE ATTRIBUTE OF AN ENTITY                     *)
(*        ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME               I/O  DESCRIPTION                        *)
(*        ====               ===  ===========                        *)
(*        FIELD_KEY           I   FIELD KEY TO BE PHYSICALIZE        *)
(*        OFFSET_LIST         I   LIST OF ATTRIBUTES OF AN ENTITY    *)
(*                                ACCORDING TO BOUNDARY ALIGNMENT    *)
(*        OFFSET_LIST_COUNT   I   NUMBER OF ATTRIBUTES IN THE LIST   *)
(*        ARRAY_TABLE_POSIT   O   NUMBER OF ARRAYS                   *)
(*        CL_POSITION         O   NUMBER OF POINTERS                 *)
(*        ENUM_TABLE_POSIT    O   NUMBER OF ENUMERATIONS             *)
(*        IRC                 O   RETURN_CODE                        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        LOOP THROUGH ATTRIBUTES IN THE LIST OF BOUNDARY ALIGNMENT  *)
(*           IF ATTRIBUTE NAME IS IN THE LIST THEN                   *)
(*              OBTAIN SIZE AND OFFSET FROM THE LIST                 *)
(*              PHYSICALIZE THE ATTRIBUTE                            *)
(*           END IF                                                  *)
(*        END LOOP                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 26 NOVEMBER 1986, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE PHALFLD **********************************************)
```

```
(* BEGIN %INCLUDE PHBYFPOS ********************************************)
(*                                                                    *)
PROCEDURE PHBYFPOS ( VAR   ORDER_INDEX    : INTEGER;
                     VAR   ORDER_REC      : T_GLOBAL_FIELD;
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_POSITION    : INTEGER;
                     VAR   OFFSET_LIST_COUNT : INTEGER;
                     VAR   OFFSET_LIST    : T_OFFSET_LIST;
                     VAR   IRC            : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PHYSICALIZE THE ATTRIBUTES THAT SPECIFIED THE FIELD         *)
(*        POSITION ORDER ( DETERMINE ATTRIBUTE SIZE AND LOCATION )    *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        ORDER_INDEX        I   NUMBER OF ATTRIBUTES THAT SPECIFIED  *)
(*                               THE ORDER                            *)
(*        ORDER_REC          I   LIST OF ATTRIBUTES THAT SPECIFIED    *)
(*                               THE ORDER                            *)
(*        ARRAY_TABLE_POS    O   NUMBER OF ARRAYS                     *)
(*        ENUM_TABLE_POSR    O   NUMBER OF ENUMERATIONS               *)
(*        CL_POSITION        O   NUMBER OF POINTERS                   *)
(*        OFFSET_LIST_COUNT  O   NUMBER OF ATTRIBUTES IN THE LIST     *)
(*        OFFSET_LIST        O   LIST OF ATTRIBUTES OF AN ENTITY      *)
(*        IRC                O   RETURN_CODE                          *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        SORT ATTRIBUTES BY THE FIELD POSITION NUMBER                *)
(*        LOOP THROUGH THE ORDER LIST                                 *)
(*           PHYSICALIZE THE ATTRIBUTE                                *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 14 OCTOBER 1986, M. H. CHOI, DBMA               *)
(*                                                                    *)
(* END %INCLUDE PHBYFPOS **********************************************)
```

```
(* BEGIN %INCLUDE PHDECBYT ******************************************)
(*                                                                  *)
PROCEDURE PHDECBYT ( CONST CNST_ADB          : ENTBLOCK;
                     CONST NAME              : T_NAME;
                     CONST ARRAY_LENGTH      : INTEGER;
                     VAR   DW_ROOT           : T_DW_POINTER;
                     VAR   FW_ROOT           : T_FW_POINTER;
                     VAR   HW_ROOT           : T_HW_POINTER;
                     VAR   BY_ROOT           : T_BY_POINTER;
                     VAR   PNTR_ROOT         : T_PNTR_POINTER;
                     VAR   IRC               : RET_REC );
        SUBPROGRAM;
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        TRANSLATE DECIMAL DIGIT PRECISION INTO BYTE PRECISION AND *)
(*        BUILD A LIST OF ATTRIBUTES OF AN ENTITY ACCORDING TO      *)
(*        BOUNDARY ALIGNMENT.                                       *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME              I/O  DESCRIPTION                        *)
(*        ====              ===  ===========                        *)
(*        CNST_ADB           I   ADB OF CONSTITUENT                 *)
(*        NAME               I   ATTRIBUTE NAME                     *)
(*        ARRAY_LENGTH       I   NUMBER OF ARRAY DIMENSIONS         *)
(*        DW_ROOT            O   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               DOUBLE WORD ALIGNMENT              *)
(*        FW_ROOT            O   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               FULL WORD ALIGNMENT                *)
(*        HW_ROOT            O   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               HALF WORD ALIGNMENT                *)
(*        BY_ROOT            O   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               BYTES                              *)
(*        PNTR_ROOT          O   LIST OF POINTER ATTRIBUTES         *)
(*        IRC                O   RETURN_CODE                        *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                 *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        CASE ATTRIBUTE TYPE OF                                    *)
(*           INTEGER : CASE DECIMAL DIGIT OF INTEGER SIZE OF        *)
(*                        1, 2 : BYTES_PRECISION = 1                *)
(*                               SIZE = BYTES_PRECISION * ARRAY_LENGTH*)
(*                               BY_ALIGNMENT, PROCEDURE (4)        *)
(*                                                                  *)
```

```
(*                      3, 4 : BYTES_PRECISION = 2                  *)
(*                             SIZE = BYTES_PRECISION * ARRAY_LENGTH*)
(*                             HW_ALIGNMENT, PROCEDURE (3)          *)
(*                                                                  *)
(*                      5, 6,: BYTES_PRECISION = 4                  *)
(*                      7, 8,  SIZE = BYTES_PRECISION * ARRAY_LENGTH*)
(*                      9      FW_ALIGNMENT, PROCEDURE (2)          *)
(*                                                                  *)
(*          REAL    : CASE DECIMAL DIGIT OF REAL SIZE OF            *)
(*                      1,2,3: BYTES_PRECISION = 4                  *)
(*                      4,5,6  SIZE = BYTES_PRECISION * ARRAY_LENGTH*)
(*                      7      FW_ALIGNMENT, PROCEDURE (2)          *)
(*                                                                  *)
(*                      8, 9, 10, 11,                               *)
(*                      12, 13, 14, 15,                             *)
(*                      16  : BYTES_PRECISION = 8                   *)
(*                            SIZE = BYTES_PRECISION * ARRAY_LENGTH *)
(*                            DW_ALIGNMENT, PROCEDURE (1)           *)
(*                                                                  *)
(*          STRING  : BYTES_PRECISION = DECIMAL SIZE OF STRING      *)
(*                    SIZE = BYTES_PRECISION * ARRAY_LENGTH         *)
(*                    BY_ALIGNMENT, PROCEDURE (4)                   *)
(*                                                                  *)
(*          LOGICAL,                                                *)
(*          ENUMERATION : BYTES_PRECISION = 1                       *)
(*                        SIZE = BYTES_PRECISION * ARRAY_LENGTH     *)
(*                        BY_ALIGNMENT, PROCEDURE (4)               *)
(*                                                                  *)
(*          POINTER : BYTES_PRECISION = 0                           *)
(*                    SIZE =BYTES_PRECISION * ARRAY_LENGTH          *)
(*                    PNTR_ALIGNMENT, PROCEDURE (5)                 *)
(*                                                                  *)
(*          END CASE                                                *)
(*     PROCEDURE (1) : DW_ALIGNMENT                                 *)
(*        BUILD A LIST OF ATTRIBUTES ( NAME AND SIZE ) OF AN ENTITY *)
(*        REQUIRE FOR DOUBLE WORD ALIGNMENT                         *)
(*                                                                  *)
(*     PROCEDURE (2) : FW_ALIGNMENT                                 *)
(*        BUILD A LIST OF ATTRIBUTES ( NAME AND SIZE ) OF AN ENTITY *)
(*        REQUIRE FOR FULL WORD ALIGNMENT                           *)
(*                                                                  *)
(*     PROCEDURE (3) : HW_ALIGNMENT                                 *)
(*        BUILD A LIST OF ATTRIBUTES ( NAME AND SIZE ) OF AN ENTITY *)
(*        REQUIRE FOR HALF WORD ALIGNMENT                           *)
(*                                                                  *)
(*     PROCEDURE (4) : BY_ALIGNMENT                                 *)
(*        BUILD A LIST OF ATTRIBUTES ( NAME AND SIZE ) OF AN ENTITY *)
(*        REQUIRE FOR BYTES                                         *)
(*                                                                  *)
```

```
(*    PROCEDURE (5) : PNTR_ALIGNMENT                                  *)
(*       BUILD A LIST OF POINTER ATTRIBUTES OF AN ENTITY              *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED: 25 AUGUST 1986, M. H. CHOI, DBMA                *)
(*                                                                    *)
(* END %INCLUDE PHDECBYT ******************************************** *)
```

```
(* BEGIN %INCLUDE PHENTITY ********************************************)
(*                                                                    *)
PROCEDURE PHENTITY ( VAR   ENTITY_KEY     : ENTKEY;
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_POSITION    : INTEGER;
                     VAR   OFFSET_LIST_COUNT : INTEGER;
                     VAR   OFFSET_LIST    : T_OFFSET_LIST;
                     VAR   IRC            : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PHYSICALIZE THE ENTITY DEFINITIONS OF THE SUBSCHEMA         *)
(*        ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                   *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        ENTITY_KEY          I   ENTITY KEY OF DEFINITION TO BE      *)
(*                                PHYSICALIZE                         *)
(*        ARRAY_TABLE_POSIT   O   NUMBER OF ARRAYS                    *)
(*        ENUM_TABLE_POSIT    O   NUMBER OF ENUMERATIONS              *)
(*        CL_TABLE_POSIT      O   NUMBER OF POINTERS                  *)
(*        OFFSET_LIST_COUNT   O   NUMBER OF ATTRIBUTES IN THE LIST    *)
(*        OFFSET_LIST         O   LIST OF ATTRIBUTES OF AN ENTITY     *)
(*        NEW_SIZE            O   SIZE DIFFERENCE BETWEEN THE PREVIOUS*)
(*                                TOTAL GLOBAL SIZE AND THE NEW TOTAL *)
(*                                GLOBAL SIZE                         *)
(*        IRC                 O   RETURN_CODE                         *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH THE ATTRIBUTES THAT SPECIFIED THE ORDER        *)
(*           PHYSICALIZE THE ATTRIBUTES                               *)
(*        END LOOP                                                    *)
(*        LOOP THROUGH THE ATTRIBUTES THAT DID NOT SPECIFIED THE ORDER*)
(*           PHYSICALIZE THE ATTRIBUTES                               *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 4 SEPTEMBER 1987, M. H. CHOI, DBMA              *)
(*                                                                    *)
(* END %INCLUDE PHENTITY **********************************************)
```

```
(* BEGIN %INCLUDE PHGLOBAL ***************************************************)
(*                                                                             *)
PROCEDURE PHGLOBAL ( VAR   LIST_OF_GLOBALS  : LISTKEY;
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_TABLE_POSITION : INTEGER;
                     VAR   OFFSET_LIST_COUNT: INTEGER;
                     VAR   OFFSET_LIST      : T_OFFSET_LIST;
                     VAR   IRC              : RET_REC );
        SUBPROGRAM;
(*                                                                             *)
(*    $FUNCTION:                                                               *)
(*         PHYSICALIZE THE GLOBAL FIELDS OF THE SCHEMA                         *)
(*         ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                           *)
(*                                                                             *)
(*    $DESCRIPTION OF ARGUMENTS:                                               *)
(*         NAME               I/O  DESCRIPTION                                 *)
(*         ====               ===  ===========                                 *)
(*         SUBSCHEMA_KEY       I   SUBSCHEMA KEY                               *)
(*         ARRAY_TABLE_POSIT   O   NUMBER OF ARRAYS                            *)
(*         ENUM_TABLE_POSIT    O   NUMBER OF ENUMERATIONS                      *)
(*         CL_TABLE_POSIT      O   NUMBER OF POINTERS                          *)
(*         OFFSET_LIST_COUNT   O   NUMBER OF ATTRIBUTES IN THE LIST            *)
(*         OFFSET_LIST         O   LIST OF ATTRIBUTES OF AN ENTITY             *)
(*                                 ACCORDING TO BOUNDARY ALIGNMENT             *)
(*         IRC                 O   RETURN_CODE                                 *)
(*                                                                             *)
(*    $COMMONS:                                                                *)
(*                                                                             *)
(*    $ENVIRONMENT:                                                            *)
(*         LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                              *)
(*                                                                             *)
(*    $EXECUTION PROCEDURE:                                                    *)
(*                                                                             *)
(*    $PROCESSING DESCRIPTION:                                                 *)
(*         IF THERE ARE ANY GLOBAL FIELDS THEN                                 *)
(*            LOOP THROUGH EACH GLOBAL FIELDS                                  *)
(*               DETERMINE THE FIELD POSITION NUMBER                           *)
(*            END LOOP                                                         *)
(*            LOOP THROUGH THE FIELDS WITH FIELD POSITION                      *)
(*               PHYSICALIZE THE FIELD BY FIELD POSITION NUMBER                *)
(*            END LOOP                                                         *)
(*            LOOP THROUGH THE FIELDS WITHOUT THE FIELD POSITION               *)
(*               PHYSICALIZE THE FIELD BY BOUNDARY ALIGNMENT                   *)
(*            END LOOP                                                         *)
(*                                                                             *)
```

```
(* BEGIN %INCLUDE PHGLOBAL ********************************************)
(*                                                                    *)
PROCEDURE PHGLOBAL ( VAR   LIST_OF_GLOBALS  : LISTKEY;
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_TABLE_POSITION : INTEGER;
                     VAR   OFFSET_LIST_COUNT: INTEGER;
                     VAR   OFFSET_LIST      : T_OFFSET_LIST;
                     VAR   IRC              : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PHYSICALIZE THE GLOBAL FIELDS OF THE SCHEMA                 *)
(*        ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                   *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        SUBSCHEMA_KEY       I   SUBSCHEMA KEY                       *)
(*        ARRAY_TABLE_POSIT   O   NUMBER OF ARRAYS                    *)
(*        ENUM_TABLE_POSIT    O   NUMBER OF ENUMERATIONS              *)
(*        CL_TABLE_POSIT      O   NUMBER OF POINTERS                  *)
(*        OFFSET_LIST_COUNT   O   NUMBER OF ATTRIBUTES IN THE LIST    *)
(*        OFFSET_LIST         O   LIST OF ATTRIBUTES OF AN ENTITY     *)
(*                                ACCORDING TO BOUNDARY ALIGNMENT     *)
(*        IRC                 O   RETURN_CODE                         *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        IF THERE ARE ANY GLOBAL FIELDS THEN                         *)
(*           LOOP THROUGH EACH GLOBAL FIELDS                          *)
(*              DETERMINE THE FIELD POSITION NUMBER                   *)
(*           END LOOP                                                 *)
(*           LOOP THROUGH THE FIELDS WITH FIELD POSITION              *)
(*              PHYSICALIZE THE FIELD BY FIELD POSITION NUMBER        *)
(*           END LOOP                                                 *)
(*           LOOP THROUGH THE FIELDS WITHOUT THE FIELD POSITION       *)
(*              PHYSICALIZE THE FIELD BY BOUNDARY ALIGNMENT           *)
(*           END LOOP                                                 *)
(*                                                                    *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 25 NOVEMBER 1986, M. H. CHOI, DBMA                 *)
(*                                                                       *)
(* END %INCLUDE PHGLOBAL ***********************************************)
```

```
(* BEGIN %INCLUDE PHGTFLD ********************************************)
(*                                                                   *)
PROCEDURE PHGTFLD ( CONST FIELD_KEY        : ENTKEY;
                    VAR   DW_ROOT          : T_DW_POINTER;
                    VAR   FW_ROOT          : T_FW_POINTER;
                    VAR   HW_ROOT          : T_HW_POINTER;
                    VAR   BY_ROOT          : T_BY_POINTER;
                    VAR   PNTR_ROOT        : T_PNTR_POINTER;
                    VAR   OFFSET_LIST      : T_OFFSET_LIST;
                    VAR   OFFSET_LIST_COUNT : INTEGER;
                    VAR   IRC              : RET_REC );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        FIELD_KEY          I                                      *)
(*        STARTING_OFFSET    0                                      *)
(*        DW_ROOT            0                                      *)
(*        FW_ROOT            0                                      *)
(*        HW_ROOT            0                                      *)
(*        BY_ROOT            0                                      *)
(*        IRC                0   RETURN_CODE                        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 26 NOVEMBER 1986, M. H. CHOI, DBMA             *)
(*                                                                   *)
(* END %INCLUDE PHGTFLD **********************************************)
```

```
(* BEGIN %INCLUDE PHPOSITN ********************************************)
(*                                                                    *)
PROCEDURE PHPOSITN ( CONST LIST_OF_FIELDS : LISTKEY;
                     VAR   ORDER_INDEX    : INTEGER;
                     VAR   ORDER_REC      : T_GLOBAL_FIELD;
                     VAR   UNORDER_LIST   : LISTKEY;
                     VAR   IRC            : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        DETERMINE THE FIELD POSITION ORDER                          *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        LIST_OF_FIELDS     I   FIELDS TO DETERMINE THE FIELD        *)
(*                               POSITION ORDER                       *)
(*        ORDER_INDEX        O   NUMBER OF ATTRIBUTES THAT SPECIFIED  *)
(*                               THE FIELD POSITION NUMBER            *)
(*        ORDER_REC          O   LIST OF ATTRIBUTES THAT SPECIFIED    *)
(*                               THE FIELD POSITION NUMBER            *)
(*        UNORDER_LIST       O   LIST OF ATTRIBUTES THAT DID NOT      *)
(*                               SPECIFIED THE ORDER                  *)
(*        IRC                O   RETURN_CODE                          *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH THE ATTRIBUTES                                 *)
(*           IF ATTRIBUTE DID NOT SPECIFIED THE POSITION THEN         *)
(*              ATTACH TO THE UNORDER LIST                            *)
(*           ELSE                                                     *)
(*              ATTACH TO THE ORDER LIST                              *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 14 OCTOBER 1986, M. H. CHOI, DBMA               *)
(*                                                                    *)
(* END %INCLUDE PHPOSITN **********************************************)
```

```
(* BEGIN %INCLUDE PHSRTFLD ******************************************)
(*                                                                  *)
PROCEDURE PHSRTFLD ( VAR  DW_ROOT         : T_DW_POINTER;
                     VAR  FW_ROOT         : T_FW_POINTER;
                     VAR  HW_ROOT         : T_HW_POINTER;
                     VAR  BY_ROOT         : T_BY_POINTER;
                     VAR  PNTR_ROOT       : T_PNTR_POINTER;
                     VAR  OFFSET_LIST     : T_OFFSET_LIST;
                     VAR  TOTAL_COUNT     : INTEGER );
          SUBPROGRAM;
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        DETERMINE THE LOCATION OF ATTRIBUTES OF AN ENTITY ACCORDING *)
(*        TO BOUNDARY ALIGNMENT.                                    *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME              I/O  DESCRIPTION                        *)
(*        ====              ===  ===========                        *)
(*        DW_ROOT            I   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               DOUBLE WORD ALIGNMENT              *)
(*        FW_ROOT            I   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               FULL WORD ALIGNMENT                *)
(*        HW_ROOT            I   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               HALF WORD ALIGNMENT                *)
(*        BY_ROOT            I   LIST OF ATTRIBUTES WHICH REQUIRE FOR *)
(*                               BYTES                              *)
(*        PNTR_ROOT          I   LIST OF POINTER ATTRIBUTES          *)
(*        OFFSET_LIST        O   LIST OF ATTRIBUTES OF AN ENTITY     *)
(*                               ACCORDING TO BOUNDARY ALIGNMENT    *)
(*        TOTAL_COUNT        O   NUMBER OF ENTRIES IN THE LIST       *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                 *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        LOOP THROUGH EACH LIST OF ALIGNMENTS TABLE ACCORDING TO   *)
(*           BOUNDARY ALIGNMENT                                     *)
(*                 (1) LIST OF DOUBLE WORD ALIGNMENT                *)
(*                 (2) LIST OF FULL WORD ALIGNMENT                  *)
(*                 (3) LIST OF HALF WORD ALIGNMENT                  *)
(*                 (4) LIST OF BYTE ALIGNMENT                       *)
(*           STORE ATTRIBUTE NAME AND SIZE INTO LIST OF OFFSET-LIST  *)
(*              TABLE                                               *)
```

```
(*           IF ATTRIBUTE NAME AND SIZE ARE THE FIRST ONE TO STORE    *)
(*              OFFSET = 12                                           *)
(*           ELSE                                                     *)
(*              OFFSET = PREVIOUS OFFSET + PREVIOUS SIZE              *)
(*           INCREMENT NUMBER OF ENTRIES IN THE TABLE                 *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)            *)
(*        ORIGINATED: 18 SEPTEMBER 1986, M. H. CHOI, DBMA             *)
(*                                                                    *)
(* END %INCLUDE PHSRTFLD  ********************************************)
```

```
(* BEGIN %INCLUDE PHSRTORD ********************************************)
(*                                                                    *)
PROCEDURE PHSRTORD ( CONST ORDER_INDEX      : INTEGER;
                     VAR   ORDER_REC        : T_GLOBAL_FIELD );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        SORT ATTRIBUTES BY THE FIELD POSITION NUMBER                *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        ORDER_INDEX         I   NUMBER OF ATTRIBUTES THAT SPECIFIED *)
(*                                THE FIELD POSITION NUMBER           *)
(*        ORDER_REC           O   LIST OF ATTRIBUTES THAT SPECIFIED   *)
(*                                THE FIELD POSITION NUMBER           *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        LOOP THROUGH LIST OF ATTRIBUTES THAT SPECIFIED THE FIELD    *)
(*        POSITION NUMBER AND PUT THEM IN ASCENDING ORDER.            *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 18 NOVEMBER 1986, M. H. CHOI, DBMA              *)
(*                                                                    *)
(* END %INCLUDE PHSRTORD **********************************************)
```

```
(* BEGIN %INCLUDE PHSUBTYP ********************************************)
(*                                                                    *)
PROCEDURE PHSUBTYP ( CONST SUBTYPE_KEY     : ENTKEY;
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_POSITION     : INTEGER;
                     VAR   OFFSET_LIST_COUNT : INTEGER;
                     VAR   OFFSET_LIST     : T_OFFSET_LIST;
                     VAR   IRC             : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PHYSICALIZE THE SUPER TYPES                                 *)
(*        ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                   *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        SUBSCHEMA_KEY       I   SUBSCHEMA KEY OF THE ENTITY         *)
(*                                DEFINITIONS TO BE PHYSICALIZE       *)
(*        ARRAY_TABLE_POSIT   O   NUMBER OF ARRAYS                    *)
(*        ENUM_TABLE_POSIT    O   NUMBER OF ENUMERATIONS              *)
(*        CL_TABLE_POSIT      O   NUMBER OF POINTERS                  *)
(*        OFFSET_LIST_COUNT   O   NUMBER OF ATTRIBUTES IN THE LIST    *)
(*        OFFSET_LIST         O   LIST OF ATTRIBUTES OF AN ENTITY     *)
(*        NEW_SIZE            O   SIZE DIFFERENCE BETWEEN THE PREVIOUS *)
(*                                TOTAL GLOBAL SIZE AND THE NEW TOTAL *)
(*                                GLOBAL SIZE                         *)
(*        IRC                 O   RETURN_CODE                         *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        PHYSICALIZE THE ATTRIBUTES WITHIN THE SUPERTYPE             *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 4 SEPTEMBER 1987, M. H. CHOI, DBMA              *)
(*                                                                    *)
(* END %INCLUDE PHSUBTYP **********************************************)
```

```
(* BEGIN %INCLUDE PHWOFPOS ********************************************)
(*                                                                     *)
PROCEDURE PHWOFPOS ( VAR   UNORDER_LIST   : LISTKEY:
                     VAR   ARRAY_TABLE_POSITION : INTEGER;
                     VAR   ENUM_TABLE_POSITION : INTEGER;
                     VAR   CL_POSITION    : INTEGER;
                     VAR   OFFSET_LIST_COUNT : INTEGER;
                     VAR   OFFSET_LIST    : T_OFFSET_LIST;
                     VAR   IRC            : RET_REC );
        SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        PHYSICALIZE THE ATTRIBUTES THAT DID NOT SPECIFIED THE        *)
(*        FIELD POSITION NUMBER ( DETERMINE ATTRIBUTE SIZE AND         *)
(*        LOCATION )                                                   *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        UNORDER_LIST       O   LIST OF ATTRIBUTES WITHOUT THE FIELD  *)
(*                               POSITION NUMBER                       *)
(*        ARRAY_TABLE_POS    O   NUMBER OF ARRAYS                      *)
(*        ENUM_TABLE_POSR    O   NUMBER OF ENUMERATIONS                *)
(*        CL_POSITION        O   NUMBER OF POINTERS                    *)
(*        OFFSET_LIST_COUNT  O   NUMBER OF ATTRIBUTES IN THE LIST      *)
(*        OFFSET_LIST        O   LIST OF ATTRIBUTES OF AN ENTITY       *)
(*        IRC                O   RETURN_CODE                           *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        LOOP THROUGH THE ATTRIBUTES THAT DID NOT SPECIFIED THE       *)
(*          FIELD POSITION NUMBER                                      *)
(*           PHYSICALIZE THE ATTRIBUTE                                 *)
(*        END LOOP                                                     *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 14 OCTOBER 1986, M. H. CHOI, DBMA                *)
(*                                                                     *)
(* END %INCLUDE PHWOFPOS **********************************************)
```

```
(* BEGIN %INCLUDE PHYSICAL ********************************************)
(*                                                                    *)
PROCEDURE PHYSICAL ( VAR   SUBSCHEMA_KEY  : ENTKEY;
                     VAR   IRC            : RET_REC );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        PHYSICALIZE THE ENTITY DEFINITIONS OF THE SUBSCHEMA         *)
(*        ( DETERMINE ATTRIBUTE SIZE AND LOCATION )                   *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME               I/O  DESCRIPTION                         *)
(*        ====               ===  ===========                         *)
(*        SUBSCHEMA_KEY       I   SUBSCHEMA KEY OF THE ENTITY         *)
(*                                DEFINITIONS TO BE PHYSICALIZE       *)
(*        IRC                 O   RETURN_CODE                         *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        PHYSICALIZE THE GLOBAL FIELDS                               *)
(*        CREATE A LIST OF ENTITIES INCLUSIVELY BY ENTITY KIND FOR    *)
(*                                              SPECIFIC SUBSCHEMA    *)
(*        LOOP THROUGH LIST OF ENTITIES                               *)
(*           IF ENTITY IS NOT PHYSICALIZED THEN                       *)
(*              LOOP THROUGH EACH ATTRIBUTES                          *)
(*                 IF ATTRIBUTES DID NOT SPECIFIED THE POSITION THEN  *)
(*                    ATTACH TO THE UNORDER LIST                      *)
(*                 ELSE                                               *)
(*                    ATTACH TO ORDER LIST                            *)
(*              END LOOP                                              *)
(*              IF ORDER LIST IS NOT EMPTY THEN                       *)
(*                 SORT ORDER LIST  : PROCEDURE SRTORDER;             *)
(*                 LOOP THROUGH THE ORDER LIST                        *)
(*                    PHYSICALIZE THE ATTRIBUTE                       *)
(*              END IF                                                *)
(*              IF UNORDER LIST IS NOT EMPTY THEN                     *)
(*                 PHYSICALIZE THE UNORDER LIST                       *)
(*              END IF                                                *)
(*           END IF                                                   *)
(*           ELSE                                                     *)
(*              IF TOTAL SIZE OF THE GLOBAL FIELDS HAS BEEN CHANGED   *)
(*                 RECALCULATE THE OFFSET OF EACH ATTRIBUTES          *)
(*              END IF                                                *)
```

```
(*        END LOOP                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 14 OCTOBER 1986, M. H. CHOI, DBMA               *)
(*                                                                    *)
(* END %INCLUDE PHYSICAL ********************************************)
```

```
(* BEGIN %INCLUDE PSORDER ********************************************)
(*                                                                   *)
PROCEDURE PSORDER ( CONST RUNTIME          : T_RUN_TIME;
                    VAR   PS_ORDER         : T_PS_ORDER );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        DETERMINE THE PHYSICAL SCHEMA ORDER OF ENTITY DEFINITION   *)
(*        BY ITS OFFSET.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        PS_ORDER           O   LIST OF ATTRIBUTES IN PHYSICAL      *)
(*                               ORDER                               *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        LOOP THROUGH THE NUMBER OF ATTRIBUTES IN THE DEFINITION    *)
(*          INSERT THE OFFSET IN THE PHYSICAL SCHEMA ORDER TABLE     *)
(*        END LOOP                                                   *)
(*        LOOP THROUGH THE NUMBER OF ATTRIBUTES IN THE DEFINITION    *)
(*          IF CURRENT OFFSET GREATER THAN NEXT OFFSET THEN          *)
(*             SWITCH CURRENT OFFSET WITH NEXT OFFSET                *)
(*          END IF                                                   *)
(*        END LOOP                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                *)
(*                                                                   *)
(* END %INCLUDE PSORDER **********************************************)
```

```
(* BEGIN %INCLUDE PSRABNDS ********************************************)
(*                                                                     *)
PROCEDURE PSRABNDS ( VAR   PSRDATA           : TEXT;
                     CONST NO_OF_DIMEN       : INTEGER;
                     CONST STARTING_ARRAY_POSITION : INTEGER;
                     VAR   POINTER           : T_VARIANT_POINTER;
                     CONST ENUM_INDEX        : INTEGER );
        SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        WRITE LOW-BOUND AND UPPER-BOUND FOR THE ARRAY ATTRIBUTE      *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT SEQUENTIAL FILE*)
(*        NO_OF_DIMENS       I   NUMBER OF ARRAY DIMENSIONS            *)
(*        STARTING_ARRAY_PO  I   STARTING POSITION IN THE ARRAY TABLE  *)
(*        POINTER            I   POINTER TO ARRAY INFORMATION TABLE     *)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM     *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        LOOP THROUGH THE NUMBER OF DIMENSIONS                        *)
(*          WRITE LOW-BOUND AND UPPER-BOUND                            *)
(*        END LOOP                                                     *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        ORIGINATED: 24 MARCH 1987, M. H. CHOI, DBMA                  *)
(*                                                                     *)
(* END %INCLUDE PSRABNDS **********************************************)
```

```
(* BEGIN %INCLUDE PSRADB ***********************************************)
(*                                                                      *)
PROCEDURE PSRADB ( VAR   PSRDATA          : TEXT;
                   CONST RUNTIME          : T_RUN_TIME;
                   CONST ENTRY            : INTEGER;
                   CONST PS_ORDER         : T_PS_ORDER );
         SUBPROGRAM;
(*                                                                      *)
(*   $FUNCTION:                                                         *)
(*        WRITE THE BASIC RECORD OF AN ENTITY TO A PHYSICAL SCHEMA      *)
(*        REPORT FILE                                                   *)
(*                                                                      *)
(*   $DESCRIPTION OF ARGUMENTS:                                         *)
(*        NAME              I/O  DESCRIPTION                            *)
(*        ====              ===  ===========                            *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT SEQUENTIAL FILE*)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION        *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION         *)
(*        PS_ORDER           I   LIST OF ATTRIBUTES IN PHYSICAL ORDER  *)
(*                                                                      *)
(*   $COMMONS:                                                          *)
(*                                                                      *)
(*   $ENVIRONMENT:                                                      *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*                                                                      *)
(*   $EXECUTION PROCEDURE:                                              *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM      *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*        WRITE THE BASIC ATTRIBUTES ( INTEGER, REAL, STRING, LOGICAL)*)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                   *)
(*                                                                      *)
(* END %INCLUDE PSRADB *************************************************)
```

```
(* BEGIN %INCLUDE PSRARRAY ********************************************)
(*                                                                    *)
PROCEDURE PSRARRAY ( VAR   PSRDATA          : TEXT;
                     CONST RUNTIME          : T_RUN_TIME;
                     CONST ENTRY            : INTEGER;
                     VAR   CL_HEADING_FLAG  : BOOLEAN;
                     VAR   ENUM_HEADING_FLAG : BOOLEAN;
                     CONST PS_ORDER         : T_PS_ORDER );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        WRITE THE ARRAY ATTRIBUTE OF AN ENTITY TO A SEQUENTIAL FILE *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT TEXT FILE     *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION       *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION        *)
(*        CL_HEADING_FLAG    I   FLAG TO DETERMINE WHETHER THE HEADING *)
(*                               HAS BEEN WRITTEN                     *)
(*        PS_ORDER           I   LIST OF ATTRIBUTES IN PHYSICAL ORDER  *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        OBTAIN THE NUMBER OF DIMENSIONS                             *)
(*        OBTAIN THE STARTING POSITION OF ARRAY TABLE                 *)
(*        CASE DATA TYPE OF                                           *)
(*          IN-ADB : WRITE BASIC DEFINITION                           *)
(*                   PSRABNDS ( EXTERNAL SUBPROGRAM TO WRITE LOW-BOUND*)
(*                              AND UPPER-BOUND )                     *)
(*          IN-CL  : WRITE BASIC DEFINITION                           *)
(*                   PSRCL ( EXTERNAL SUBPROGRAM FOR CONSTITUENT LIST)*)
(*                   PSRABNDS ( EXTERNAL SUBPROGRAM TO WRITE LOW-BOUND*)
(*                              AND UPPER-BOUND )                     *)
(*        END CASE                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                 *)
(*                                                                    *)
(* END %INCLUDE PSRARRAY **********************************************)
```

```
(* BEGIN %INCLUDE PSRCL ********************************************)
(*                                                                  *)
PROCEDURE PSRCL ( VAR   PSRDATA          : TEXT;
                  CONST RUNTIME          : T_RUN_TIME;
                  CONST ENTRY            : INTEGER;
                  VAR   CL_HEADING_FLAG  : BOOLEAN;
                  CONST NO_OF_DIMEN      : INTEGER;
                  CONST STARTING_POSITION : INTEGER;
                  VAR   ARRAY_POINTER    : T_VARIANT_POINTER );
        SUBPROGRAM;
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        WRITE THE CONSTITUENT REFERENCES OF AN ENTITY TO A        *)
(*        SEQUENTIAL FILE                                           *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME              I/O  DESCRIPTION                        *)
(*        ====              ===  ==========                         *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT TEXT FILE   *)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION     *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION      *)
(*        CL_HEADING_FLAG    I   FLAG TO DETERMINE WHETHER THE HEADING *)
(*                               HAS BEEN WRITTEN                   *)
(*        NO_OF_DIMEN        I   NUMBER OF ARRAY DIMENSIONS         *)
(*        STARTING_ARRAY_PO  I   STARTING POSITION IN THE ARRAY TABLE *)
(*        ARRAY_POINTER      I   POINTER TO ARRAY TABLE             *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                 *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM  *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        IF NOT ARRAY ATTRIBUTE THEN                               *)
(*           WRITE BASIC DEFINITION                                 *)
(*        END IF                                                    *)
(*        OBTAIN THE NUMBER OF ELIGIBLE KINDS                       *)
(*        OBTAIN STARTING POSITION OF CONSTITUENT LIST TABLE        *)
(*        LOOP THROUGH THE NUMBER OF ELIGIBLE KINDS                 *)
(*           WRITE ELIGIBLE KIND IN THE CONSTITUENT LIST TABLE      *)
(*        END LOOP                                                  *)
(*                                                                  *)
```

```
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA              *)
(*                                                                 *)
(* END %INCLUDE PSRCL ************************************************)
```

```
(* BEGIN %INCLUDE PSRENUM ********************************************)
(*                                                                   *)
PROCEDURE PSRENUM ( VAR   PSRDATA          : TEXT;
                    CONST RUNTIME          : T_RUN_TIME;
                    CONST ENTRY            : INTEGER;
                    CONST PS_ORDER         : T_PS_ORDER;
                    VAR   ENUM_HEADING_FLAG : BOOLEAN;
                    CONST ENUM_TABLE_INDEX : INTEGER;
                    CONST NO_OF_VALUES     : INTEGER );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        WRITE THE ENUMERATION ATTRIBUTE OF AN ENTITY TO A          *)
(*        SEQUENTIAL FILE                                            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT SEQUENTIAL FILE*)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        ENTRY              I   ENTRY ORDER IN THE DEFINITION       *)
(*        PS_ORDER           I   LIST OF ATTRIBUTES IN PHYSICAL ORDER *)
(*        ENUM_HEADING_FLAG  I   FLAG TO DETERMINE WHETHER THE HEADING *)
(*                               HAS BEEN WRITTEN                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        WRITE BASIC DEFINITION                                     *)
(*        OBTAIN THE NUMBER OF ENUMERATION VALUES                    *)
(*        OBTAIN THE STARTING POSITION OF ENUMERATION VALUE TABLE    *)
(*        LOOP THROUGH THE NUMBER OF ENUMERATION VALUES              *)
(*           WRITE THE ENUMERATION VALUE FROM THE TABLE              *)
(*        END LOOP                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                *)
(*                                                                   *)
(* END %INCLUDE PSRENUM **********************************************)
```

```
(* BEGIN %INCLUDE PSREPORT ********************************************)
(*                                                                    *)
PROCEDURE PSREPORT ( VAR   SUBSCHEMA_KEY : ENTKEY;
                     VAR   IRC           : RET_REC );
          SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        FILE PHYSICAL SCHEMA REPEORT TO SEQUENTIAL FILE             *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        SUBSCHEMA_KEY      I                                        *)
(*        IRC                O                                       *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                 *)
(*                                                                    *)
(* END %INCLUDE PSREPORT **********************************************)
```

```
(* BEGIN %INCLUDE PSRHEAD ********************************************)
(*                                                                   *)
PROCEDURE PSRHEAD ( VAR   PSRDATA          : TEXT;
                    CONST RUNTIME          : T_RUN_TIME;
                    CONST PAGE_NO          : INTEGER );
        SUBPROGRAM;
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        WRITE THE PHYSICAL SCHEMA REPORT HEADING                   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT SEQUENTIAL FILE*)
(*        RUNTIME            I   CONTAINS THE ENTITY DEFINITION      *)
(*        PAGE_NO            I   NUMBER OF PAGE                      *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                  *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM   *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        WRITE THE HEADING                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                *)
(*                                                                   *)
(* END %INCLUDE PSRHEAD **********************************************)
```

```
(* BEGIN %INCLUDE PSRINDEX ********************************************)
(*                                                                    *)
PROCEDURE PSRINDEX ( VAR   PSRDATA          : TEXT;
                     CONST PAGE_NO          : INTEGER;
                     CONST ENTITY_INDEX     : T_ENTITY_INDEX;
                     CONST LIST_OF_ENTITIES : LISTKEY );
        SUBPROGRAM;
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        WRITE THE TABLE OF CONTENTS FOR THE PHYSICAL SCHEMA REPORT  *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME              I/O  DESCRIPTION                          *)
(*        ====              ===  ===========                          *)
(*        PSRDATA            O   PHYSICAL SCHEMA REPORT TEXT FILE     *)
(*        PAGE_NO            I   PAGE NUMBER                          *)
(*        ENTITY_INDEX       I   ENTITY NAME AND THE KIND NUMBER      *)
(*        LIST_OF_ENTITIES   I   LIST OF ENTITY KEYS TO SORT LATER    *)
(*                               BY THE ENTITY NAME                   *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                   *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        CALLED FROM EITHER PASCAL OR FORTRAN APPLICATION PROGRAM    *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        WRITE THE TABLE OF CONTENTS IN ORDER OF ENTITY KIND         *)
(*        WRITE THE TABLE OF CONTENTS IN ORDER OF ENTITY NAME         *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*        ORIGINATED: 09 MARCH 1987, M. H. CHOI, DBMA                 *)
(*                                                                    *)
(* END %INCLUDE PSRINDEX **********************************************)
```

```
(* %INCLUDE REARRAY *)
(**)
PROCEDURE REARRAY(VAR MESS    : MESSAGE;
                  VAR LBND    : CHAR8;
                  VAR HBND    : CHAR8;
                  VAR FTYPE   : CHAR12;
                  VAR NEXT_OP : OPERATIONS;
                  VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      THIS FUNCTION:                                                    *)
(*                    DISPLAYS THE REVIEW ARRAY PANEL.                    *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        LBND           I   THE LOWER BOUND OF THE ARRAY                 *)
(*        HBND           I   THE UPPER BOUND OF THE ARRAY                 *)
(*        FTYPE          I   THE ARRAY TYPE                               *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                    *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE REVIEW ARRAY PANEL (REARRAY) BY MAKING ISPLNK       *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED      *)
(*        TYPE.                                                           *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE RECLASS *)
(**)
PROCEDURE RECLASS(VAR MESS       : MESSAGE;
                  VAR NAME       : T_NAME;
                  VAR KNUM       : CHAR8;
                  VAR CLAS       : T_ARRAY23;
                  VAR ARRAY_SIZE : INTEGER;
                  VAR MEMBER     : T_NAME;
                  VAR NEXT_OP    : OPERATIONS;
                  VAR RR         : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW CLASS PANEL                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE CLASS NAME                          *)
(*        KNUM           I   THE CLASS KIND NUMBER                   *)
(*        CLAS           I   THE ARRAY OF MEMBERS                    *)
(*        ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        MEMBER         O   THE MEMBER SELECTED                     *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW CLASS PANEL (RECLASS) BY MAKING ISPLNK  *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
(*                                                                   *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE REDEFTYP *)
(**)
PROCEDURE REDEFTYP(VAR MESS    : MESSAGE;
                   VAR NAME    : CHAR16;
                   VAR FTYPE   : CHAR12;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE DEFINED TYPE REVIEW PANEL         *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           O   THE NAME OF THE DEFINED TYPE ENTERED   *)
(*        FTYPE          O   TYPES INTEGER,STRING,REAL...ETC.       *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW DEFINED TYPE PANEL (REDEFTYP) BY MAKING *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE REENTITY *)
(**)
PROCEDURE REENTITY(VAR MESS    : MESSAGE;
                   VAR NAME    : T_NAME;
                   VAR KNUM    : CHAR8;
                   VAR MEMBERS : T_ARRAY16;
                   VAR SIZE    : INTEGER;
                   VAR FNAME   : T_NAME;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*            THIS PROCEDURE :                                            *)
(*                 DISPLAYS THE REVIEW ENTITY MENU                        *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NAME           I   THE ENTITY NAME                              *)
(*        KNUM           I   THE ENTITY KIND NUMBER                       *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM          *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS             *)
(*        FNAME          O   THE MEMBER SELECTED                          *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                    *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DISPLAY THE REVIEW ENTITY PANEL (REENTITY) BY MAKING ISPLNK     *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED      *)
(*        TYPE.                                                           *)
(*                                                                        *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE REENUM *)
(**)
PROCEDURE REENUM(VAR MESS    : MESSAGE;
                 VAR MEMBERS : T_ARRAY16;
                 VAR SIZE    : INTEGER;
                 VAR NEXT_OP : OPERATIONS;
                 VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                    DISPLAYS THE REVIEW ENUMERATION MENU                 *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO DISPLAY              *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF NUMBERS             *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISLAY THE REVIEW ENUMERATION MENU (REENUM) BY MAKING            *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN           *)
(*        ENUMERATED TYPE.                                                 *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE REFIELD *)
(**)

PROCEDURE REFIELD(VAR MESS     : MESSAGE;
                  VAR NAME     : T_NAME;
                  VAR POS      : CHAR8;
                  VAR PURP     : CHAR8;
                  VAR REQD     : CHAR8;
                  VAR DEPD     : CHAR12;
                  VAR FTYPE    : CHAR12;
                  VAR FLDT     : CHAR9;
                  VAR NEXT_OP  : OPERATIONS;
                  VAR RR       : RET_REC);

SUBPROGRAM;

(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS PROCEDURE :                                           *)
(*                         DISPLAYS THE REVIEW FIELD PANEL           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE NAME OF THE ARRAY                   *)
(*        PURP           I   THE PURPOSE OF THE ARRAY                *)
(*        REQD           I   THE REQUIREDNESS OF THE ARRAY           *)
(*        DEPD           I   THE DEPENDENCE/INDEPENDENCE OF THE ARRAY *)
(*        FTYPE          I   THE TYPE OF ELEMENT STORED IN THE ARRAY  *)
(*        FLDT           I   THE TYPE OF ARRAY (GLOBAL, STRUCTURE,    *)
(*                                             ENTITY)               *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE REVIEW FIELD PANEL (REFIELD) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED   *)
(*        TYPE.                                                        *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*        NONE                                                         *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
```

```
(* %INCLUDE REFIELD1 *)
(**)
PROCEDURE REFIELD1(VAR MESS       : MESSAGE;
                   VAR FIELD_TYPE : T_FIELDTYPE;
                   VAR NAME       : T_NAME;
                   VAR KNUM       : CHAR8;
                   VAR MEMBERS    : T_ARRAYID;
                   VAR SIZE       : INTEGER;
                   VAR FNAME      : T_NAME;
                   VAR NEXT_OP    : OPERATIONS;
                   VAR RR         : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*            THIS PROCEDURE :                                       *)
(*                 DISPLAYS THE REVIEW FIELD A MENU OR THE           *)
(*                 REVIEW FIELD B MENU                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        FIELD_TYPE     I   THE TYPE OF FIELD TO BE REVIEWED        *)
(*        NAME           I   THE ENTITY NAME                         *)
(*        KNUM           I   THE ENTITY KIND NUMBER                  *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM     *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        FNAME          O   THE MEMBER SELECTED                     *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE REVIEW ENTITY PANEL (REFIELD1) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*        TYPE.                                                       *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                             *)
(*                                                                    *)
(*    REVISED: 09/28/87          C. H. MOHME                DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                          *)
(*                                                                    *)
(*    REVISED: 07/02/87          C. H. MOHME                DBMA     *)
(*     CHANGED CURSOR POSITIONING.                                    *)
(*                                                                    *)
(*    ORIGINATED: 06/25/86       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE REFIELD1 *)
```

```
(* %INCLUDE REFIELD2 *)
(**)
PROCEDURE REFIELD2(VAR MESS     : MESSAGE;
                   VAR NAME     : T_NAME;
                   VAR POS      : CHAR8;
                (* VAR PURP     : CHAR8; *)
                   VAR REQD     : CHAR8;
                (* VAR DEPD     : CHAR12; *)
                   VAR COM      : CHAR50;
                   VAR FTYPE    : CHAR12;
                   VAR FLDT     : CHAR9;
                   VAR NEXT_OP  : OPERATIONS;
                   VAR RR       : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS PROCEDURE :                                                 *)
(*                          DISPLAYS THE REVIEW FIELD PANEL                *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NAME           I   THE NAME OF THE ARRAY                        *)
(*        POS            I   THE POSITION OF THE FIELD IN THE ADB         *)
(*        PURP           I   THE PURPOSE OF THE ARRAY                     *)
(*        REQD           I   THE REQUIREDNESS OF THE ARRAY                *)
(*        DEPD           I   THE DEPENDENCE/INDEPENDENCE OF THE ARRAY     *)
(*        FTYPE          I   THE TYPE OF ELEMENT STORED IN THE ARRAY      *)
(*        FLDT           I   THE TYPE OF ARRAY (GLOBAL, STRUCTURE,        *)
(*                                             ENTITY)                     *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE REVIEW FIELD PANEL (REFIELD2) BY MAKING ISPLNK   *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED   *)
(*        TYPE.                                                        *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*        NONE                                                         *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
(*    REVISED: 09/28/87          C. H. MOHME                 DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                           *)
(*                                                                     *)
(*    REVISED: 08/13/87          C. H. MOHME                 DBMA     *)
(*     CHANGED PANEL OPTION NUMBERS; CHANGED FIELD ADB DATA; ADDED     *)
(*     LIST AND SET.  NOTE:  THE LIST AND SET DATA TYPES ARE IMPLE-    *)
(*     MENTED IN THE SOFTWARE AS AN ARRAY.  A FIELD WAS ADDED TO THE   *)
(*     ARRAY ADB TO SPECIFY WHETHER THE ARRAY IS A CONCEPTUAL ARRAY,   *)
(*     LIST, OR SET.                                                   *)
(*                                                                     *)
(*    REVISED: 07/02/87          C. H. MOHME                 DBMA     *)
(*     CHANGED CURSOR POSITIONING.                                     *)
(*                                                                     *)
(*    ORIGINATED: 07/07/86       C. H. MOHME                 DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE REFIELD2 *)
```

```
(* %INCLUDE REFSUP *)
(**)
  PROCEDURE REFSUP(VAR IRC          :  RET_REC;
                   VAR TRANS_STACK  :  TRANSPTR;
                   VAR TOKEN_VALUE  :  T_TOKEN_VALUE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*       Batch Interface routine that attempts to resolve a reference *)
(*       to a supertype.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   INTERNAL RETURN CODE                   *)
(*        TRANS_STACK    I/O  TRANSACTION STACK                      *)
(*        TOKEN_VALUE    I/O  TOKEN VALUE FROM BATCH INPUT           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE BATCH INPUT    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*                                                                   *)
(*    INITIALIZE VARIABLES                                           *)
(*    DETERMINE IF THE SUPERTYPE HAS ALREADY BEEN MODELED.           *)
(*    IF SUPERTYPE MODELED, PUSH ITS KEY ONTO THE TRANSACTION STACK. *)
(*    IF SUPERTYPE NOT MODELED, PUSH UNRESOLVED TRANSACTION ONTO THE *)
(*      TRANSACTION STACK.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    ORIGINATED: 09/29/87       C. H. MOHME                 DBMA    *)
(*                                                                   *)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE REFSUP *)
```

```
(* %INCLUDE REINTGR *)
(**)
PROCEDURE REINTGR(VAR MESS    : MESSAGE;
                  VAR PREC    : CHAR8;
                  VAR NEXT_OP : OPERATIONS;
                  VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW INTEGER MENU               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        PREC           O   THE PRECISION OF THE INTEGER ENTERED    *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW INTEGER PANEL (REINTGR) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*        TYPE.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE RELIST *)
(**)
PROCEDURE RELIST(VAR MESS    : MESSAGE;
                 VAR MIN     : CHAR8;
                 VAR MAX     : CHAR8;
                 VAR FTYPE   : CHAR12;
                 VAR NEXT_OP : OPERATIONS;
                 VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW LIST PANEL.                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        MIN            I   THE MINIMUM NUMBER OF OCCURRENCES IN THE *)
(*                           LIST                                    *)
(*        MAX            I   THE MAXIMUM NUMBER OF OCCURRENCES IN THE *)
(*                           LIST                                    *)
(*        FTYPE          I   THE LIST TYPE                           *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW LIST PANEL (RELIST) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
```

```
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                             *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    REVISED: MM/DD/YY          I. M. APERSON              GROUP_ID *)
(*     DESCRIPTION OF THE CHANGE MADE                                 *)
(*                                                                    *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE RELIST *)
```

```
(* %INCLUDE REPNTR *)
(**)
PROCEDURE REPNTR(VAR MESS       : MESSAGE;
                 VAR MEMBERS    : T_ARRAY23;
                 VAR ARRAY_SIZE : INTEGER;
                 VAR MEMBER     : T_NAME;
                 VAR NEXT_OP    : OPERATIONS;
                 VAR RR         : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW POINTER MENU               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM      *)
(*        ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS         *)
(*        MEMBER         O   THE MEMBER SELECTED                      *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW POINTER PANEL (REPNTR) BY MAKING ISPLNK  *)
(*        CALLS.   THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
(*                                                                   *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE REREAL *)
(**)
PROCEDURE REREAL(VAR MESS    : MESSAGE;
                 VAR SIZE    : CHAR8;
                 VAR NEXT_OP : OPERATIONS;
                 VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW REAL PANEL                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        SIZE           O   THE SIZE OF THE REAL ENTERED            *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW REAL PANEL (REREAL) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE RESET *)
(**)
PROCEDURE RESET(VAR MESS    : MESSAGE;
                VAR MIN     : CHAR8;
                VAR MAX     : CHAR8;
                VAR FTYPE   : CHAR12;
                VAR NEXT_OP : OPERATIONS;
                VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE REVIEW SET PANEL.                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MIN            I   THE MINIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        MAX            I   THE MAXIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        FTYPE          I   THE SET TYPE                                 *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE REVIEW SET PANEL (RESET) BY MAKING ISPLNK            *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.                                                            *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
```

```
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*    REVISED: MM/DD/YY          I. M. APERSON              GROUP_ID *)
(*     DESCRIPTION OF THE CHANGE MADE                                   *)
(*                                                                       *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE RESET *)
```

```
(* %INCLUDE RESTRING *)
(**)
PROCEDURE RESTRING(VAR MESS    : MESSAGE;
                   VAR SLEN    : CHAR8;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW STRING PANEL               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        SIZE           O   THE SIZE OF THE STRING ENTERED          *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW STRING PANEL (RESTRING) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.                                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE RESTRUC *)
(**)
PROCEDURE RESTRUC(VAR MESS    : MESSAGE;
                  VAR MEMBERS : T_ARRAY16;
                  VAR SIZE    : INTEGER;
                  VAR MEMBER  : T_NAME;
                  VAR NEXT_OP : OPERATIONS;
                  VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW STRUCTURE PANEL            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM      *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS         *)
(*        MEMBER         O   THE MEMBER SELECTED                      *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW STRUCTURE PANEL (RESTRUC) BY MAKING     *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE RESUBSCM *)
(**)
PROCEDURE RESUBSCM(VAR MESS    : MESSAGE;
                   VAR NAME    : T_NAME;
                   VAR MEMBERS : T_ARRAY23;
                   VAR SIZE    : INTEGER;
                   VAR MEMBER  : T_NAME;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE REVIEW SUBSCHEMA PANEL            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE SUBSCHEMA NAME                      *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM     *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        MEMBER         O   THE MEMBER SELECTED                     *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW SUBSCHEMA PANEL (RESUBSCM) BY MAKING    *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE RESUPTYP *)
(**)
PROCEDURE RESUPTYP(VAR MESS    : MESSAGE;
                   VAR NAME    : T_NAME;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*             THIS PROCEDURE :                                      *)
(*                  DISPLAYS THE REVIEW SUPERTYPE MENU               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE SUPERTYPE NAME                      *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE REVIEW SUPERTYPE PANEL (RESUPTYP) BY MAKING    *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN     *)
(*        ENUMERATED TYPE.                                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID  *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
```

```
(*    REVISED: 09/28/87          C. H. MOHME                  DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                            *)
(*                                                                      *)
(*    REVISED: 07/02/87          C. H. MOHME                  DBMA     *)
(*     CHANGED CURSOR POSITIONING.                                      *)
(*                                                                      *)
(*    ORIGINATED: 06/25/86       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE RESUPTYP *)
```

```
(* BEGIN %INCLUDE RSCPAI ************************************************)
(*                                                                       *)
PROCEDURE RSCPAI ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_ARRAY_INDEX;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        COPY THE ARRAY INDEX TABLE INFORMATION INTO THE RUN-TIME       *)
(*        SUBSCHEMA.                                                     *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME                 I/O  DESCRIPTION                         *)
(*        ====                 ===  ===========                         *)
(*        INPUT_VALUE           I   INPUT VALUE OF ARBITRARY SIZE       *)
(*        OUTPUT_VALUE          O   OUTPUT VALUE OF ARBITRARY SIZE      *)
(*        SIZE_OF_VALUE         I   SIZE OF VALUE TO BE COPIED          *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        CALL MACHINE DEPENDENT ROUTINE TO COPY ARRAY TABLE INFO        *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 01 OCTOBER 1986, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSCPAI **************************************************)
```

```
(* BEGIN %INCLUDE RSCPAT ************************************************)
(*                                                                       *)
PROCEDURE RSCPAT ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_ARRAY_LIST;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        COPY THE SIZE AND THE LOWER BOUND OF THE ARRAY INTO THE        *)
(*        RUN-TIME SUBSCHEMA.                                            *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME                 I/O  DESCRIPTION                         *)
(*        ====                 ===  ===========                         *)
(*        INPUT_VALUE           I   INPUT VALUE OF ARBITRARY SIZE       *)
(*        OUTPUT_VALUE          O   OUTPUT VALUE OF ARBITRARY SIZE      *)
(*        SIZE_OF_VALUE         I   SIZE OF VALUE TO BE COPIED          *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        CALL MACHINE DEPENDENT ROUTINE TO COPY THE SIZE AND THE        *)
(*        LOWER BOUND OF THE ARRAY                                       *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 01 OCTOBER 1986, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSCPAT **************************************************)
```

```
(* BEGIN %INCLUDE RSCPCI ************************************************)
(*                                                                       *)
PROCEDURE RSCPCI ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_CL_INDEX;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        COPY THE POINTER INDEX TABLE INFORMATION INTO THE              *)
(*        RUN-TIME SUBSCHEMA.                                            *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME                 I/O  DESCRIPTION                         *)
(*        ====                 ===  ===========                         *)
(*        INPUT_VALUE           I   INPUT VALUE OF ARBITRARY SIZE       *)
(*        OUTPUT_VALUE          O   OUTPUT VALUE OF ARBITRARY SIZE      *)
(*        SIZE_OF_VALUE         I   SIZE OF VALUE TO BE COPIED          *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        CALL MACHINE DEPENDENT ROUTINE TO COPY CL INDEX INFO           *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 29 JANUARY 1987, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSCPCI **************************************************)
```

```
(* BEGIN %INCLUDE RSCPCT ************************************************)
(*                                                                       *)
PROCEDURE RSCPCT ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_CL_KINDS;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        COPY THE KINDS OF POINTERS INTO THE RUN-TIME SUBSCHEMA.        *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME                 I/O  DESCRIPTION                        *)
(*        ====                 ===  ===========                        *)
(*        INPUT_VALUE           I   INPUT VALUE OF ARBITRARY SIZE       *)
(*        OUTPUT_VALUE          O   OUTPUT VALUE OF ARBITRARY SIZE      *)
(*        SIZE_OF_VALUE         I   SIZE OF VALUE TO BE COPIED          *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        CALL MACHINE DEPENDENT ROUTINE TO COPY KINDS OF POINTER        *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 29 JANUARY 1987, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSCPCT **************************************************)
```

```
(* BEGIN %INCLUDE RSCPEI ************************************************)
(*                                                                       *)
PROCEDURE RSCPEI ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_ENUM_INDEX;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        COPY THE ENUMERATION INDEX TABLE INFORMATION INTO THE          *)
(*        RUN-TIME SUBSCHEMA.                                            *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME                 I/O  DESCRIPTION                        *)
(*        ====                 ===  ===========                        *)
(*        INPUT_VALUE           I   INPUT VALUE OF ARBITRARY SIZE      *)
(*        OUTPUT_VALUE          O   OUTPUT VALUE OF ARBITRARY SIZE     *)
(*        SIZE_OF_VALUE         I   SIZE OF VALUE TO BE COPIED         *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        CALL MACHINE DEPENDENT ROUTINE TO COPY ENUMERATION INFO        *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 01 OCTOBER 1986, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSCPEI **************************************************)
```

```
(* BEGIN %INCLUDE RSCPET ************************************************)
(*                                                                       *)
PROCEDURE RSCPET ( VAR   OUTPUT_VALUE  : T_DATA_VALUE;
                   CONST INPUT_VALUE   : T_ENUMERATION;
                   CONST SIZE_OF_VALUE : INTEGER);
         EXTERNAL;
(*                                                                       *)
(*    $FUNCTION:                                                         *)
(*         COPY THE ENUMERATION VALUES INTO THE RUN-TIME SUBSCHEMA.      *)
(*                                                                       *)
(*    $DESCRIPTION OF ARGUMENTS:                                         *)
(*         NAME                  I/O  DESCRIPTION                      *)
(*         ====                  ===  ===========                      *)
(*         INPUT_VALUE            I   INPUT VALUE OF ARBITRARY SIZE    *)
(*         OUTPUT_VALUE           O   OUTPUT VALUE OF ARBITRARY SIZE   *)
(*         SIZE_OF_VALUE          I   SIZE OF VALUE TO BE CO IED       *)
(*                                                                       *)
(*    $COMMONS:                                                          *)
(*                                                                       *)
(*    $ENVIRONMENT:                                                      *)
(*         LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                     *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*                                                                       *)
(*    $EXECUTION PROCEDURE:                                              *)
(*         RUN-TIME SUBSCHEMA                                            *)
(*                                                                       *)
(*    $PROCESSING DESCRIPTION:                                           *)
(*         CALL MACHINE DEPENDENT ROUTINE TO COPY ENUMERATION TABLE      *)
(*                                                                       *)
(*    $COMMENTS:                                                         *)
(*                                                                       *)
(*    $CHANGE CONTROL:                                                   *)
(*         REVISED: 16 MAY 1986, GEORGE A. WHITE, FRMI, REORGANIZED      *)
(*                  GLOBAL DECLARATIONS INTO 'NVITYP'.                   *)
(*         ORIGINATED: 15 OCTOBER 1985, G. A. WHITE, FRMI                *)
(*                                                                       *)
(* END %INCLUDE RSCPET **************************************************)
```

```
(* BEGIN %INCLUDE RSFILE ************************************************)
(*                                                                       *)
PROCEDURE RSFILE ( VAR   SUBSCHEMA_KEY    : ENTKEY;
                   VAR   RETURN_CODE      : EXT_RET_CODE );
          SUBPROGRAM;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        FILE RUN-TIME SUBSCHEMA INTO SEQUENTIAL FILE                   *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME             I/O  DESCRIPTION                             *)
(*        ====             ===  ===========                             *)
(*        RTS_RETURN_CODE  O    RETURN CODE                             *)
(*                              = 0  SUCCESS                            *)
(*                              > 0  CRITICAL ERROR:                    *)
(*                                1 KIND NOT IN RUN-TIME SUBSCHEMA      *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*        CALLED FROM THE SCHEMA EXECUTIVE                               *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        OPEN DATAFILE                                                  *)
(*        OPEN INXFILE                                                   *)
(*        LOOP THROUGH LIST OF SUBSCHEMA                                 *)
(*           GET RUN_TIME SUBSCHEMA ( RSGRSM )                           *)
(*           WRITE KIND, RECORD NO, OFFSET, RUNTIME_SIZE INTO INXFILE    *)
(*           FOR 1 TO RUNTIME_SIZE                                       *)
(*              WRITE RUN_TIME SUBSCHEMA INTO DATAFILE                   *)
(*        END LOOP                                                       *)
(*        CLOSE DATAFILE                                                 *)
(*        CLOSE INXFILE                                                  *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 27 JANUARY 1987, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RSFILE **************************************************)
```

```
(* BEGIN %INCLUDE RSGTSM ********************************************)
(*                                                                  *)
PROCEDURE RSGTSM ( CONST ENTITY_KEY      : ENTKEY;
                   VAR   RUN_TIME        : T_RUN_TIME;
                   VAR   RUN_TIME_SIZE   : INTEGER;
                   VAR   RTS_RETURN_CODE : EXT_RET_CODE );
         SUBPROGRAM;
(*                                                                  *)
(*    $FUNCTION:                                                    *)
(*         BUILD RUN-TIME SUBSCHEMA FROM SCHEMA MODEL               *)
(*                                                                  *)
(*    $DESCRIPTION OF ARGUMENTS:                                    *)
(*         NAME            I/O  DESCRIPTION                         *)
(*         ====            ===  ===========                         *)
(*         ENTITY_KEY       I   KEY FROM SCHEMA MODEL WHICH THE     *)
(*                              TRANSLATION WILL BE PERFORMED.      *)
(*         RTS_RETURN_CODE  O   RETURN CODE                         *)
(*                              = 0  SUCCESS                        *)
(*                              > 0  CRITICAL ERROR:                *)
(*                                1 KIND NOT IN RUN-TIME SUBSCHEMA  *)
(*         RUN_TIME         O   RUN-TIME SUBSCHEMA WHICH CONTAINS THE *)
(*                              ENTITY DEFINITION, ALONG WITH ANY   *)
(*                              ENUMERATION VALUES, IN A COMPACTED FORM.*)
(*         RUN_TIME_SIZE    O   THE NUMBER OF BYTES ACTUALLY REQUIRED *)
(*                              FOR THE COMPACTED RUN-TIME SUBSCHEMA. *)
(*                                                                  *)
(*    $COMMONS:                                                     *)
(*                                                                  *)
(*    $ENVIRONMENT:                                                 *)
(*         LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                  *)
(*    $EXECUTION PROCEDURE:                                         *)
(*         RUN-TIME SUBSCHEMA                                       *)
(*         CALLED FROM THE NAME/VALUE INTERFACE                     *)
(*                                                                  *)
(*    $PROCESSING DESCRIPTION:                                      *)
(*         TRANSLATE SCHEMA MODEL ENTRY INTO ENTITY ATTRIBUTES      *)
(*            AND ENUMERATION VALUES AND ARRAY INFORMATION          *)
(*         IF THERE WERE ANY ENUMERATION ATTRIBUTES THEN            *)
(*            CALCULATE THE STARTING POSITION OF THE ENUMERATION    *)
(*              INDEX TABLE AND STORE INTO RUN-TIME SUBSCHEMA       *)
(*            DETERMINE THE ACTUAL SIZE OF THE ENUMERATION INDEX TABLE *)
(*            COPY ENUMERATION INDEX TABLE INFORMATION INTO RUN-TIME *)
(*                                                    SUBSCHEMA     *)
(*         ENDIF                                                    *)
(*         IF THERE WERE ANY ENUMERATION ATTRIBUTES THEN            *)
(*            CALCULATE THE STARTING POSITION OF THE ENUMERATION    *)
(*              VALUE TABLE AND STORE INTO RUN-TIME SUBSCHEMA       *)
(*            DETERMINE THE ACTUAL SIZE OF THE ENUMERATION VALUE TABLE *)
(*            COPY THE ENUMERATION VALUES INTO THE RUN-TIME SUBSCHEMA *)
```

```
(*        ENDIF                                                          *)
(*        IF THERE WERE ANY ARRAY ATTRIBUTES THEN                        *)
(*           CALCULATE THE STARTING POSITION OF THE ARRAY INDEX TABLE     *)
(*             AND STORE INTO RUN-TIME SUBSCHEMA                         *)
(*           DETERMINE THE ACTUAL SIZE OF THE ARRAY INDEX TABLE          *)
(*           COPY ARRAY TABLE INDEX INFORMATION INTO RUN-TIME SUBSCHEM*)
(*        ENDIF                                                          *)
(*        IF THERE WERE ANY ARRAY ATTRIBUTES THEN                        *)
(*           CALCULATE THE STARTING POSITION OF THE ARRAY LIST TABLE     *)
(*             AND STORE INTO RUN-TIME SUBSCHEMA                         *)
(*           DETERMINE THE ACTUAL SIZE OF THE ARRAY LIST TABLE           *)
(*           COPY ARRAY LIST INFORMATION INTO RUN-TIME SUBSCHEMA         *)
(*        ENDIF                                                          *)
(*        CALCULATE THE SIZE OF THE RUN-TIME SUBSCHEMA                   *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 11 AUGUST 1986, M. H. CHOI, FRMI                   *)
(*                                                                       *)
(* END %INCLUDE RSGTSM ***************************************************)
```

```
(* BEGIN %INCLUDE RSMASKND ********************************************)
(*                                                                     *)
PROCEDURE RSMASKND ( VAR   ENTITY           : T_SCHEMA );
         SUBPROGRAM;
(*                                                                     *)
(*   $FUNCTION:                                                        *)
(*        INSERT THE MODEL ACCESS SOFTWARE(MAS) ATTRIBUTES ( KIND,     *)
(*        LENGTH, SYSUSE ) INTO A RUN-TIME SUBSCHEMA.                  *)
(*                                                                     *)
(*   $DESCRIPTION OF ARGUMENTS:                                        *)
(*        NAME              I/O  DESCRIPTION                           *)
(*        ====              ===  ===========                           *)
(*        ENTITY             O   RUN-TIME SUBSCHEMA ENTITY DEFINITION.*)
(*                                                                     *)
(*   $COMMONS:                                                         *)
(*                                                                     *)
(*   $ENVIRONMENT:                                                     *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                    *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                       *)
(*                                                                     *)
(*   $EXECUTION PROCEDURE:                                             *)
(*        NAME/VALUE INTERFACE                                         *)
(*        CALLED FROM THE NAME/VALUE INTERFACE                         *)
(*                                                                     *)
(*   $PROCESSING DESCRIPTION:                                          *)
(*        INSERT KIND, LENGTH, SYSUSE ATTRIBUTES INTO A RUN-TIME       *)
(*        SUBSCHEMA ENTITY DEFINITION.                                 *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)             *)
(*        ORIGINATED:  15 SEPTEMBER 1987, M. H. CHOI, DBMA             *)
(*                                                                     *)
(* END %INCLUDE RSMASKND ***********************************************)
```

```
(* BEGIN %INCLUDE RSTRGF ************************************************)
(*                                                                       *)
PROCEDURE RSTRGF ( CONST ENTITY_KEY      : ENTKEY;
                   VAR   ENTITY          : T_SCHEMA;
                   VAR   ENUM            : T_ENUM_COMPACTOR;
                   VAR   ENUM_INDEX      : T_ENUM_INX_COMPACTOR;
                   VAR   ARRAY_LIST      : T_ARRAY_LIST_COMPACTOR;
                   VAR   ARRAY_INDEX     : T_ARRAY_INX_COMPACTOR;
                   VAR   CL_INDEX        : T_CL_INX_COMPACTOR;
                   VAR   CL_LIST         : T_CL_KINDS_COMPACTOR );
         SUBPROGRAM;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*                                                                       *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME            I/O  DESCRIPTION                             *)
(*        ====            ===  ===========                             *)
(*        ENTITY_KEY       I   KEY FROM SCHEMA MODEL WHICH THE         *)
(*                             TRANSLATION WILL BE PERFORMED.          *)
(*        RTS_RETURN_CODE  O   RETURN CODE                             *)
(*                             = 0  SUCCESS                            *)
(*                             > 0  CRITICAL ERROR:                    *)
(*                               1 KIND NOT IN RUN-TIME SUBSCHEMA      *)
(*        RUN_TIME         O   RUN-TIME SUBSCHEMA WHICH CONTAINS THE   *)
(*                             ENTITY DEFINITION, ALONG WITH ANY       *)
(*                             ENUMERATION VALUES, IN A COMPACTED FORM.*)
(*        RUN_TIME_SIZE    O   THE NUMBER OF BYTES ACTUALLY REQUIRED   *)
(*                             FOR THE COMPACTED RUN-TIME SUBSCHEMA.   *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*        CALLED FROM THE NAME/VALUE INTERFACE                           *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        ORIGINATED: 20 NOVEMBER 1986, M. H. CHOI, DBMA                 *)
(*                                                                       *)
(* END %INCLUDE RSTRGF **************************************************)
```

```
(* BEGIN %INCLUDE RSTRSM ********************************************)
(*                                                                  *)
PROCEDURE RSTRSM ( CONST ENTITY_KEY       : ENTKEY;
                   VAR   ENTITY           : T_SCHEMA;
                   VAR   ENUM             : T_ENUM_COMPACTOR;
                   VAR   ENUM_INDEX       : T_ENUM_INX_COMPACTOR;
                   VAR   ARRAY_LIST       : T_ARRAY_LIST_COMPACTOR;
                   VAR   ARRAY_INDEX      : T_ARRAY_INX_COMPACTOR;
                   VAR   CL_INDEX         : T_CL_INX_COMPACTOR;
                   VAR   CL_LIST          : T_CL_KINDS_COMPACTOR;
                   VAR   RTS_RETURN_CODE  : INTEGER );
        SUBPROGRAM;
(*                                                                  *)
(*   $FUNCTION:                                                     *)
(*        TRANSLATE A SCHEMA MODEL ENTRY INTO A RUN-TIME SUBSCHEMA  *)
(*        ENTITY, ENUMERATION TABLE AND ARRAY INFO TABLE.           *)
(*                                                                  *)
(*   $DESCRIPTION OF ARGUMENTS:                                     *)
(*        NAME              I/O  DESCRIPTION                        *)
(*        ====              ===  ===========                        *)
(*        ARRAY_INDEX        O   RUN-TIME SUBSCHEMA ARRAY TABLE INDEX *)
(*                               INFORMATION.                       *)
(*        ARRAY_LIST         O   RUN-TIME SUBSCHEMA ARRAY TABLE     *)
(*                               AND COMPACTION INFORMATION.        *)
(*        ENUM               O   RUN-TIME SUBSCHEMA ENUMERATION TABLE *)
(*                               AND COMPACTION INFORMATION.        *)
(*        ENUM_INDEX         O   RUN-TIME SUBSCHEMA ENUMERATION TABLE *)
(*                               INDEX INFORMATION.                 *)
(*        ENTITY             O   RUN-TIME SUBSCHEMA ENTITY DEFINITION.*)
(*        ENTITY_KEY         I   KEY FROM SCHEMA MODEL WHICH THE    *)
(*                               TRANSLATION WILL BE PERFORMED.     *)
(*        RTS_RETURN_CODE    O   RETURN CODE                        *)
(*                               = 0 SUCCESS                        *)
(*                               > 0 CRITICAL ERROR:                *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                 *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        NAME/VALUE INTERFACE                                      *)
(*        CALLED FROM THE NAME/VALUE INTERFACE                      *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        OBTAIN ENTITY NAME AND KIND FROM SCHEMA MODEL             *)
```

```
(*        STORE ENTITY NAME AND KIND INTO RUN-TIME SUBSCHEMA          *)
(*        LOOP THROUGH SCHEMA MODEL ENTRIES                           *)
(*           OBTAIN ATTRIBUTE ENTRY FROM SCHEMA MODEL                 *)
(*           CASE DATA TYPE OF                                        *)
(*              INTEGER, REAL, STRING, LOGICAL                        *)
(*                  : APPLICATION_DATA_BLOCK_ATTRIBUTE, PROCEDURE (1) *)
(*              POINTER :   CONSTITUENT_LIST_ATTRIBUTE, PROCEDURE (2) *)
(*              ARRAY   :             ARRAY_ATTRIBUTE, PROCEDURE (3) *)
(*              DEFINED_TYPE :  DEFINED_TYPE_ATTRIBUTE, PROCEDURE (4) *)
(*              OTHERWISE :  ERROR MESSAGE = 'UNKNOWN ATTRIBUTE TYPE' *)
(*           ENDCASE                                                  *)
(*        ENDLOOP                                                     *)
%PAGE
(*    PROCEDURE (1) : APPLICATION_DATA_BLOCK_ATTRIBUTE                *)
(*        STORE ATTRIBUTE DEFINITION FOR TYPE IN SCHEMA MODEL ENTRY   *)
(*                                                                    *)
(*    PROCEDURE (2) : CONSTITUENT_LIST_ATTRIBUTE                      *)
(*        OBTAIN CONSTITUENT LIST POSITION FROM SCHEMA MODEL          *)
(*        STORE ATTRIBUTE DEFINITION FOR TYPE IN SCHEMA MODEL ENTRY   *)
(*                                                                    *)
(*    PROCEDURE (3) : ARRAY_ATTRIBUTE                                 *)
(*        DETERMINE THE NUMBER OF ARRAY DIMENSIONS                    *)
(*        STORE ARRAY INFORMATION INTO RUN-TIME SUBSCHEMA             *)
(*        STORE TABLE INDEX POSITION FOR ARRAY LIST TABLE AND THE     *)
(*           NUMBER OF DIMENSIONS INTO ARRAY INDEX TABLE              *)
(*        CALCULATE TOTAL SIZE OF THE ARRAY AND STORE INTO ARRAY      *)
(*           INDEX TABLE                                              *)
(*        FOR THE NUMBER OF ARRAY DIMENSIONS                          *)
(*           CALCULATE THE SIZE OF EACH ARRAY                         *)
(*           STORE SIZE AND LOW-BOUND INTO ARRAY LIST TABLE           *)
(*        END LOOP                                                    *)
(*                                                                    *)
(*    PROCEDURE (4) : DEFINED_TYPE_ATTRIBUTE                          *)
(*        OBTAIN DATA TYPE FOR DEFINED TYPE ATTRIBUTE IN SCHEMA MODEL *)
(*        CASE DATA_TYPE OF                                           *)
(*              INTEGER, REAL, STRING, LOGICAL                        *)
(*                  : APPLICATION_DATA_BLOCK_ATTRIBUTE, PROCEDURE (1) *)
(*              ENUMERATION :    ENUMERATION_ATTRIBUTE, PROCEDURE (5) *)
(*              POINTER :   CONSTITUENT_LIST_ATTRIBUTE, PROCEDURE (2) *)
(*              ARRAY   :             ARRAY_ATTRIBUTE, PROCEDURE (3) *)
(*              OTHERWISE :  ERROR MESSAGE = 'UNKNOWN ATTRIBUTE TYPE' *)
(*           ENDCASE                                                  *)
(*                                                                    *)
(*    PROCEDURE (5) : ENUMERATION_ATTRIBUTE                           *)
(*        STORE ATTRIBUTE DEFINITION FOR ENUMERATION TYPE             *)
(*        OBTAIN NUMBER OF ENUMERATION VALUES FROM SCHEMA MODEL       *)
(*        STORE NUMBER OF ENUMERATION VALUE IN ENUMERATION INDEX TABLE*)
(*        STORE ENUMERATION VALUE TABLE INDEX POSITION IN ENUMERATION *)
(*          INDEX TABLE                                               *)
```

```
(*        LOOP THROUGH ENUMERATION VALUES                                *)
(*          OBTAIN ENUMERATION VALUE FROM SCHEMA MODEL                   *)
(*          STORE ENUMERATION VALUE IN ENUMERATION VALUE TABLE           *)
(*        END LOOP                                                       *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)               *)
(*        ORIGINATED:  06 AUGUST 1986, M. H. CHOI, FRMI                  *)
(*                                                                       *)
(* END %INCLUDE RSTRSM ***************************************************)
```

```
(* BEGIN %INCLUDE RSTRST ************************************************)
(*                                                                       *)
PROCEDURE RSTRST ( CONST SUBTYPE_KEY     : ENTKEY;
                   VAR   ENTITY          : T_SCHEMA;
                   VAR   ENUM            : T_ENUM_COMPACTOR;
                   VAR   ENUM_INDEX      : T_ENUM_INX_COMPACTOR;
                   VAR   ARRAY_LIST      : T_ARRAY_LIST_COMPACTOR;
                   VAR   ARRAY_INDEX     : T_ARRAY_INX_COMPACTOR;
                   VAR   CL_INDEX        : T_CL_INX_COMPACTOR;
                   VAR   CL_LIST         : T_CL_KINDS_COMPACTOR;
                   VAR   IRC             : RET_REC );
         SUBPROGRAM;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        TRANSLATE SUPER TYPE INTO A RUN-TIME SUBSCHEMA                 *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME            I/O  DESCRIPTION                               *)
(*        ====            ===  ===========                               *)
(*        SUBTYPE_KEY      I   KEY FROM SCHEMA MODEL WHICH THE           *)
(*                             TRANSLATION WILL BE PERFORMED.            *)
(*                             > 0  CRITICAL ERROR:                      *)
(*        ENTITY           O   RUN-TIME SUBSCHEMA WHICH CONTAINS THE     *)
(*                             ENTITY DEFINITION                         *)
(*        ENUM             O   ENUMERATION VALUES                        *)
(*        ENUM_INDEX       O   ENUMERATION INDEX TABLE                   *)
(*        ARRAY_LIST       O   LOWER BOUND AND ARRAY SIZE                *)
(*        ARRAY_INDEX      O   ARRAY INDEX TABLE                         *)
(*        CL_INDEX         O   CONSTITUENT INDEX TABLE                   *)
(*        CL_LIST          O   CONSTITUENT KINDS                         *)
(*        IRC              O   RETURN CODE                               *)
(*                             = 0  SUCCESS                              *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        RUN-TIME SUBSCHEMA                                             *)
(*        CALLED FROM THE NAME/VALUE INTERFACE                           *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        MAKE A LIST OF SUPER TYPE                                      *)
(*        LOOP THROUGH A LIST OF SUPER TYPE                              *)
(*           TRANSLATE SUPER TYPE INTO A RUN-TIME SUBSCHEMA              *)
(*                                                                       *)
```

```
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*        ORIGINATED: 9 SEPTEMBER 1987, M. H. CHOI, DBMA           *)
(*                                                                 *)
(* END %INCLUDE RSTRST ***********************************************)
```

```
(* BEGIN %INCLUDE RS1100 ************************************************)
(*                                                                       *)
PROCEDURE RS1100 ( CONST SUBSCHEMA_KEY    : ENTKEY;
                   VAR   RUN_TIME         : T_RUN_TIME;
                   VAR   RUN_TIME_SIZE    : INTEGER;
                   VAR   IRC              : RET_REC );
         SUBPROGRAM;
(*                                                                       *)
(*   $FUNCTION:                                                          *)
(*        STORE ARRAY_ENTITY(1100) IN THE DATA DICTIONARY AND            *)
(*        THE RUN-TIME SUBSCHEMA                                         *)
(*                                                                       *)
(*   $DESCRIPTION OF ARGUMENTS:                                          *)
(*        NAME              I/O  DESCRIPTION                             *)
(*        ====              ===  ===========                             *)
(*        RUN_TIME           O   RUN-TIME SUBSCHEMA WHICH CONTAINS       *)
(*                               THE ENTITY DEFINITION, ALONG WITH       *)
(*                               ANY ENUMERATION VALUES, CONSTITUENT     *)
(*                               LIST, AND ARRAY INFORMATION, IN A       *)
(*                               COMPACTED FORM.                         *)
(*        RUN_TIME_SIZE      O   THE NUMBER OF BYTES ACTUALLY REQUIRED*)
(*                               FOR THE COMPACTED RUN-TIME SUBSCHEMA.*)
(*        IRC                O   RETURN CODE                             *)
(*                               = 0 SUCCESS                             *)
(*                               > 0 CRITICAL ERROR:                     *)
(*                                                                       *)
(*   $COMMONS:                                                           *)
(*                                                                       *)
(*   $ENVIRONMENT:                                                       *)
(*        LANGUAGE: IBM PASCAL (SEGMENT SUBPROGRAM)                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                       *)
(*   $EXECUTION PROCEDURE:                                               *)
(*        NAME/VALUE INTERFACE                                           *)
(*        CALLED FROM THE NAME/VALUE INTERFACE                           *)
(*                                                                       *)
(*   $PROCESSING DESCRIPTION:                                            *)
(*        OBTAIN ENTITY NAME AND KIND FROM SCHEMA MODEL                  *)
(*        STORE ENTITY NAME AND KIND INTO RUN-TIME SUBSCHEMA             *)
(*        STORE GLOBAL ATTRIBUTES                                        *)
(*        STORE CL_ENTITIES ATTRIBUTE                                    *)
(*                                                                       *)
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*        REVISED: (DATE, NAME, GROUP, REASON/DESCRIPTION)               *)
(*        ORIGINATED:  20 AUGUST 1987, M. H. CHOI, DBMA                  *)
(*                                                                       *)
(* END %INCLUDE RS1100 **************************************************)
```

```
(* %INCLUDE SCALFSRT *)
(**)
  PROCEDURE SCALFSRT(CONST CURRENT : ENTBLOCK;
                     CONST NEXT    : ENTBLOCK;
                     VAR   FLIP    : BOOLEAN;
                     VAR   RRC     : EXT_RET_CODE;
                     VAR   PROC    : ROUTINE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*         THIS ROUTINE IS THE ORDER FUNCTION CALLED BY MALSRT             *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME           I/O  DESCRIPTION                                 *)
(*         ====           ===  ===========                                 *)
(*         CURRENT         I   THE ADB OF THE CURRENT ENTITY               *)
(*         NEXT            I   THE ADB OF THE NEXT ENTITY                  *)
(*         FLIP            O   INDICATES IF THE ENTITIES SHOULD BE         *)
(*                             FLIPPED                                     *)
(*         RRC             O   THE ROUTINE'S RETURN CODE                   *)
(*                             = 0 OK                                      *)
(*                             <> 0 ERROR                                  *)
(*         XRC             O   EXTERNAL RETURN CODE FROM MAS               *)
(*                             = 0  OK                                     *)
(*                             >= 10  ERROR                                *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*       NONE                                                              *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*         THIS ROUTINE IS CALLED BY THE MAS ROUTINE MALSRT.  THE          *)
(*         TWO ENTITIES ARE COMPARED AND IF THEY ARE OUT OF ALPHA-         *)
(*         BETICAL ORDER THE FLIP FLAG IS SET TO TRUE OTHERWISE THE        *)
(*         FLAG REMAINS FALSE.  IF THE FLAG IS TRUE THE ENTITIES ARE       *)
(*         SWAPPED OTHERWISE THEY ARE NOT.                                 *)
(*                                                                         *)
(*    $COMMENTS:                                                           *)
(*                                                                         *)
(*    $CHANGE CONTROL:                                                     *)
(*                                                                         *)
```

```
(* %INCLUDE SCARYCR *)
(**)
  PROCEDURE SCARYCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE     *)
(*         ARRAY ENTITY AND PUSHES THE DATA ON THE TRANSACTION       *)
(*         STACK.                                                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   THE RETURN CODE                      *)
(*         TRANS_STACK    I/O  THE TRANSACTION STACK                *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION*)
(*                             IS CHOSEN WITHIN ANOTHER ROUTINE      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRARRAY     *)
(*         TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS   *)
(*         VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION  *)
(*         STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE  *)
(*         DATA ENTERED IS INVALID THEN THE PANEL IS REDISPLAYED     *)
(*         WITH AN ERROR MESSAGE.                                    *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCARYUP *)
(**)
  PROCEDURE SCARYUP(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR;
                    VAR ARRAY_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS : LISTKEY;
                    VAR DELETE_LIST : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE AN ARRAY.          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*     IRC               O    RETURN CODE                           *)
(*     TRANS_STACK       I/O  POINTS TO THE TRANSACTION STACK       *)
(*     ARRAY_KEY         I/O  KEY OF THE ARRAY TO BE UPDATED        *)
(*     NUMBER_OF_USERS   I/O  THE NUMBER OF USERS OF THE ARRAY      *)
(*     DELETE_LIST       I/O  LIST OF ENTITIES TO BE DELETED        *)
(*     NEW_KEYS_LIST     I/O  LIST OF NEWLY CREATED ENTITIES        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE CURRENT ARRAY DATA IS DISPLAYED ON THE UPARRAY PANEL.  *)
(*        THE DATA CAN THEN BE UPDATED BY THE USER.                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCBASIN *)
(**)
  PROCEDURE SCBASIN(VAR IRC        : RET_REC;
                    VAR INCLD      : TEXT;
                    VAR ENTITY_KEY : ENTKEY;
                    VAR ADB        : ENTITY_ADB);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES THE ENTITY KIND CONSTANTS TO THE       *)
(*        PASCAL INCLUDE FILE.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC            I/O  RETURN CODE                           *)
(*        INCLD           I   THE FILE NAME                         *)
(*        ENTITY_LIST     I   A LIST OF ENTITIES                    *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        GET THE ADB OF THE PRIMITIVE ENTITY                        *)
(*        CASE PRIMITIVE ENTITY KIND OF                              *)
(*          INTEGER     :                                            *)
(*            CASE PRECISION IN DECIMAL DIGITS OF                    *)
(*              1, 2      : WRITE TO THE FILE PACKED 0..255;         *)
(*              3, 4      : WRITE TO THE FILE PACKED 0..65535;       *)
(*              5, 6, 7,                                             *)
(*                 8, 9   : WRITE TO THE FILE INTEGER;               *)
(*            END;                                                   *)
(*          REAL          :                                          *)
(*            CASE PRECISION IN DECIMAL DIGITS OF                    *)
(*              1, 2, 3, 4,                                          *)
(*                 5, 6, 7  : WRITE TO THE FILE SHORTREAL;           *)
(*               8,  9, 10,                                          *)
(*              11, 12, 13,                                          *)
(*              14, 15, 16  : WRITE TO THE FILE REAL;                *)
(*            END;                                                   *)
```

```
(*          STRING       :                                              *)
(*            WRITE TO THE FILE PACKED ARRAY(.1..<STRING LENGTH>.)     *)
(*                                                        OF CHAR;     *)
(*          DEFINED TYPE :                                             *)
(*            WRITE TO THE FILE T_<DEFINED TYPE NAME>;                 *)
(*          LOGICAL      :                                             *)
(*            WRITE TO THE FILE BOOLEAN;                               *)
(*          ENUMERATION  :                                             *)
(*            WRITE TO THE FILE (                                      *)
(*            SET THE ENUMERATION'S CONSTITUENT LIST TO READ FORWARD   *)
(*            COUNT THE NUMBER OF CONSTITUENTS                         *)
(*            FOR INDEX EQUALS ONE TO THE NUMBER OF CONSTITUENTS DO    *)
(*              GET THE KEY TO THE NEXT CONSTITUENT IN THE LIST        *)
(*              GET THE CONSTITUENT'S ADB                              *)
(*              WRITE TO THE FILE <ENUMERITEM NAME>                    *)
(*              IF THE COUNT IS EQUAL TO THE NUMBER OF CONSTITUENTS    *)
(*                WRITE TO THE FILE );                                 *)
(*              ELSE                                                   *)
(*                WRITE TO THE FILE ,                                  *)
(*            END;                                                     *)
(*          ARRAY        :                                             *)
(*            GET THE ADB OF THE ARRAY ENTITY                          *)
(*            WRITE TO THE FILE ARRAY(.<LOW BOUND>..<HIGH BOUND>.) OF  *)
(*            SET THE ARRAY ENTITY'S CONSTITUENT LIST TO READ FORWARD  *)
(*            GET THE KEY TO THE ARRAY ENTITY'S FIRST CONSTITUENT      *)
(*            GET THE CONSTITUENT'S ADB                                *)
(*            CALL THIS ROUTINE TO WRITE OUT THE ARRAY'S TYPE          *)
(*              INTEGER                                                *)
(*              REAL                                                   *)
(*              STRING                                                 *)
(*              LOGICAL                                                *)
(*              ARRAY                                                  *)
(*              DEFINED TYPE                                           *)
(*              POINTER                                                *)
(*            END;                                                     *)
(*          STRUCTURE    :                                             *)
(*            WRITE TO THE FILE RECORD                                 *)
(*            SET THE STRUCTURE'S CONSTITUENT LIST TO BE READ FORWARD  *)
(*            CONTINUE TO READ CONSTITUENTS UNTIL THE END OF LIST      *)
(*            GET THE KEY TO THE NEXT CONSTITUENT ENTITY ON THE LIST   *)
(*            GET THE CONSTITUENT'S ADB                                *)
(*            WRITE TO THE FILE <FIELD NAME> :                         *)
(*            GET THE KEY TO THE CONSTITUENT'S FIRST CONSTITUENT       *)
(*            GET THIS CONSTITUENT'S ADB                               *)
(*            CALL THIS ROUTINE TO WRITE OUT THE PRIMITIVE TYPE        *)
(*              INTEGER                                                *)
(*              REAL                                                   *)
(*              STRING                                                 *)
(*              LOGICAL                                                *)
```

```
(*              ARRAY                                                   *)
(*              DEFINED TYPE                                            *)
(*              POINTER (IS NOT ALLOWED WITHIN A STRUCTURE)             *)
(*            IF THE END OF LIST IS FOUND THEN                          *)
(*              WRITE TO THE FILE END;                                  *)
(*            END;                                                      *)
(*      END;                                                            *)
(*                                                                      *)
(*    $COMMENTS:                                                        *)
(*                                                                      *)
(*    $CHANGE CONTROL:                                                  *)
(*                                                                      *)
```

```
(* %INCLUDE SCCHRCK *)
(**)
  PROCEDURE SCCHRCK(VAR IRC     : RET_REC;
                    VAR NAME    : T_NAME;
                    VAR INVALID : BOOLEAN);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS CHECKS THAT THE CHARACTERS IN AN ENTITY NAME ARE VALID *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                          *)
(*         ====          ===  ===========                          *)
(*         IRC           I/O  RETURN CODE                          *)
(*         NAME           I   NAME TO CHECK FOR VALID CHARACTERS    *)
(*         INVALID        O   INDICATES IF THE NAME CONTAINS AN IN- *)
(*                            VALID CHARACTER OR A SPACE INBETWEEN   *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CHECKS THAT THE NAME CONTAINS VALID CHARAC-  *)
(*         TERS.  VALID CHARACTERS ARE THE LETTERS A THRU Z, THE     *)
(*         DIGITS 0 THRU 9, BLANKS AND UNDERSCORES ARE ALLOWED.  ANY *)
(*         OTHER CHARACTERS ARE INVALID.  THE NAME IS ALSO CHECKED   *)
(*         FOR SPACES INBETWEEN CHARACTERS.  THIS TOO IS INVALID.    *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCCLSCR1 *)
(**)
  PROCEDURE SCCLSCR1(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE NAME AND KIND NUMBER TO BE        *)
(*        ASSIGNED TO THE CLASS ENTITY TO BE CREATED.                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   RETURN CODE                           *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        REF                                                        *)
(*           INSIDE      I/O  INDICATES IF THE EXIT OPTION OR RETURN*)
(*                            HAS BEEN CHOSEN WITHIN ANOTHER CREATE  *)
(*                            PROCEDURE                              *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (CRCLASS1)   *)
(*        WHICH PUTS UP THE MENU TO GATHER THE NAME AND KIND NUMBER  *)
(*        FOR THE CLASS ENTITY.  THE DATA IS VERIFIED FOR UNIQUENESS.*)
(*        IF THE INFORMATION ENTERED IS INDEED UNIQUE THEN IT IS     *)
(*        PUSHED ONTO THE TRANSACTION STACK. THEN (SCCLSCR2) IS      *)
(*        CALLED TO GATHER THE CONSTITUENTS. WHEN ALL THE REQUIRED   *)
(*        DATA HAS BEEN ENTERED TO CREATE A CLASS ENTITY THE         *)
(*        TRANSACTION PROCESSOR ROUTINE (SCTRSPR) IS CALLED TO       *)
(*        PROCESS THE TRANSACTIONS.                                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCCLSCR2 *)
(**)
  PROCEDURE SCCLSCR2(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE CONSTITUENTS OF THE CLASS ENTITY.  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        IRC            O   RETURN CODE                            *)
(*        TRANS_STACK   I/O  POINTS TO THE TRANSACTION STACK        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        REF                                                        *)
(*           INSIDE     I/O  INDICATES IF THE EXIT OPTION OR RETURN  *)
(*                           HAS BEEN CHOSEN WITHIN ANOTHER CREATE   *)
(*                           PROCEDURE                               *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (CRCLASS2)   *)
(*        WHICH PUTS UP THE MENU TO GATHER THE CONSTITUENTS.  THE    *)
(*        CONSTITUENTS ARE PUSHED ON THE STACK AFTER VERIFYING THAT  *)
(*        THEY DO EXIST.                                             *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCCLSUP *)
(**)
  PROCEDURE SCCLSUP(VAR IRC       : RET_REC;
                    VAR CLASS_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE CLASS ENTITY.  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        CLASS_KEY      I   KEY OF THE ENTITY TO BE UPDATED       *)
(*        IRC            O   RETURN CODE                           *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*        NONE                                                      *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINES WHICH      *)
(*        DISPLAY THE DATA DESCRIBING THE CLASS ENTITY.  THE UPDATE *)
(*        CLASS OPTIONS INCLUDE : CHANGE THE NAME AND/OR KIND NUMBER,*)
(*                                UPDATE THE CONSTITUENT LIST BY    *)
(*                                     ADDING OR REMOVING ELEMENTS, *)
(*                                REVIEW A CONSTITUENT,             *)
(*                                DELETE THE CLASS ENTITY,          *)
(*                                SAVE THE CHANGES,                 *)
(*                                RETURN AND EXIT.                  *)
(*        THE CHANGES MADE TO THE CLASS ENTITY ARE KEPT ONLY IF THE *)
(*        OPTION SAVE THE CHANGES IS SELECTED.  OTHERWISE ANY CHANGES*)
(*        MADE ARE IGNORED.                                         *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
```

```
(* %INCLUDE SCCOMPAR *)
(**)
  PROCEDURE SCCOMPAR(VAR  FIRST_NAME  :  T_NAME;
                     VAR  SECOND_NAME :  T_NAME;
                     VAR  DIFFERENT   :  BOOLEAN);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE COMPARES TWO NAMES FOR THE LENGTH SPECIFIED   *)
(*         BY 'UNIQUENESS_LENGTH' IN SCECON.  THE VARIABLE 'DIFFERENT' *)
(*         IS SET TO TRUE IF THE TWO NAMES DIFFER.                   *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                           *)
(*         ====           ===  ===========                           *)
(*         FIRST_NAME      I   THE FIRST NAME                        *)
(*         SECOND_NAME     I   THE SECOND NAME                       *)
(*         DIFFERENT       O   SET TO TRUE IF THE TWO NAMES DIFFER   *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         INITIALIZE VARIABLES                                      *)
(*         COMPARE FIRST_NAME TO SECOND_NAME CHARACTER-BY-CHARACTER  *)
(*           UNTIL LENGTH IS SURPASSED OR A DIFFERENCE IS DETECTED.  *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*      NARATION ON THE NEXT LINE.                                   *)
(*                                                                   *)
```

```
(*    REVISED: MM/DD/YY CCXX     I. M. APERSON                GROUP_ID *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                                *)
(*                                                                      *)
(*    ORIGINATED: 06/12/87       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE SCCOMPAR *)
```

```
(* %INCLUDE SCCONIN *)
(**)
  PROCEDURE SCCONIN(VAR IRC         : RET_REC;
                    VAR INCLD       : TEXT;
                    VAR ENTITY_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE WRITES THE ENTITY KIND CONSTANTS TO THE      *)
(*         PASCAL INCLUDE FILE.                                      *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                           *)
(*         ====           ===  ===========                           *)
(*         IRC            I/O  RETURN CODE                           *)
(*         INCLD           I   THE FILE NAME                         *)
(*         ENTITY_LIST     I   A LIST OF ENTITIES                    *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         WRITE THE ENTITY CONSTANT HEADING TO THE FILE             *)
(*         WRITE 'CONST' TO THE FILE                                 *)
(*         SET THE LIST OF ENTITIES TO BE READ FORWARD               *)
(*         CONTINUE TO READ THE ENTITIES UNTIL THE END OF THE LIST   *)
(*           GET THE KEY TO THE NEXT ENTITY IN THE LIST              *)
(*           GET THIS ENTITY'S ADB                                   *)
(*           IF THE ENTITY KIND IS AN ENTITY THEN                    *)
(*             WRITE TO THE FIELD K_<ADB.ENT.NAME> = <ADB.ENT.KIND> ;*)
(*         END;                                                      *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCCREATE *)
(**)
  PROCEDURE SCCREATE(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE *)
(*         CREATE OPTION CHOSEN.                                     *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         IRC            O   RETURN CODE                           *)
(*         TRANS_STACK   I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         DEF                                                       *)
(*            INSIDE     I/O  INDICATES IF THE EXIT OPTION HAS      *)
(*                            BEEN CHOSEN WITHIN ANOTHER CREATE      *)
(*                            PROCEDURE                              *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         CALLS THE CREATE MENU INTERFACE ROUTINE (MCREATE) AND     *)
(*         PROCESSES THE DATA RECIEVED FROM THE MENU EITHER BY       *)
(*         CALLING THE APPROPRIATE ROUTINE OR EXITING THE PROCEDURE. *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*      NARATION ON THE NEXT LINE.                                   *)
(*                                                                   *)
```

```
(*    REVISED: 09/28/87          C. H. MOHME                    DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                              *)
(*                                                                        *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                    DBMA     *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCCREATE *)
```

```
(* %INCLUDE SCDEFCR *)
(**)
  PROCEDURE SCDEFCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE     *)
(*         DEFINED TYPE ENTITY AND PUSHES THE DATA ON THE TRANS-     *)
(*         ACTION STACK.                                             *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                            *)
(*         ====          ===  ===========                            *)
(*         IRC            O   THE RETURN CODE                        *)
(*         TRANS_STACK   I/O  THE TRANSACTION STACK                  *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         INSIDE        I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE       *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRDEFTYP     *)
(*         TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS   *)
(*         VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION  *)
(*         STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE  *)
(*         DATA ENTERED IS INVALID THEN THE PANEL IS REDISPLAYED     *)
(*         WITH AN ERROR MESSAGE.                                    *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCDEFUP *)
(*)
  PROCEDURE SCDEFUP(VAR IRC              : RET_REC;
                    VAR TRANS_STACK      : TRANSPTR;
                    VAR DEFINED_TYPE_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS  : INTEGER;
                    VAR DELETE_LIST      : LISTKEY;
                    VAR NEW_KEYS_LIST    : LISTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE DEFINED TYPE        *)
(*        ENTITY.                                                         *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME               I/O  DESCRIPTION                             *)
(*        ====               ===  ===========                             *)
(*        DEFINED_TYPE_KEY   I/O  KEY OF THE ENTITY TO BE UPDATED         *)
(*        DELETE_LIST        I/O LIST OF ENTITIES TO BE DELETED           *)
(*        NEW_KEYS_LIST      I/O LIST OF ENTITIES TO BE DELETED IF        *)
(*                               THE CHANGES MADE ARE REJECTED            *)
(*        NUMBER_OF_USERS     I  INDICATES THE NUMBER OF USERS            *)
(*        TRANS_STACK        I/O POINTS TO THE TRANSACTION STACK          *)
(*        IRC                 O  RETURN CODE                              *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (UPDEFTYP)        *)
(*        TO DISPLAY THE INFORMATION ABOUT THE DEFINED TYPE ENTITY.       *)
(*        THE USER CAN CHANGE THE NAME OR TYPE OF THE ENTITY OR           *)
(*        SELECT ONE OF THE FOLLOWING OPTIONS ON THE PANEL :              *)
(*             REVIEW THE CURRENT TYPE,                                   *)
(*             UPDATE THE CURRENT TYPE,                                   *)
(*             SAVE THE CHANGES MADE,                                     *)
(*             RETURN OR EXIT.                                            *)
```

```
(*        IF SAVE THE CHANGES IS SELECTED AND THE NUMBER OF USERS OF  *)
(*        THE DEFINED TYPE IS ONE THEN THE OLD DEFINED TYPE KEY IS    *)
(*        PLACED ON THE DELETE LIST AND A NEW DEFINED TYPE ENTITY IS  *)
(*        CREATED.  THE NEW DEFINED TYPE KEY IS THEN PLACED ON THE    *)
(*        NEW KEYS LIST.  IF SAVE THE CHANGES IS SELECTED AND THE     *)
(*        NUMBER OF USERS OF THE DEFINED TYPE IS GREATER THAN ONE     *)
(*        THEN THE DEFINED TYPE ENTITY IS UPDATED.                    *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
```

```
(* %INCLUDE SCENMUP *)
(**)
  PROCEDURE SCENMUP(VAR IRC           : RET_REC;
                    VAR TRANS_STACK   : TRANSPTR;
                    VAR ENUM_KEY      : ENTKEY;
                    VAR DELETE_LIST   : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE ENUMERATION     *)
(*        ENTITY.                                                     *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        TRANS_STACK   I/O  POINTS TO THE TRANSACTION STACK        *)
(*        ENUM_KEY      I/O  KEY OF THE ENTITY TO BE UPDATED        *)
(*        DELETE_LIST   I/O  LIST OF ENTITIES TO BE DELETED         *)
(*        NEW_KEYS_LIST I/O  LIST OF NEWLY CREATED ENTITIES         *)
(*        IRC            O   RETURN CODE                            *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                 *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (UPENUM)      *)
(*        TO DISPLAY DATA ABOUT THE ENUMERATION ENTITY.               *)
(*        THE UPDATE OPTIONS INCLUDE THE FOLLOWING :                  *)
(*            ADD AN ITEM TO THE ENUMERATION,                         *)
(*            REMOVE AN ITEM FROM THE ENUMERATION,                    *)
(*            SAVE THE CHANGES MADE,                                  *)
(*            RETURN AND EXIT.                                        *)
(*        THE CHANGES MADE TO THE ENUMERATION ARE KEPT ONLY IF THE    *)
(*        OPTION SAVE THE CHANGES IS SELECTED, OTHERWISE ANY CHANGES  *)
(*        THAT WERE MADE ARE IGNORED.                                 *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE SCENTCR *)
(**)
  PROCEDURE SCENTCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO MODEL THE       *)
(*        ENTITY ENTITY.                                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   RETURN CODE                           *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE.      *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (CRENTITY)   *)
(*        WHICH DISPLAYS THE CREATE ENTITY PANEL.  THE NAME AND      *)
(*        KIND NUMBER IS CHECKED FOR UNIQUENESS.  IF THEY ARE UNIQUE *)
(*        THEN THE DATA IS PUSHED ONTO THE TRANSACTION STACK AND     *)
(*        THE ROUTINE (SCFLDCR) IS CALLED TO ENTER THE ENTITY'S      *)
(*        FIELDS.  AFTER ALL OF THE FIELDS HAVE BEEN ENTERED THE     *)
(*        TRANSACTION PROCESSING ROUTINE IS CALLED TO MODEL THE      *)
(*        ENTITY.                                                    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCENTIN *)
(**)
  PROCEDURE SCENTIN(VAR IRC         : RET_REC;
                    VAR INCLD       : TEXT;
                    VAR ENTITY_LIST : LISTKEY;
                    VAR ADB_DEFN    : LISTKEY;
                    VAR CL_DEFN     : LISTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES THE ENTITY TYPE DECLARATIONS TO THE         *)
(*        PASCAL INCLUDE FILE.                                            *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC            I/O  RETURN CODE                                 *)
(*        INCLD           I   THE FILE NAME                               *)
(*        ENTITY_LIST     I   A LIST OF ENTITIES                          *)
(*        ADB_DEFN        O   A LIST OF ENTITIES ADB DEFINITIONS WERE     *)
(*                            GENERATED FOR                               *)
(*        CL_DEFN         O   A LIST OF ENTITIES CONSTITUENT DEFINI-      *)
(*                            TIONS WERE GENERATED FOR                    *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        WRITE TO THE FILE THE ENTITY DECLARATIONS HEADING               *)
(*        WRITE TYPE TO THE FILE                                          *)
(*        SET THE LIST OF ENTITIES TO BE READ FORWARD                     *)
(*        CONTINUE TO READ ENTITIES UNTIL THE END OF LIST IS FOUND        *)
(*          GET THE KEY TO THE NEXT ENTITY ON THE LIST                    *)
(*          GET THE ENTITY'S ADB                                          *)
(*          CALL THE PROCEDURE TO SORT THE FIELDS                         *)
(*          COUNT THE NUMBER OF ITEMS IN THE LIST OF ADB FIELDS           *)
(*          IF THE NUMBER OF ITEMS ON THE LIST IS GREATER THAN 0 THEN     *)
(*            WRITE TO THE FILE E_<ENTITY NAME> = RECORD                  *)
(*            CALL THE FIELD INCLUDE PROCEDURE                            *)
(*            WRITE TO THE FILE END;                                      *)
(*            ADD THE ENTITY KEY TO THE LIST OF ENTITY ADB DEFINITIONS    *)
(*          COUNT THE NUMBER OF ITEMS ON THE CONSTITUENT FIELD LIST       *)
```

```
(*          IF THE NUMBER OF ITEMS ON THE LIST IS GREATER THAN 0 THEN *)
(*            WRITE TO THE FILE C_<ENTITY NAME> = RECORD              *)
(*            CALL THE FIELD INCLUDE PROCEDURE                        *)
(*            WRITE TO THE FILE END;                                  *)
(*            WRITE TO THE FILE P_T_<ENTITY NAME> = RECORD            *)
(*            CALL THE FIELD INCLUDE PROCEDURE                        *)
(*            WRITE TO THE FILE END;                                  *)
(*            WRITE TO THE FILE CONST                                 *)
(*            WRITE TO THE FILE P_<ENTITY NAME> = P_T_<ENTITY NAME>(  *)
(*            FOR INDEX EQUALS ONE TO NUMBER OF CONSTITUENT FIELDS DO *)
(*              WRITE TO THE FILE INDEX                               *)
(*              IF INDEX EQUALS THE NUMBER OF CONSTITUENT FIELDS THEN *)
(*                WRITE TO THE FILE );                                *)
(*              ELSE                                                  *)
(*                WRITE TO THE FILE ,                                 *)
(*            ADD THE KEY TO THE LIST OF GENERATED CONSTITUENT        *)
(*                                                    DEFINITIONS     *)
(*        END;                                                        *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
(*     REVISED: 07/30/87          C. H. MOHME                 DBMA    *)
(*       CHANGED NAMING CONVENTION FROM P_ TO C_ FOR THE CONSTANT FOR *)
(*       CONSTITUENT LIST POSITION AND C_ TO P_ FOR ENTITY TYPE       *)
(*       DECLARATIONS FOR THE CONSTITUENT REFERENCED BY THE CONSTI-   *)
(*       TUENT KEY.                                                   *)
(*                                                                    *)
(*     REVISED: 07/30/87          C. H. MOHME                 DBMA    *)
(*      CHANGED NAMING CONVENTION FROM P_T_ TO C_T_ FOR THE ENTITY    *)
(*      TYPE DECLARATION FOR THE CONSTITUENT REFERENCED BY THE CON-   *)
(*      STITUENT LIST POSITION.                                       *)
(*                                                                    *)
(*     REVISED: 07/22/87          C. H. MOHME                 DBMA    *)
(*       CHANGED NAMING CONVENTION FROM C_ TO K_ FOR ENTITY KIND      *)
(*       CONSTANTS AND FROM K_ TO C_ FOR ENTITY TYPE DECLARATIONS     *)
(*       FOR THE CONSTITUENTS REFERENCED BY THE CONSTITUENT KEY.      *)
(*                                                                    *)
(*     ORIGINATED: 10/23/86       L. J. BEHAN                 DBMA    *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE SCENTIN *)
```

```
(* %INCLUDE SCENTUP *)
(**)
  PROCEDURE SCENTUP(VAR IRC         : RET_REC;
                    VAR TRANS_STACK : TRANSPTR;
                    VAR ENT_KEY     : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE UPDATES THE ENTITY ENTITY.                   *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK      *)
(*         ENT_KEY        I/O  KEY OF THE ENTITY TO BE UPDATED      *)
(*         IRC             O   RETURN CODE                          *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINES TO DISPLAY  *)
(*         THE DATA ABOUT THE ENTITY.  THE UPDATE ENTITY OPTIONS IN-  *)
(*         CLUDE THE FOLLOWING :                                     *)
(*             CHANGE THE ENTITY NAME OR KIND NUMBER,                *)
(*             UPDATE THE ENTITY'S CONSTITUENTS (FIELDS) BY          *)
(*                ADDING FIELDS, REMOVING FIELDS OR UPDATING FIELDS, *)
(*             REVIEW A FIELD,                                       *)
(*             DELETE THE ENTITY,                                    *)
(*             SAVE THE CHANGES MADE,                                *)
(*             RETURN OR EXIT.                                       *)
(*          IF SAVE THE CHANGES WAS SELECTED THE ENTITY IS UPDATED   *)
(*          AND THE ENTITIES ON THE DELETE LIST WILL BE DELETED IF   *)
(*          POSSIBLE.  IF RETURN OR EXIT IS SELECTED THE ENTITIES ON *)
(*          THE NEW KEYS LIST WILL BE DELETED IF POSSIBLE.           *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCENUCR *)
(**)
  PROCEDURE SCENUCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE      *)
(*        ENUMERITEM ENTITY AND PUSHES THE DATA ON THE TRANSACTION   *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*        TRANS_STACK    I/O  THE TRANSACTION STACK                 *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRENUM       *)
(*        TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS    *)
(*        VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION   *)
(*        STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE   *)
(*        DATA ENTERED IS INVALID THEN THE PANEL IS REDISPLAYED WITH *)
(*        AN ERROR MESSAGE.                                          *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCEXIST *)
(**)
  PROCEDURE SCEXIST(VAR IRC         : RET_REC;
                    VAR KIND_NUMBER : INTEGER;
                    VAR PRESENT     : BOOLEAN;
                    VAR KEY_OF_ENT  : ENTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE VERIFIES THE EXISTENCE OF AN ENTITY AND            *)
(*        RETURNS THE ENTITY KEY IF IT DOES IN FACT EXIST.                *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        KIND_NUMBER     I   THE USER DEFINED KIND NUMBER OF THE         *)
(*                            ENTITY TO BE FOUND                          *)
(*        PRESENT         O   INDICATES IF THE ENTITY EXISTS              *)
(*        KEY_OF_ENT      O   THE KEY OF THE ENTITY FOUND                 *)
(*        IRC             O   RETURN CODE                                 *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THIS ROUTINE IS CALLED BY A MAS EXECUTE PROCEDURE.  THE         *)
(*        SCHEMA MODEL IS SEARCHED FOR A CLASS OR ENTITY WITH A           *)
(*        USER DEFINED KIND NUMBER IDENTICAL TO THE ONE PASSED IN.        *)
(*        IF A MATCH IS FOUND THEN THE KEY TO THE ENTITY IS RETURNED      *)
(*        TO THE CALLING PROCEDURE AND A FLAG IS SET TO INDICATE THAT     *)
(*        A MATCH WAS INDEED FOUND.  IF A MATCH IS NOT FOUND THEN         *)
(*        THE FLAG IS SET TO FALSE.                                       *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE SCFLDAD *)
(**)
  PROCEDURE SCFLDAD(VAR IRC           : RET_REC;
                    VAR TRANS_STACK   : TRANSPTR;
                    VAR LIST_OF_CNSTS : LISTKEY;
                    VAR FIELDTYP      : T_FIELDTYPE;
                    VAR FIELD_KEY     : ENTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE GATHERS THE DATA TO ADD A FIELD TO AN ENTITY.       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK              *)
(*        LIST_OF_CNSTS  I/O  LIST OF CONSTITUENTS OF THE ENTITY           *)
(*        FIELDTYP        I   INDICATES THE TYPE OF FIELD                  *)
(*        FIELD_KEY       O   KEY TO THE FIELD CREATED                     *)
(*        IRC             O   RETURN CODE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THIS ROUTINE ALLOWS A USER TO ADD A FIELD TO AN ENTITY           *)
(*        DURING AN UPDATE.  THE ADD FIELD (ADDFIELD) ROUTINE IS           *)
(*        CALLED AND THE DATA IS GATHERED.  THEN THE TRANSACTION           *)
(*        PROCESSING ROUTINE (SCTRSPR) IS CALLED TO MODEL THE FIELD        *)
(*        ENTITY.                                                          *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCFLDCR *)
(**)
  PROCEDURE SCFLDCR(VAR IRC : RET_REC;
                    VAR FIELDTYP : T_FIELDTYPE;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO MODEL THE       *)
(*        FIELD ENTITY.                                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        FIELDTYP        I   INDICATES THE TYPE OF FIELD           *)
(*        IRC             O   RETURN CODE                           *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        REF                                                        *)
(*          INSIDE       I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (CRFIELD)    *)
(*        TO DISPLAY THE CREATE FIELD PANEL.  THE DATA ENTERED IS    *)
(*        VERIFIED AND IF IT IS VALID IT IS PUSHED ON TO THE TRANS-  *)
(*        ACTION STACK AND THE APPROPRIATE FIELD TYPE ROUTINE IS     *)
(*        CALLED.  IF THE DATA IS INVALID THE PANEL IS REDISPLAYED   *)
(*        WITH AN ERROR MESSAGE.  THIS PANEL CONTINUES TO BE DIS-    *)
(*        PLAYED UNTIL EXIT OR NO MORE FIELDS IS CHOSEN.             *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCFLDIN *)
(**)
  PROCEDURE SCFLDIN(VAR   IRC        : RET_REC;
                    VAR   INCLD      : TEXT;
                    VAR   FIELD_LIST : LISTKEY;
                    CONST FIELD_DEF  : T_FIELD_DEF);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES THE FIELD TYPE DECLARATION TO THE      *)
(*        PASCAL INCLUDE FILE.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC           I/O  RETURN CODE                           *)
(*        INCLD          I   THE FILE NAME                         *)
(*        FIELD_LIST     I   A LIST OF FIELDS                      *)
(*        FIELD_DEF      I   INDICATES THE FIELD DEFINITION TYPE   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        SET THE LIST OF FIELDS TO BE READ FORWARD                  *)
(*        CONTINUE TO READ FIELDS UNTIL THE END OF LIST IS FOUND     *)
(*          GET THE KEY TO THE NEXT FIELD                            *)
(*          GET THE FIELD'S ADB                                      *)
(*          WRITE TO THE FILE <FIELD NAME>                           *)
(*          IF THE TYPE OF DEFINITION IS ADB THEN                    *)
(*            SET THE FIELD'S CONSTITUENT LIST TO BE READ FORWARD    *)
(*            GET THE KEY TO THE FIRST ENTITY ON THE CONSTITUENT LIST *)
(*            GET THE CONSTITUENT'S ADB                              *)
(*            CALL THE ROUTINE TO WRITE OUT THE PRIMITIVE TYPES THAT *)
(*                WILL APPEAR IN THE ADB OF THE ENTITY               *)
(*              INTEGER                                              *)
(*              REAL                                                 *)
(*              LOGICAL                                              *)
(*              STRING                                               *)
```

```
(*              DEFINED TYPE                                          *)
(*              ARRAY                                                 *)
(*          ELSE                                                      *)
(*            IF THE TYPE OF DEFINITION IS THE CONSTITUENT LIST       *)
(*                                                   POSITION THEN    *)
(*              WRITE TO THE FILE T_CL_POSITION;                      *)
(*            ELSE                                                    *)
(*              IF THE DEFINITION TYPE IS CONSTITUENT KEY THEN        *)
(*                 WRITE TO THE FILE ENTKEY;                          *)
(*        END;                                                        *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE SCFLDSRT *)
(**)
  PROCEDURE SCFLDSRT(CONST CURRENT : ENTBLOCK;
                     CONST NEXT    : ENTBLOCK;
                     VAR   FLIP    : BOOLEAN;
                     VAR   RRC     : EXT_RET_CODE;
                     VAR   PROC    : ROUTINE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE FINDS THE KEY AND RETURNS IT TO THE CALLING         *)
(*        PROCEDURE GIVEN A NAME OR USER DEFINED KIND NUMBER AS WELL       *)
(*        AS THE ENTITY KIND.                                              *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ==========                                   *)
(*        ENTITY_KEY      I   KEY TO THE ENTITY                            *)
(*        ADB             O   DATA STORED IN THE ADB                       *)
(*        DATA           I/O  THE USER DEFINED DATA STRUCTURE USED         *)
(*                            TO PASS DATA INTO THIS PROCEDURE AND         *)
(*                            TO GET THE DESIRED OUTPUT FROM THE           *)
(*                            PROCEDURE                                    *)
(*        RRC             O   EXTERNAL RETURN CODE                         *)
(*                            = 0  OK                                      *)
(*                            >= 10  ERROR                                 *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THIS ROUTINE IS CALLED BY THE MAS EXECUTE ROUTINES,              *)
(*        MAKXEQ AND MAEXEQ.  THE LIST IS SEARCHED FOR IDENTICAL           *)
(*        DATA AND IF IT IS FOUND THE ENTITY'S KEY IS RETURNED TO          *)
(*        THE CALLING PROCEDURE AS WELL A FLAG INDICATING THAT THE         *)
(*        KEY RETURNED IS THE CORRECT ONE.                                 *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCFLDST *)
(**)
  PROCEDURE SCFLDST(VAR   IRC           : RET_REC;
                    VAR   FIELD_LIST    : LISTKEY;
                    VAR   FIELDS_IN_ADB : LISTKEY;
                    VAR   FIELDS_IN_CL  : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE SORTS THE FIELDS INTO TWO GROUPS AND THEN           *)
(*        ORDERS THEM ACCORDING TO OFFSET OR POSITION DEPENDING ON         *)
(*        WHETHER OR NOT THEY ARE IN THE LIST OF ADB FIELDS OR THE         *)
(*        LIST OF CONSTITUENT FIELDS.                                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC            I/O  RETURN CODE                                  *)
(*        FIELD_LIST      I   A LIST OF FIELDS                             *)
(*        FIELDS_IN_ADB   O   A LIST OF FIELDS IN THE ADB                  *)
(*        FIELDS_IN_CL    O   A LIST OF FIELDS IN THE CONSTITUENT LIST      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THIS ROUTINE READS EACH ENTITY FROM THE LIST AND WRITES          *)
(*        OUT THE ENTITY NAME AND CONSTANT TO THE INCLUDE FILE.            *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCFLDUP *)
(**)
  PROCEDURE SCFLDUP(VAR IRC             : RET_REC;
                    VAR TRANS_STACK     : TRANSPTR;
                    VAR LIST_OF_FIELDS : LISTKEY;
                    VAR FIELDTYP        : T_FIELDTYPE;
                    VAR FIELD_KEY       : ENTKEY;
                    VAR DELETE_LIST     : LISTKEY;
                    VAR NEW_KEYS_LIST   : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE FIELD ENTITY.  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME            I/O  DESCRIPTION                           *)
(*        ====            ===  ===========                           *)
(*        IRC              O   RETURN CODE                           *)
(*        TRANS_STACK     I/O  POINTS TO THE TRANSACTION STACK       *)
(*        LIST_OF_FIELDS I/O  LIST OF THE ENTITY'S FIELDS FROM WHICH *)
(*                            A FIELD IS TO BE UPDATED               *)
(*        FIELD_KEY       I/O  KEY OF THE FIELD TO BE UPDATED        *)
(*        DELETE_LIST     I/O  LIST OF ENTITIES TO BE DELETED        *)
(*        NEW_KEYS_LIST   I/O  LIST OF NEWLY CREATED ENTITIES        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE DATA ABOUT THE FIELD ENTITY IS DISPLAYED BY CALLING    *)
(*        THE MENU INTERFACE ROUTINE (UPFIELD).  THE USER CAN CHANGE *)
(*        ANY OF THE DATA ON THE PANEL OR SELECT ONE OF THE FOLLOWING *)
(*        OPTIONS :                                                  *)
(*          REVIEW THE CURRENT TYPE,                                 *)
(*          UPDATE THE CURRENT TYPE,                                 *)
(*          SAVE THE CHANGES MADE,                                   *)
(*          RETURN OR EXIT.                                          *)
```

```
(*        IF SAVE THE CHANGES IS CHOSEN AND THE USER HAS CHANGED ANY  *)
(*        OF THE FIELD DATA THE FIELD KEY IS PLACED ON THE DELETE     *)
(*        LIST AND A NEW FIELD ENTITY IS MODELED.  THE KEY TO THE NEW *)
(*        FIELD ENTITY IS PLACE ON THE NEW KEYS LIST.                 *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
```

```
(* %INCLUDE SCFNDKEY *)
(**)
  PROCEDURE SCFNDKEY(CONST ENTITY_KEY : ENTKEY;
                     VAR  ADB : ENTBLOCK;
                     VAR  DATA : BLKDATA;
                     VAR  RRC : EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE FINDS THE KEY AND RETURNS IT TO THE CALLING   *)
(*        PROCEDURE GIVEN A NAME OR USER DEFINED KIND NUMBER AS WELL *)
(*        AS THE ENTITY KIND.                                        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        ENTITY_KEY      I   KEY TO THE ENTITY                     *)
(*        ADB             O   DATA STORED IN THE ADB                *)
(*        DATA           I/O  THE USER DEFINED DATA STRUCTURE USED  *)
(*                            TO PASS DATA INTO THIS PROCEDURE AND  *)
(*                            TO GET THE DESIRED OUTPUT FROM THE    *)
(*                            PROCEDURE                             *)
(*        RRC             O   EXTERNAL RETURN CODE                  *)
(*                            = 0  OK                               *)
(*                            >= 10  ERROR                          *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE IS CALLED BY THE MAS EXECUTE ROUTINES,        *)
(*        MAKXEQ AND MAEXEQ.  THE LIST IS SEARCHED FOR IDENTICAL     *)
(*        DATA AND IF IT IS FOUND THE ENTITY'S KEY IS RETURNED TO    *)
(*        THE CALLING PROCEDURE AS WELL A FLAG INDICATING THAT THE   *)
(*        KEY RETURNED IS THE CORRECT ONE.                           *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCGENRPT *)
(**)
  PROCEDURE SCGENRPT(VAR IRC         :  RET_REC;
                     VAR REPORT_TYPE :  OPERATIONS;
                     VAR MSG         :  MESSAGE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DETERMINES THE SUBSCHEMA FOR WHICH A REPORT   *)
(*        IS TO BE GENERATED AND CALLS THE APPROPRIATE ROUTINE TO    *)
(*        PRODUCE THE REPORT.                                        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC            I/O  RETURN CODE                           *)
(*        REPORT_TYPE     I   TYPE OF REPORT TO BE GENERATED        *)
(*        MSG            I/O  PANEL MESSAGE                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY A PANEL CONTAINING A LIST OF ALL OF THE SUBSCHEMAS *)
(*                                          WITHIN THE SCHEMA MODEL. *)
(*        IF RETURN OR EXIT WAS NOT CHOSEN ON THE PANEL THEN         *)
(*          IF A MULTIPLE SELECT WAS MADE ON THE PANEL THEN          *)
(*            DISPLAY AN ERROR MESSAGE                               *)
(*          ELSE                                                     *)
(*            GET THE KEY OF THE SUBSCHEMA SELECTED                  *)
(*            IF THE SUBSCHEMA HAS NOT BEEN PHYSICALIZED THEN        *)
(*              PHYSICALIZE THE SUBSCHEMA                            *)
(*            CALL THE APPROPRIATE ROUTINE TO PRODUCE A REPORT       *)
(*        ELSE                                                       *)
(*          IF RETURN WAS SELECTED THEN                              *)
(*            RETURN TO THE REPORT MENU                              *)
(*          ELSE                                                     *)
(*            IF EXIT WAS SELECTED THEN                              *)
(*              RETURN TO THE MAIN MENU                              *)
```

```
(*            ELSE                                                    *)
(*              DISPLAY AN ERROR MESSAGE FOR AN INVALID OPTION        *)
(*        END;                                                        *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE SCHDVR *)
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        THIS IS THE MAINLINE PROGRAM WHICH DRIVES THE SCHEMA        *)
(*        EXECUTIVE PACKAGE.                                          *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*        DEF                                                         *)
(*          CURRENT_LIST   I/O   POINTS TO THE LIST KEY CURRENTLY     *)
(*                               IN USE                               *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*        DDNAMES USED WITH STANDARD FILES:                           *)
(*             NONE                                                   *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERFACE PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        THIS ROUTINE INITIALIZES THE NETWORK, THEN DETERMINES       *)
(*        IF THE PACKAGE IS TO BE USED INTERACTIVELY OR BY BATCH      *)
(*        AND CONTINUES PROCESSING ACCORDINGLY.                       *)
(*                                                                    *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE SCINCLD *)
(**)
  PROCEDURE SCINCLD(VAR IRC    : RET_REC;
                    VAR MSG    : MESSAGE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GENERATES THE PASCAL INCLUDE FILES AND WRITES  *)
(*        THESE DEFINITIONS TO A FILE                                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC            I/O  RETURN CODE                           *)
(*        MSG            I/O  PANEL MESSAGE                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY A PANEL CONTAINING A LIST OF ALL OF THE SUBSCHEMAS  *)
(*                                          WITHIN THE SCHEMA MODEL. *)
(*        IF RETURN OR EXIT WAS NOT CHOSEN ON THE PANEL THEN         *)
(*          IF A MULTIPLE SELECT WAS MADE ON THE PANEL THEN          *)
(*            DISPLAY AN ERROR MESSAGE                               *)
(*          ELSE                                                     *)
(*            GET THE KEY OF THE SUBSCHEMA SELECTED                  *)
(*            IF THE SUBSCHEMA HAS NOT BEEN PHYSICALIZED THEN        *)
(*              PHYSICALIZE THE SUBSCHEMA                            *)
(*            WRITE OUT TO THE FILE THE HEADING FOR THE INCLUDES     *)
(*            MAKE AN INCLUSIVE LIST OF ENTITIES WITHIN THE SUBSCHEMA *)
(*            DELETE ANY DUPLICATES ON THE LIST                      *)
(*            ALPHABETICALLY SORT THE ENTITIES                       *)
(*            WRITE OUT TO THE FILE THE ENTITY KIND CONSTANTS        *)
(*            WRITE OUT TO THE FILE THE DEFINED TYPE DECLARATIONS    *)
(*            WRITE OUT TO THE FILE THE ENTITY DECLARATIONS          *)
(*            WRITE OUT TO THE FILE THE MAS ENTITY DECLARATIONS      *)
(*            WRITE OUT TO THE FILE THE KEYBLOCK DECLARATIONS        *)
```

```
(*        ELSE                                                          *)
(*          IF RETURN WAS SELECTED THEN                                 *)
(*            RETURN TO THE REPORT MENU                                 *)
(*          ELSE                                                        *)
(*            IF EXIT WAS SELECTED THEN                                 *)
(*              RETURN TO THE MAIN MENU                                 *)
(*            ELSE                                                      *)
(*              DISPLAY AN ERROR MESSAGE FOR AN INVALID OPTION          *)
(*        END;                                                          *)
(*                                                                      *)
(*    $COMMENTS:                                                        *)
(*                                                                      *)
(*    $CHANGE CONTROL:                                                  *)
(*                                                                      *)
```

```
(* %INCLUDE SCINTCR *)
(**)
  PROCEDURE SCINTCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE     *)
(*         INTEGER ENTITY AND PUSHES THE DATA ON THE TRANSACTION     *)
(*         STACK.                                                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   THE RETURN CODE                      *)
(*         TRANS_STACK    I/O  THE TRANSACTION STACK                *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         INSIDE         I/O  INDICATES IF THE EXIT OPTION OR      *)
(*                             THE RETURN OPTION IS CHOSEN WITHIN    *)
(*                             ANOTHER ROUTINE                       *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRINTEGR    *)
(*         TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS   *)
(*         VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION  *)
(*         STACK.  OTHERWISE, IF THE DATA ENTERED IS INVALID THE     *)
(*         PANEL IS REDISPLAYED.                                     *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCINTUP *)
(**)
  PROCEDURE SCINTUP(VAR IRC : RET_REC;
                    VAR INTEGER_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS : INTEGER;
                    VAR DELETE_LIST : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*         THIS ROUTINE UPDATES THE INTEGER ENTITY.                        *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME             I/O  DESCRIPTION                               *)
(*         ====             ===  ===========                               *)
(*         INTEGER_KEY      I/O  KEY OF THE ENTITY TO BE UPDATED           *)
(*         NUMBER_OF_USERS   I   NUMBERS OF USERS OF THE INTEGER ENTITY    *)
(*         DELETE_LIST      I/O  LIST OF ENTITIES TO BE DELETED            *)
(*         NEW_KEYS_LIST    I/O  LIST OF NEWLY CREATED ENTITIES            *)
(*         IRC               O   RETURN CODE                               *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*         NONE                                                            *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*         THIS ROUTINE DISPLAYS THE PRECISION OF THE INTEGER ENTITY       *)
(*         AND ALLOWS THE USER TO CHANGE THE INFORMATION IF HE/SHE SO      *)
(*         DESIRES.  IF A CHANGE IS MADE AND THE NUMBER OF USERS OF        *)
(*         THE OLD INTEGER ENTITY IS ONE OR LESS THEN ITS KEY IS           *)
(*         PLACED ON THE DELETE LIST.  AFTER THE NEW INTEGER ENTITY        *)
(*         IS CREATED ITS KEY IS PLACED ON THE NEW KEYS LIST.              *)
(*                                                                         *)
(*    $COMMENTS:                                                           *)
(*                                                                         *)
(*    $CHANGE CONTROL:                                                     *)
(*                                                                         *)
```

```
(* %INCLUDE SCKEFIND *)
(**)
  PROCEDURE SCKEFIND(VAR IRC : RET_REC;
                     CONST ENT_KIND : INTEGER;
                     VAR ENT_KEY : ENTKEY;
                     VAR CREATE  : BOOLEAN);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE GETS THE KEY TO THE ENTITY TO BE UPDATED OR        *)
(*        REVIEWED.                                                       *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        ENT_KIND       I   KIND OF THE ENTITY                           *)
(*        ENT_KEY        O   KEY OF THE ENTITY                            *)
(*        CREATE         O   INDICATES WHETHER CREATE OR REVIEW           *)
(*        IRC            O   RETURN CODE                                  *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THIS ROUTINE MAKES A LIST OF ENTITIES OF THE KIND               *)
(*        SPECIFIED AND DISPLAYS THIS LIST OF ENTITIES BY CALLING         *)
(*        THE MENU INTERFACE ROUTINE (DISPLIST).  IF AN ENTITY IS         *)
(*        CHOSEN THEN MAKXEQ IS CALLED TO GET THE KEY TO THE ENTITY       *)
(*        IF IT EXISTS.  IF AN INVALID PICK IS MADE A MESSAGE IS          *)
(*        DISPLAYED ON THE DISPLIST PANEL.                                *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE SCKEYIN *)
(**)
  PROCEDURE SCKEYIN(VAR IRC         : RET_REC;
                    VAR INCLD       : TEXT;
                    VAR ENTITY_LIST : LISTKEY;
                    VAR CL_DEFN     : LISTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE WRITES THE KEYBLOCK TYPE DECLARATION TO THE        *)
(*        PASCAL INCLUDE FILE.                                            *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC            I/O  RETURN CODE                                 *)
(*        INCLD           I   THE FILE NAME                               *)
(*        ENTITY_LIST     I   A LIST OF ENTITIES                          *)
(*        CL_DEFN         I   A LIST OF ENTITIES WHO HAVE CONSTITUENT     *)
(*                            LIST DEFINITIONS                            *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        WRITE THE KEYBLOCK HEADING TO THE FILE                          *)
(*        WRITE TYPE TO THE FILE                                          *)
(*        WRITE OUT TO THE FILE KEYBLOCK = RECORD                         *)
(*        WRITE OUT TO THE FILE CASE INTEGER OF                           *)
(*        SET THE LIST OF ENTITIES TO BE READ IN THE FORWARD              *)
(*        CONTINUE TO READ ENTITIES UNTIL THE END OF LIST IS FOUND        *)
(*          GET THE KEY TO THE NEXT ENTITY ON THE LIST                    *)
(*          GET THE ENTITY'S ADB                                          *)
(*          WRITE TO THE FILE C_<ENTITY NAME> :                           *)
(*          IF THE ENTITY IS IN THE LIST OF GENERATED CONSTITUENT         *)
(*                                                  DEFINITIONS THEN      *)
(*            WRITE TO THE FILE (<ENTITY NAME> : C_<ENTITY NAME>);        *)
(*          ELSE                                                          *)
(*            WRITE TO THE FILE ();                                       *)
(*        END;                                                            *)
(*                                                                        *)
```

```
(*    $COMMENTS:                                                    *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE SCLISTCR *)
(**)
  PROCEDURE SCLISTCR(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE     *)
(*         LIST ENTITY AND PUSHES THE DATA ON THE TRANSACTION        *)
(*         STACK.                                                    *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   THE RETURN CODE                      *)
(*         TRANS_STACK    I/O  THE TRANSACTION STACK                *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION*)
(*                             IS CHOSEN WITHIN ANOTHER ROUTINE      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRLIST      *)
(*         TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS   *)
(*         VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION  *)
(*         STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE  *)
(*         DATA ENTERED IS INVALID THEN THE PANEL IS REDISPLAYED     *)
(*         WITH AN ERROR MESSAGE.                                    *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*      NARATION ON THE NEXT LINE.                                   *)
(*                                                                   *)
```

```
(*    REVISED: 10/09/87          C. H. MOHME                  DBMA     *)
(*     ADDED PARAMETER TO SCDEFCR CALL.                                 *)
(*                                                                      *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*------------------------------------------------------------------------*)
(*                                                                      *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCLISTCR *)
```

```
(* %INCLUDE SCLISTUP *)
(**)
  PROCEDURE SCLISTUP(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR;
                     VAR LIST_KEY : ENTKEY;
                     VAR NUMBER_OF_USERS : LISTKEY;
                     VAR DELETE_LIST : LISTKEY;
                     VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE A LIST.            *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*     IRC               O    RETURN CODE                           *)
(*     TRANS_STACK       I/O  POINTS TO THE TRANSACTION STACK       *)
(*     LIST_KEY          I/O  KEY OF THE LIST TO BE UPDATED         *)
(*     NUMBER_OF_USERS   I/O  THE NUMBER OF USERS OF THE LIST       *)
(*     DELETE_LIST       I/O  LIST OF ENTITIES TO BE DELETED        *)
(*     NEW_KEYS_LIST     I/O  LIST OF NEWLY CREATED ENTITIES        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE CURRENT LIST DATA IS DISPLAYED ON THE UPLIST PANEL.    *)
(*        THE DATA CAN THEN BE UPDATED BY THE USER.                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE         *)
(*     NARATION ON THE NEXT LINE.                                    *)
```

```
(*                                                                    *)
(*    REVISED: MM/DD/YY CCXX      I. M. APERSON              GROUP_ID *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                              *)
(*                                                                    *)
(*    ORIGINATED: 08/13/87        C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE SCLISTUP *)
```

```
(* %INCLUDE SCMASIN *)
(**)
  PROCEDURE SCMASIN(VAR IRC         : RET_REC;
                    VAR INCLD       : TEXT;
                    VAR ENTITY_LIST : LISTKEY;
                    VAR ADB_DEFN    : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE WRITES THE MAS ADB DECLARATION TO THE PASCAL        *)
(*        INCLUDE FILE.                                                    *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC            I/O  RETURN CODE                                  *)
(*        INCLD           I   THE FILE NAME                                *)
(*        ENTITY_LIST     I   A LIST OF ENTITIES                           *)
(*        ADB_DEFN        I   A LIST OF ENTITIES THAT HAVE AN ADB          *)
(*                            DEFINITION GENERATED                         *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        WRITE THE HEADING TO THE FILE                                    *)
(*        WRITE THE FOLLOWING TO THE FILE                                  *)
(*                    ENTBLOCK = RECORD                                    *)
(*                      KIND      : INTEGER;                               *)
(*                      LENGTH    : INTEGER;                               *)
(*                      SYSUSE    : INTEGER;                               *)
(*                      VERSION   : INTEGER;                               *)
(*                      SYS_IDENT : INTEGER;                               *)
(*                      IDENT     : INTEGER;                               *)
(*                      CASE INTEGER OF                                    *)
(*        SET THE LIST OF ENTITIES TO BE READ FORWARD                      *)
(*        CONTINUE TO READ ENTITIES UNTIL THE END OF LIST IS FOUND         *)
(*          GET THE KEY TO THE NEXT ENTITY IN THE LIST                     *)
(*          GET THE ENTITY'S ADB                                           *)
(*          WRITE TO THE FILE K_<ENTITY NAME> :                            *)
```

```
(*          IF THE ENTITY IS IN THE LIST OF GENERATED ADB DEFINITIONS *)
(*            WRITE TO THE FILE (<ENTITY NAME> : E_<ENTITY NAME>);    *)
(*          ELSE                                                      *)
(*            WRITE TO THE FILE ();                                   *)
(*        END;                                                        *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
```

```
(* %INCLUDE SCMEMAD *)
(**)
  PROCEDURE SCMEMAD(VAR IRC : RET_REC;
                    VAR ENT_KEY: ENTKEY;
                    VAR LIST_OF_CNSTS : LISTKEY;
                    VAR CHOSEN_KEY : ENTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE DISPLAYS A LIST OF MEMBERS FROM WHICH THE    *)
(*         USER CAN SELECT.  THE ENTITY KEY OF THE MEMBER SELECTED IS*)
(*         RETURNED.                                                 *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   THE INTERNAL RETURN CODE INDICATING  *)
(*                             WHETHER AN ERROR HAS OCCURRED.        *)
(*         ENT_KEY         I   THE KEY OF THE ENTITY TO WHICH MEMBERS*)
(*                             ARE BEING ADDED                       *)
(*         LIST_OF_CNSTS   I   THE LIST OF ALL CONSTITUENTS          *)
(*         CHOSEN_KEY      O   THE ENTITY KEY OF THE MEMBER SELECTED *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE BEGINS WITH THE CONSTITUENT LIST AND REMOVES *)
(*         FROM IT THOSE CONSTITUENTS WHICH ARE ALREADY MEMBERS.  IT *)
(*         ALSO REMOVES THE CONSTITUENT TC WHICH MEMBERS ARE BEING   *)
(*         ADDED.  THE PANEL DISPLIST DISPLAYS THE MEMBERS AND THE   *)
(*         USER CAN SELECT ONE.  THE ENTITY KEY OF THE MEMBER SE-    *)
(*         LECTED, IF ANY, IS RETURNED.                              *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCMEMLST *)
(**)
  PROCEDURE SCMEMLST(VAR IRC         : RET_REC;
                     VAR TRANS_STACK : TRANSPTR;
                     VAR ENT_TYP     : ENTITY_TYPE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE LISTS OUT THE CURRENT MEMBERS/CONSTITUENTS OF *)
(*         THE ENTITY, CLASS, SUBSCHEMA, POINTER, STRUCTURE, GLOBAL  *)
(*         FIELD AND ENUMERATION ENTITIES.                           *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         TRANS_STACK   I/O  POINTER TO THE TRANSACTION STACK      *)
(*         ENT_TYP        I   THE ENTITY TYPE                       *)
(*         IRC            O   INTERNAL RETURN CODE                  *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       NONE                                                        *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE SEARCHES THROUGH THE TRANSACTION STACK FOR   *)
(*         THE CONSTITUENTS OF THE ENTITY TYPE PASSED IN.  THE CON-  *)
(*         STITUENT NAMES ARE PLACED IN AN ARRAY THAT IS DISPLAYED   *)
(*         BY CALLING THE MENU INTERFACE ROUTINE (LMEM23).           *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCPOPKE *)
(**)
  PROCEDURE SCPOPKE(VAR IRC : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE 'POPS' A KEY OFF OF THE KEY STACK.                  *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   THE RETURN CODE                             *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      REF                                                                *)
(*        CURRENT_LIST    O   LIST KEY POPPED OFF OF THE KEY STACK         *)
(*        KEY_STACK      I/O  POINTER TO THE KEY STACK                     *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        IF THE KEY STACK IS NIL THEN AN ERROR MESSAGE IS RETURNED        *)
(*        TO THE CALLING PROCEDURE.  OTHERWISE, THE KEY DATA IS            *)
(*        'POPPED' OFF OF THE STACK AND THE NODE IS DISPOSED OF BY         *)
(*        CALLING THE BUILT IN PASCAL PROCEDURE 'DISPOSE'.                 *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND      *)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                          *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCPOPTR *)
(**)
  PROCEDURE SCPOPTR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR;
                    VAR DATA : TRANSACTION);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE 'POPS' THE TRANSACTION DATA OFF OF THE       *)
(*         DYNAMICALLY ALLOCATED STACK.                              *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   INTERNAL RETURN CODE                 *)
(*         TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK      *)
(*         DATA            O   THE TRANSACTION DATA                 *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         NONE                                                      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         IF THE TRANSACTION STACK IS NIL THEN AN ERROR MESSAGE IS  *)
(*         RETURNED TO THE CALLING PROCEDURE. OTHERWISE, THE TRANS-  *)
(*         ACTION DATA IS 'POPPED' OFF OF THE STACK AND THE NODE     *)
(*         IS DISPOSED OF BY CALLING THE BUILT IN PASCAL PROCEDURE   *)
(*         DISPOSE.                                                  *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
```

```
(* %INCLUDE SCPRMFL *)
(**)
  PROCEDURE SCPRMFL(VAR IRC              : RET_REC;
                    VAR LIST_OF_ENTITIES : LISTKEY;
                    VAR LISTARRAY        : T_ARRAY23;
                    VAR INDEX            : INTEGER);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    $FUNCTION:                                                          *)
(*         THIS ROUTINE FILLS AN ARRAY WITH THE NAMES OF THE ENTITIES     *)
(*         IN THE LIST OF ENTITIES.                                       *)
(*                                                                        *)
(*    $DESCRIPTION OF ARGUMENTS:                                          *)
(*         NAME              I/O  DESCRIPTION                            *)
(*         ====              ===  ===========                            *)
(*         LIST_OF_ENTITIES   I   LIST OF ENTITIES TO BE PUT IN ARRAY     *)
(*         LISTARRAY         I/O  ARRAY CONTAINING THE ENTITY NAMES       *)
(*         INDEX             I/O  INDEX OF THE CURRENT ARRAY POSITION     *)
(*         IRC                O   INTERNAL RETURN CODE                    *)
(*                                                                        *)
(*    $COMMONS:                                                           *)
(*         NONE                                                           *)
(*                                                                        *)
(*    $ENVIRONMENT:                                                       *)
(*         LANGUAGE: IBM PASCAL                                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                        *)
(*    $EXECUTION PROCEDURE:                                               *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                    *)
(*                                                                        *)
(*    $PROCESSING DESCRIPTION:                                            *)
(*         THE LIST OF ENTITIES PASSED IN IS TRAVERSED AND EACH           *)
(*         ENTITY'S ADB IS RETRIEVED.  THE ENTITY'S ADB DATA IS           *)
(*         TRANSLATED INTO CHARACTER FORMAT IF NECESSARY, AND IS          *)
(*         PLACED INTO THE LISTARRAY.  THIS PROCESS CONTINUES UNTIL       *)
(*         THE END OF THE LIST IS ENCOUNTERED OR THE MAXIMUM ARRAY        *)
(*         SIZE IS EXCEEDED.                                              *)
(*                                                                        *)
(*    $COMMENTS:                                                          *)
(*                                                                        *)
(*    $CHANGE CONTROL:                                                    *)
(*                                                                        *)
```

```
(* %INCLUDE SCPRMRE *)
(**)
  PROCEDURE SCPRMRE(VAR IRC     : RET_REC;
                    VAR ENT_KEY : ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE GATHERS THE DATA TO REVIEW THE ENTITIES AND         *)
(*        CALLS THE MENU INTERFACE ROUTINES TO DISPLAY THIS DATA.          *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        ENT_KEY        I   KEY OF THE ENTITY TO BE REVIEWED              *)
(*        IRC            O   INTERNAL RETURN CODE                          *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        FIRST THE ADB OF THE ENTITY PASSED IN MUST BE RETRIEVED,         *)
(*        THEN ACCORDING TO THE ENTITY'S TYPE THE ADB DATA IS TRANS-       *)
(*        LATED INTO CHARACTER FORMAT IF IT IS NECESSARY.  IF THE          *)
(*        ENTITY HAS CONSTITUENTS THAT ARE TO BE DISPLAYED WITH THE        *)
(*        ENTITY'S DATA, THEN THE DATA ABOUT THE CONSTITUENTS MUST         *)
(*        BE PUT INTO A FORMAT THE CAN BE DISPLAYED ALSO. THEN THE         *)
(*        APPROPRIATE MENU INTERFACE ROUTINE IS CALLED TO DISPLAY          *)
(*        THIS INFORMATION.                                                *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCPTRCR *)
(**)
  PROCEDURE SCPTRCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE      *)
(*        POINTER ENTITY AND PUSHES THE DATA ON THE TRANSACTION      *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*        TRANS_STACK    I/O  THE TRANSACTION STACK                 *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRPNTR       *)
(*        TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS    *)
(*        VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION   *)
(*        STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE   *)
(*        DATA IS INVALID THE PANEL IS REDISPLAYED WITH AN ERROR     *)
(*        MESSAGE.                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCPTRUP *)
(**)
  PROCEDURE SCPTRUP(VAR IRC             : RET_REC;
                    VAR POINTER_KEY     : ENTKEY;
                    VAR NUMBER_OF_USERS : INTEGER;
                    VAR DELETE_LIST     : LISTKEY;
                    VAR NEW_KEYS_LIST   : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE POINTER ENTITY. *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME             I/O  DESCRIPTION                         *)
(*        ====             ===  ===========                         *)
(*        POINTER_KEY      I/O  KEY OF THE ENTITY TO BE UPDATED     *)
(*        NUMBER_OF_USERS   I   THE NUMBER OF USERS OF THIS ENTITY  *)
(*        DELETE_LIST      I/O  LIST OF ENTITIES TO DELETE IF THE   *)
(*                              CHANGES MADE ARE SAVED               *)
(*        NEW_KEYS_LIST    I/O  LIST OF ENTITIES JUST CREATED DURING *)
(*                              THE UPDATE PROCESS                   *)
(*        IRC               O   RETURN CODE                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (UPPNTR)     *)
(*        WHICH DISPLAYS INFORMATION ABOUT THE POINTER ENTITY.  THE  *)
(*        UPDATE POINTER OPTIONS INCLUDE THE FOLLOWING :             *)
(*           UPDATE THE CONSTITUENT LIST BY ADDING OR REMOVING AN    *)
(*             ELEMENT,                                              *)
(*           REVIEW A CONSTITUENT,                                   *)
(*           SAVE THE CHANGES MADE,                                  *)
(*           RETURN AND EXIT.                                        *)
(*        IF THE CONSTITUENT LIST WAS CHANGED AND NO MORE CHANGES    *)
(*        WAS SELECTED THE OLD POINTER ENTITY KEY IS PLACED ON THE   *)
(*        DELETE LIST AND A NEW POINTER ENTITY IS MODELED.  THE NEW  *)
(*        POINTER ENTITY KEY IS PLACED ON THE NEW KEYS LIST.         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCPUSHKE *)
(**)
  PROCEDURE SCPUSHKE(VAR IRC : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE 'PUSHES' THE KEY DATA ON TO THE DYNAMICALLY   *)
(*        ALLOCATED KEY STACK.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST   I/O  A KEY TO THE LIST CURRENTLY IN USE     *)
(*        KEY_STACK      I/O  POINTER TO THE KEY STACK               *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        A NODE IS DYNAMICALLY ALLOCATED BY CALLING THE BUILT IN    *)
(*        PROCEDURE 'NEW'.  THE KEY DATA IS THEN 'PUSHED' ONTO THE   *)
(*        KEY STACK.                                                 *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        TEXT OF ANY FURTHER COMMENTS WHICH MIGHT HELP TO UNDERSTAND *)
(*        THE FUNCTION/EXECUTION OF THIS ROUTINE.                    *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCPUSHTR *)
(**)
  PROCEDURE SCPUSHTR(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR;
                     CONST DATA : TRANSACTION);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE 'PUSHES' THE TRANSACTION DATA ON TO THE       *)
(*        DYNAMICALLY ALLOCATED STACK.                               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   INTERNAL RETURN CODE                   *)
(*        TRANS_STACK    I/O  POINTER TO THE TRANSACTION STACK       *)
(*        DATA            I   THE TRANSACTION DATA                   *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        A NODE IS ALLOCATED BY CALLING THE BUILT IN PASCAL         *)
(*        PROCEDURE 'NEW'. THE TRANSACTION DATA IS THEN 'PUSHED'     *)
(*        ONTO THE STACK.                                            *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCRELCR *)
(**)
  PROCEDURE SCRELCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE      *)
(*        REAL ENTITY AND PUSHES THE DATA ON THE TRANSACTION         *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*        TRANS_STACK    I/O  THE TRANSACTION STACK                 *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OPTION OR       *)
(*                            THE RETURN OPTION IS CHOSEN WITHIN     *)
(*                            ANOTHER ROUTINE                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRREAL       *)
(*        TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS    *)
(*        VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION   *)
(*        STACK.  OTHERWISE, IF THE DATA ENTERED IS INVALID THE      *)
(*        PANEL IS REDISPLAYED.                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCRELUP *)
(**)
  PROCEDURE SCRELUP(VAR IRC : RET_REC;
                    VAR REAL_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS : INTEGER;
                    VAR DELETE_LIST : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE UPDATES THE REAL ENTITY.                      *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME              I/O  DESCRIPTION                         *)
(*        ====              ===  ===========                         *)
(*        REAL_KEY          I/O  KEY OF THE ENTITY TO BE UPDATED     *)
(*        NUMBER_OF_USERS    I   NUMBER OF USERS OF THE REAL ENTITY  *)
(*        DELETE_LIST       I/O  LIST OF ENTITIES TO BE DELETED      *)
(*        NEW_KEYS_LIST     I/O  LIST OF NEWLY CREATED ENTITIES      *)
(*        IRC                O   RETURN CODE                         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE DISPLAYS THE PRECISION OF THE REAL ENTITY AND *)
(*        ALLOWS THE USER TO CHANGE THE INFORMATION IF HE/SHE SO     *)
(*        DESIRES.  IF A CHANGE IS MADE AND THE NUMBER OF USERS OF   *)
(*        THE OLD REAL ENTITY IS ONE OR LESS THEN ITS KEY IS PLACED  *)
(*        ON THE DELETE LIST.  AFTER THE NEW REAL ENTITY IS CREATED  *)
(*        THIS KEY IS PLACED ON THE NEW KEYS LIST.                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCREVIEW *)
(**)
  PROCEDURE SCREVIEW(VAR IRC         : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE  *)
(*        EDIT OPTION CHOSEN.                                        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   RETURN CODE                           *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        DEF                                                        *)
(*           INSIDE      I/O  INDICATES IF THE EXIT OPTION HAS      *)
(*                            BEEN CHOSEN WITHIN ANOTHER CREATE      *)
(*                            PROCEDURE                              *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        CALLS THE REVIEW MENU INTERFACE ROUTINE (MREVIEW) AND      *)
(*        PROCESSES THE DATA RECIEVED FROM THE MENU EITHER BY        *)
(*        CALLING THE APPROPRIATE ROUTINE OR EXITING THE PROCEDURE.  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
```

```
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    REVISED: 09/28/87          C. H. MOHME                DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                          *)
(*                                                                    *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE SCREVIEW *)
```

```
(* %INCLUDE SCRPTM *)
(**)
  PROCEDURE SCRPTM(VAR IRC : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DETERMINES THE REPORT OPTION SELECTED AND     *)
(*        CALLS THE ROUTINE WHICH GENERATES THE REPORT.              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   RETURN CODE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        CALL THE MENU INTERFACE ROUTINE TO DISPLAY THE REPORT      *)
(*        MENU (RPTMENU) AND THEN CALL THE APPROPRIATE ROUTINE       *)
(*        TO GENERATED THE REPORT.                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCSETCR *)
(**)
  PROCEDURE SCSETCR(VAR IRC         : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE      *)
(*        SET ENTITY AND PUSHES THE DATA ON THE TRANSACTION          *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*        TRANS_STACK    I/O  THE TRANSACTION STACK                 *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRSET        *)
(*        TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS    *)
(*        VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION   *)
(*        STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE   *)
(*        DATA ENTERED IS INVALID THEN THE PANEL IS REDISPLAYED      *)
(*        WITH AN ERROR MESSAGE.                                     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE         *)
(*     NARATION ON THE NEXT LINE.                                    *)
(*                                                                   *)
```

```
(*    REVISED: 10/09/87          C. H. MOHME                    DBMA     *)
(*     ADDED PARAMETER TO SCDEFCR CALL.                                   *)
(*                                                                        *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                    DBMA     *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCSETCR *)
```

```
(* %INCLUDE SCSETUP *)
(**)
  PROCEDURE SCSETUP(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR;
                    VAR SET_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS : LISTKEY;
                    VAR DELETE_LIST : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE A SET.             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*     IRC               O    RETURN CODE                           *)
(*     TRANS_STACK       I/O  POINTS TO THE TRANSACTION STACK       *)
(*     SET_KEY           I/O  KEY OF THE SET TO BE UPDATED          *)
(*     NUMBER_OF_USERS   I/O  THE NUMBER OF USERS OF THE LIST       *)
(*     DELETE_LIST       I/O  LIST OF ENTITIES TO BE DELETED        *)
(*     NEW_KEYS_LIST     I/O  LIST OF NEWLY CREATED ENTITIES        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE CURRENT SET DATA IS DISPLAYED ON THE UPSET PANEL.      *)
(*        THE DATA CAN THEN BE UPDATED BY THE USER.                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE         *)
(*     NARATION ON THE NEXT LINE.                                    *)
(*                                                                   *)
```

```
(*    REVISED: MM/DD/YY CCXX      I. M. APERSON                GROUP_ID *)
(*     DESCRIPTION OF FIRST CHANGE MADE.                                 *)
(*                                                                       *)
(*    ORIGINATED: 08/13/87        C. H. MOHME                  DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE SCSETUP *)
```

```
(* %INCLUDE SCSTCUP *)
(**)
  PROCEDURE SCSTCUP(VAR IRC           : RET_REC;
                    VAR TRANS_STACK   : TRANSPTR;
                    VAR STRUCTURE_KEY : ENTKEY;
                    VAR DELETE_LIST   : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
       SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE STRUCTURE           *)
(*        ENTITY.                                                         *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        TRANS_STACK   I/O  POINTS TO THE TRANSACTION STACK              *)
(*        STRUCTURE_KEY I/O  KEY OF THE ENTITY TO BE UPDATED              *)
(*        DELETE_LIST   I/O  LIST OF ENTITIES TO BE DELETED               *)
(*        NEW_KEYS_LIST I/O  LIST OF NEWLY CREATED ENTITIES               *)
(*        IRC            O   RETURN CODE                                  *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THIS ROUTINE UPDATES THE FIELDS IN A STRUCTURE.  FIELDS         *)
(*        CAN BE ADDED, REMOVED, REVIEWED, AND UPDATED.  IF ANY           *)
(*        CHANGES WERE MADE TO THE STRUCTURE ENTITY'S CONSTITUENTS        *)
(*        THE OLD STRUCTURE ENTITY'S KEY IS PLACED ON THE DELETE LIST     *)
(*        AND A NEW STRUCTURE ENTITY IS MODELED AND ITS KEY IS PLACED     *)
(*        ON THE NEW KEYS LIST.                                           *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE SCSTGCR *)
(**)
  PROCEDURE SCSTGCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NECESSARY TO CREATE THE      *)
(*        STRING ENTITY AND PUSHES THE DATA ON THE TRANSACTION       *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   THE RETURN CODE                       *)
(*        TRANS_STACK    I/O  THE TRANSACTION STACK                 *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OPTION OR       *)
(*                            THE RETURN OPTION IS CHOSEN WITHIN     *)
(*                            ANOTHER ROUTINE                        *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE CRSTRING     *)
(*        TO DISPLAY THE PANEL.  THE DATA ENTERED ON THE PANEL IS    *)
(*        VERIFIED AND IF IT IS OK IT IS PUSHED ON THE TRANSACTION   *)
(*        STACK.  OTHERWISE, IF THE DATA ENTERED IS INVALID THE      *)
(*        PANEL IS REDISPLAYED.                                      *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCSTGUP *)
(**)
  PROCEDURE SCSTGUP(VAR IRC : RET_REC;
                    VAR STRING_KEY : ENTKEY;
                    VAR NUMBER_OF_USERS : INTEGER;
                    VAR DELETE_LIST : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE UPDATES THE STRING ENTITY.                         *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME             I/O  DESCRIPTION                               *)
(*        ====             ===  ===========                               *)
(*        STRING_KEY       I/O  KEY OF THE ENTITY TO BE UPDATED           *)
(*        NUMBER_OF_USERS   I   NUMBER OF USERS OF THE STRING ENTITY      *)
(*        DELETE_LIST      I/O  LIST OF ENTITIES TO BE DELETED            *)
(*        NEW_KEYS_LIST    I/O  LIST OF NEWLY CREATED ENTITIES            *)
(*        IRC               O   RETURN CODE                               *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        THIS ROUTINE DISPLAYS THE LENGTH OF THE STRING ENTITY IN        *)
(*        BYTES AND ALLOWS THE USER TO CHANGE THE INFORMATION IF HE/      *)
(*        SHE SO DESIRES.  IF A CHANGE IS MADE AND THE NUMBER OF          *)
(*        USERS OF THE OLD STRING ENTITY IS ONE OR LESS THEN ITS KEY      *)
(*        IS PLACED ON THE DELETE LIST.  AFTER THE NEW STRING ENTITY      *)
(*        IS CREATED ITS KEY IS PLACED ON THE NEW KEYS LIST.              *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
```

```
(* %INCLUDE SCSUBCR *)
(**)
  PROCEDURE SCSUBCR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA NEEDED TO MODEL THE          *)
(*        SUBSCHEMA ENTITY.                                          *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC             O   INTERNAL RETURN CODE                  *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        INSIDE         I/O  INDICATES IF THE EXIT OR RETURN OPTION *)
(*                            HAS BEEN CHOSEN WITHIN ANOTHER ROUTINE *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THE MENU INTERFACE ROUTINE (CRSUBSCM) IS CALLED TO DISPLAY *)
(*        THE CREATE SUBSCHEMA PANEL.  THE DATA GATHERED IS VERIFIED *)
(*        AND IF IT IS VALID IT IS PUSHED ON TO THE TRANSACTION      *)
(*        STACK OR THE APPROPRIATE ACTION IS TAKEN.  IF ANY OF THE   *)
(*        DATA IS INVALID THE PANEL IS REDISPLAYED WITH AN ERROR     *)
(*        MESSAGE.  THIS PANEL CONTINUES TO BE DISPLAYED UNTIL EXIT  *)
(*        OR NO MORE MEMBERS IS CHOSEN.                              *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCSUBUP *)
(**)
  PROCEDURE SCSUBUP(VAR IRC           : RET_REC;
                    VAR SUBSCHEMA_KEY : ENTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GATHERS THE DATA TO UPDATE THE SUBSCHEMA      *)
(*        ENTITY.                                                    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        SUBSCHEMA_KEY  I   KEY OF THE ENTITY TO BE UPDATED       *)
(*        IRC            O   RETURN CODE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS THE MENU INTERFACE ROUTINES (UPSUB1 AND *)
(*        UPSUB2) WHICH DISPLAY THE DATA DESCRIBING THE SUBSCHEMA.   *)
(*        THE UPDATE SUBSCHEMA OPTIONS INCLUDE :                     *)
(*                   CHANGE THE SUBSCHEMA NAME,                      *)
(*                   UPDATE THE CONSTITUENT LIST BY ADDING/          *)
(*                                          REMOVING ELEMENTS,       *)
(*                   REVIEW A CONSTITUENT,                           *)
(*                   DELETE THE SUBSCHEMA,                           *)
(*                   SAVE THE CHANGES MADE,                          *)
(*                   RETURN AND EXIT.                                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCSUPCR *)
(**)
  PROCEDURE SCSUPCR(VAR IRC           : RET_REC;
                    VAR TRANS_STACK   : TRANSPTR;
                    VAR CREATE_ONLY   : BOOLEAN;
                    VAR BUILD         : BOOLEAN;
                    VAR SUPERTYPE_KEY : ENTKEY);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE GATHERS THE DATA NECESSARY TO MODEL THE      *)
(*         SUPERTYPE ENTITY.                                         *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                           *)
(*         ====          ===  ===========                           *)
(*         IRC            O   RETURN CODE                           *)
(*         TRANS_STACK   I/O  POINTS TO THE TRANSACTION STACK       *)
(*         CREATE_ONLY    I   INDICATES IF SUPERTYPE IS TO BE ONLY   *)
(*                            CREATED OR IF ONE ALREADY EXISTING CAN *)
(*                            BE REFERENCED.                         *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*       REF                                                         *)
(*         INSIDE        I/O  INDICATES IF THE EXIT OR RETURN OPTION*)
(*                            IS CHOSEN WITHIN ANOTHER ROUTINE.      *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINE (CRSUPTYP)   *)
(*         WHICH DISPLAYS THE CREATE SUPERTYPE PANEL.  THE NAME IS   *)
(*         CHECKED FOR US CHECKED FOR UNIQUENESS.  IF THEY ARE UNIQUE*)
(*         THEN THE DATA IS PUSHED ONTO THE TRANSACTION STACK AND    *)
(*         THE ROUTINE (SCFLDCR) IS CALLED TO ENTER THE ENTITY'S     *)
(*         FIELDS.  AFTER ALL OF THE FIELDS HAVE BEEN ENTERED THE    *)
(*         TRANSACTION PROCESSING ROUTINE IS CALLED TO MODEL THE     *)
(*         ENTITY.                                                   *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*    ORIGINATED: 09/28/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCSUPCR *)
```

```
(* %INCLUDE SCSUPUP *)
(**)
  PROCEDURE SCSUPUP(VAR IRC              : RET_REC;
                    VAR TRANS_STACK      : TRANSPTR;
                    VAR SUP_KEY          : ENTKEY;
                    VAR REMOVE_SUPERTYPE : BOOLEAN);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    $FUNCTION:                                                          *)
(*         THIS ROUTINE UPDATES THE SUPERTYPE ENTITY.                     *)
(*                                                                        *)
(*    $DESCRIPTION OF ARGUMENTS:                                          *)
(*         NAME           I/O  DESCRIPTION                               *)
(*         ====           ===  ===========                               *)
(*     IRC                 O   RETURN CODE                               *)
(*     TRANS_STACK        I/O  POINTS TO THE TRANSACTION STACK           *)
(*     SUP_KEY            I/O  KEY OF THE ENTITY TO BE UPDATED           *)
(*     REMOVE_SUPERTYPE   I/O  INDICATES IF SUPERTYPE CAN BE REMOVED     *)
(*                             OR IF ONLY THE REFERENCE TO THE SUPER-     *)
(*                             TYPE CAN BE REMOVED.                       *)
(*                                                                        *)
(*    $COMMONS:                                                           *)
(*         NONE                                                           *)
(*                                                                        *)
(*    $ENVIRONMENT:                                                       *)
(*         LANGUAGE: IBM PASCAL                                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                        *)
(*    $EXECUTION PROCEDURE:                                               *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                    *)
(*                                                                        *)
(*    $PROCESSING DESCRIPTION:                                            *)
(*         THIS ROUTINE CALLS THE MENU INTERFACE ROUTINES TO DISPLAY      *)
(*         THE DATA ABOUT THE SUPERTYPE. THE UPDATE SUPERTYPE OPTIONS     *)
(*         INCLUDE THE FOLLOWING:                                         *)
(*             CHANGE THE SUPERTYPE NAME,                                 *)
(*             UPDATE THE SUPERTYPE'S CONSTITUENTS (FIELDS) BY            *)
(*                ADDING FIELDS, REMOVING FIELDS OR UPDATING FIELDS,      *)
(*             REVIEW A FIELD,                                            *)
(*             DELETE THE SUPERTYPE,                                      *)
(*             SAVE THE CHANGES MADE,                                     *)
(*             RETURN OR EXIT.                                            *)
(*          IF SAVE THE CHANGES WAS SELECTED THE SUPERTYPE IS UPDATED     *)
(*          AND THE SUPERTYPES ON THE DELETE LIST WILL BE DELETED IF      *)
(*          POSSIBLE.  IF RETURN OR EXIT IS SELECTED THE SUPERTYPES ON    *)
(*          THE NEW KEYS LIST WILL BE DELETED IF POSSIBLE.                *)
(*                                                                        *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    ORIGINATED: 09/28/87       C. H. MOHME               DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCSUPUP *)
```

```
(* %INCLUDE SCTRSPR *)
(**)
  PROCEDURE SCTRSPR(VAR IRC : RET_REC;
                    VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE BEGINS THE PROCESSING OF THE TRANSACTION      *)
(*        STACK.                                                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   RETURN CODE                           *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK       *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      REF                                                          *)
(*        CURRENT_LIST      I/O    POINTS TO THE LIST OF KEYS       *)
(*                                 CURRENTLY IN USE                  *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS SCPOPTR TO POP THE TRANSACTION STACK.   *)
(*        THEN EACH TRANSACTION IS PROCESSED ACCORDING TO ITS TYPE   *)
(*        BY CALLING THE APPROPRIATE ROUTINE TO MODEL THE ENTITIES.  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCTYPIN *)
(**)
  PROCEDURE SCTYPIN(VAR IRC           :  RET_REC;
                    VAR INCLD         :  TEXT;
                    VAR SUBSCHEMA_KEY :  ENTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE WRITES THE DEFINED TYPE DECLARATIONS TO THE   *)
(*        PASCAL INCLUDE FILE.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        IRC            I/O  RETURN CODE                           *)
(*        INCLD           I   THE FILE NAME                         *)
(*        SUBSCHEMA_KEY   I   THE KEY TO THE SUBSCHEMA              *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        WRITE A DESCRIPTION OF THE BASIC TYPES AND HOW THEY ARE    *)
(*                                                    IMPLEMENTED    *)
(*        WRITE THE DEFINED TYPES HEADING TO THE FILE                *)
(*        WRITE TYPE TO THE FILE                                     *)
(*        MAKE AN INCLUSIVE LIST OF DEFINED TYPES WITHIN THE         *)
(*          SUBSCHEMA                                                *)
(*        DELETE ANY DUPLICATES ON THE LIST OF DEFINED TYPES         *)
(*        SET THE LIST OF DEFINED TYPES TO BE READ FORWARD           *)
(*        CONTINUE TO READ ITEMS UNTIL THE END OF LIST IS ENCOUNTERED *)
(*          GET THE KEY TO THE NEXT DEFINED TYPE IN THE LIST         *)
(*          GET THIS DEFINED TYPE'S ADB                              *)
(*          GET THE DEFINED TYPE'S CONSTITUENT                       *)
(*          GET THE CONSTITUENT'S ADB                                *)
(*          WRITE TO THE FILE T_<ADB.DEF_TYP_NAME> =                 *)
```

```
(*          CALL THE ROUTINE TO WRITE OUT THE PRIMITIVE TYPES OF      *)
(*            INTEGER                                                 *)
(*            REAL                                                    *)
(*            STRING                                                  *)
(*            LOGICAL                                                 *)
(*            ARRAY                                                   *)
(*            DEFINED TYPE                                            *)
(*            ENUMERATION                                             *)
(*            STRUCTURE                                               *)
(*            POINTER                                                 *)
(*        END                                                         *)
(*                                                                    *)
(*    $COMMENTS:                                                      *)
(*                                                                    *)
(*    $CHANGE CONTROL:                                                *)
(*                                                                    *)
```

```
(* %INCLUDE SCTYPUP *)
(**)
  PROCEDURE SCTYPUP(VAR IRC           : RET_REC;
                    VAR TRANS_STACK   : TRANSPTR;
                    VAR CNST_KEY      : ENTKEY;
                    VAR CREATE        : BOOLEAN;
                    VAR NEWTYPE       : ENTITY_TYPE;
                    VAR DELETE_LIST   : LISTKEY;
                    VAR NEW_KEYS_LIST : LISTKEY);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE UPDATES THE DEFINED TYPE, ARRAY, OR FIELD           *)
(*        ENTITY'S TYPE.                                                   *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK              *)
(*        CNST_KEY       I/O  KEY TO THE TYPE TO BE UPDATED                *)
(*        CREATE          I   INDICATES IF A NEW ENTITY IS TO BE MADE      *)
(*        NEWTYPE         I   ENTITY TYPE TO UPDATE                        *)
(*        DELETE_LIST    I/O  KEY TO LIST OF ENTITIES TO BE DELETED        *)
(*                            IF THE OPTION SAVE THE CHANGED IS CHOSEN     *)
(*        NEW_KEYS_LIST  I/O  KEY TO LIST OF ENTITIES CREATED DURING       *)
(*                            AN UPDATE                                    *)
(*        IRC             O   RETURN CODE                                  *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THIS ROUTINE DETERMINES IF A NEW ENTITY IS TO BE CREATED         *)
(*        OR IF THE OLD ENTITY SHOULD BE UPDATED.  THEN IT CALLS           *)
(*        THE ROUTINE ACCORDING TO THE ENTITY TYPE ENTERED.                *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE SCUNIQUE *)
(**)
  PROCEDURE SCUNIQUE(CONST ENTITY_KEY : ENTKEY;
                     VAR   ADB : ENTBLOCK;
                     VAR   DATA : BLKDATA;
                     VAR   RRC : INTEGER);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE VERIFIES THE UNIQUENESS OF NAMES WITHIN THE   *)
(*        SCHEMA FOR THE SUBSCHEMA, CLASS, ENTITY, GLOBAL FIELD,     *)
(*        FIELD, ENUMERITEM, AND DEFINED TYPE ENTITIES AS WELL AS    *)
(*        THE USER DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES.    *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        ENTITY_KEY     I   THE KEY TO THE ENTITY                 *)
(*        ADB            O   THE ADB INFORMATION                   *)
(*        DATA          I/O  THE USER DEFINED DATA STRUCTURE USED  *)
(*                           TO PASS DATA INTO THIS PROCEDURE AND  *)
(*                           TO GET THE DESIRED OUTPUT FROM THIS   *)
(*                           PROCEDURE                             *)
(*        RRC            O   THE ROUTINE RETURN CODE               *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE IS EXECUTED BY CALLING MAKXEQ FOR A GIVEN     *)
(*        KIND.  THE NAMES AND USER DEFINED KIND NUMBERS ARE         *)
(*        COMPARED TO ONE ANOTHER TO VERIFY THAT THE DATA ENTERED    *)
(*        IS UNIQUE. THE DATA STRUCTURE BLKDATA CONTAINS THE         *)
(*        INFORMATION WHICH INDICATES THE UNIQUENESS.                *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCUNQEST *)
(**)
  PROCEDURE SCUNQEST(VAR IRC : RET_REC;
                     CONST ENT_KIND : INTEGER;
                     VAR UNIQUE_NAME : T_NAME;
                     VAR UNIQUE_NUMBER : INTEGER;
                     VAR FOUND : MATCH);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE CALLS MAKXEQ TO EXECUTE THE PROCEDURE SCUNIQUE *)
(*        WHICH VERIFIES THE UNIQUENESS OF NAMES WITHIN THE SCHEMA   *)
(*        FOR THE SUBSCHEMA, CLASS, ENTITY, GLOBAL FIELD, FIELD,     *)
(*        ENUMERITEM, AND DEFINED TYPE ENTITIES AS WELL AS THE USER  *)
(*        DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES.             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        ENT_KIND        I   THE ENTITY KIND NUMBER                *)
(*        UNIQUE_NAME     I   THE NAME TO BE VERIFIED FOR UNIQUENESS *)
(*        UNIQUE_NUMBER   I   THE NUMBER TO BE VERIFIED FOR UNIQUENESS *)
(*        FOUND           O   A RECORD WHICH INDICATES WHETHER OR    *)
(*                            NOT A MATCH TO A NAME OR KIND NUMBER   *)
(*                            HAS BEEN FOUND                         *)
(*        IRC             O   RETURN CODE                           *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE CALLS MAKXEQ TO EXECUTE THE PROCEDURE         *)
(*        SCUNIQUE WHICH DOES THE ACTUAL COMPARISON FOR UNIQUENESS.  *)
(*        THIS CONTINUES UNTIL A MATCH IS FOUND OR ALL THE ENTITIES  *)
(*        HAVE BEEN CHECKED.                                         *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCUNQPND *)
(**)
  PROCEDURE SCUNQPND(VAR IRC : RET_REC;
                     VAR TRANS_STACK : TRANSPTR;
                     VAR UNIQUE_NAME : T_NAME;
                     VAR UNIQUE_NUMBER : INTEGER;
                     VAR TRANSTYPE : TRANS_TYPE;
                     VAR FLDTYP : T_FIELDTYPE;
                     VAR FOUND : MATCH);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE VERIFIES THE UNIQUENESS OF NAMES AND USER     *)
(*        DEFINED KIND NUMBERS FOR CLASSES AND ENTITIES, BY SEARCHING *)
(*        THE TRANSACTION STACK FOR THE CLASS, ENTITY, FIELD,        *)
(*        ENUMERITEM, AND DEFINED TYPE ENTITIES.                     *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        UNIQUE_NAME     I   THE NAME TO BE VERIFIED FOR UNIQUENESS *)
(*        UNIQUE_NUMBER   I   THE NUMBER TO BE VERIFIED FOR UNIQUENESS *)
(*        TRANS_STACK     I   POINTER TO THE TRANSACTION STACK       *)
(*        TRANSTYPE       I   INDICATES THE TYPE OF TRANSACTION      *)
(*        FIELDTYP        I   INDICATES FOR THE FIELD ENTITY WHAT TYPE *)
(*                            OF FIELD IT IS                         *)
(*        FOUND           O   A RECORD WHICH INDICATES WHETHER OR NOT *)
(*                            A MATCH BEEN FOUND                     *)
(*        IRC             O   RETURN CODE                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE SEARCHES THROUGH THE TRANSACTION STACK AND    *)
(*        COMPARES THE NAME ENTERED TO THOSE TRANSACTIONS WHICH      *)
(*        MUST HAVE UNIQUE NAMES WITHIN THE SCHEMA.  FIELD ENTITY    *)
(*        NAMES ARE CHECKED FOR UNIQUENESS AMONG THOSE FIELDS WITHIN *)
(*        THE ENTITY BEING CREATED. THIS CONTINUES UNTIL A MATCH IS  *)
(*        FOUND OR THE END OF THE STACK IS ENCOUNTERED.              *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE SCUPDATE *)
(**)
  PROCEDURE SCUPDATE(VAR IRC         : RET_REC;
                     VAR TRANS_STACK : TRANSPTR);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE DETERMINES THE NEXT MENU TO DISPLAY FROM THE *)
(*         UPDATE OPTION CHOSEN.                                     *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME           I/O  DESCRIPTION                          *)
(*         ====           ===  ===========                          *)
(*         IRC             O   RETURN CODE                          *)
(*         TRANS_STACK    I/O  POINTS TO THE TRANSACTION STACK      *)
(*                                                                   *)
(*    $COMMONS:                                                      *)
(*         DEF                                                       *)
(*            INSIDE      I/O  INDICATES IF THE EXIT OPTION HAS     *)
(*                             BEEN CHOSEN WITHIN ANOTHER CREATE     *)
(*                             PROCEDURE                             *)
(*                                                                   *)
(*    $ENVIRONMENT:                                                  *)
(*         LANGUAGE: IBM PASCAL                                      *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                   *)
(*    $EXECUTION PROCEDURE:                                          *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                   *)
(*    $PROCESSING DESCRIPTION:                                       *)
(*         CALLS THE UPDATE MENU INTERFACE ROUTINE (MUPDATE) AND     *)
(*         PROCESSES THE DATA RECIEVED FROM THE MENU EITHER BY       *)
(*         CALLING THE APPROPRIATE ROUTINE OR EXITING THE PROCEDURE. *)
(*                                                                   *)
(*    $COMMENTS:                                                     *)
(*                                                                   *)
(*    $CHANGE CONTROL:                                               *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                   *)
(*     REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE        *)
(*      NARATION ON THE NEXT LINE.                                   *)
(*                                                                   *)
```

```
(*    REVISED: 09/28/87          C. H. MOHME                    DBMA     *)
(*     INCORPORATED THE SUPERTYPE DATA TYPE.                              *)
(*                                                                        *)
(*    ORIGINATED: 08/13/87       C. H. MOHME                    DBMA     *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE SCUPDATE *)
```

```
(* %INCLUDE SORTKIND *)
(**)
  PROCEDURE SORTKIND(CONST CURRENT : ENTBLOCK;
                     CONST NEXT    : ENTBLOCK;
                     VAR   FLIP    : BOOLEAN;
                     VAR   RRC     : EXT_RET_CODE;
                     VAR   PROC    : ROUTINE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*    $FUNCTION:                                                     *)
(*         THIS ROUTINE IS THE ORDER FUNCTION CALLED BY MALSRT       *)
(*                                                                   *)
(*    $DESCRIPTION OF ARGUMENTS:                                     *)
(*         NAME          I/O  DESCRIPTION                          *)
(*         ====          ===  ===========                          *)
(*         CURRENT        I   THE ADB OF THE CURRENT ENTITY        *)
(*         NEXT           I   THE ADB OF THE NEXT ENTITY           *)
(*         FLIP           O   INDICATES IF THE ENTITIES SHOULD BE  *)
(*                            FLIPPED                              *)
(*         RRC            O   THE ROUTINE'S RETURN CODE            *)
(*                            = 0 OK                               *)
(*                            <> 0 ERROR                           *)
(*         XRC            O   EXTERNAL RETURN CODE FROM MAS        *)
(*                            = 0  OK                              *)
(*                            >= 10  ERROR                         *)
(*                                                                 *)
(*    $COMMONS:                                                    *)
(*       NONE                                                      *)
(*                                                                 *)
(*    $ENVIRONMENT:                                                *)
(*         LANGUAGE: IBM PASCAL                                    *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                  *)
(*                                                                 *)
(*    $EXECUTION PROCEDURE:                                        *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE             *)
(*                                                                 *)
(*    $PROCESSING DESCRIPTION:                                     *)
(*         THIS ROUTINE IS CALLED BY THE MAS ROUTINE MALSRT.  THE  *)
(*         TWO ENTITIES ARE COMPARED AND IF THEY ARE OUT OF ALPHA- *)
(*         BETICAL ORDER THE FLIP FLAG IS SET TO TRUE OTHERWISE THE*)
(*         FLAG REMAINS FALSE.  IF THE FLAG IS TRUE THE ENTITIES ARE*)
(*         SWAPPED OTHERWISE THEY ARE NOT.                         *)
(*                                                                 *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED:  3/06/87       M. H. CHOI                  DBMA      *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE SORTKIND *)
```

```
(* %INCLUDE SORTNAME *)
(**)
  PROCEDURE SORTNAME(CONST CURRENT : ENTBLOCK;
                     CONST NEXT    : ENTBLOCK;
                     VAR   FLIP    : BOOLEAN;
                     VAR   RRC     : EXT_RET_CODE;
                     VAR   PROC    : ROUTINE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE IS THE ORDER FUNCTION CALLED BY MALSRT        *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        CURRENT        I   THE ADB OF THE CURRENT ENTITY         *)
(*        NEXT           I   THE ADB OF THE NEXT ENTITY            *)
(*        FLIP           O   INDICATES IF THE ENTITIES SHOULD BE   *)
(*                           FLIPPED                                 *)
(*        RRC            O   THE ROUTINE'S RETURN CODE             *)
(*                           = 0 OK                                  *)
(*                           <> 0 ERROR                              *)
(*        XRC            O   EXTERNAL RETURN CODE FROM MAS         *)
(*                           = 0  OK                                 *)
(*                           >= 10  ERROR                            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        THIS ROUTINE IS CALLED BY THE MAS ROUTINE MALSRT.  THE     *)
(*        TWO ENTITIES ARE COMPARED AND IF THEY ARE OUT OF ALPHA-    *)
(*        BETICAL ORDER THE FLIP FLAG IS SET TO TRUE OTHERWISE THE   *)
(*        FLAG REMAINS FALSE.  IF THE FLAG IS TRUE THE ENTITIES ARE  *)
(*        SWAPPED OTHERWISE THEY ARE NOT.                            *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED:  3/10/87       M. H. CHOI                  DBMA       *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE SORTNAME *)
```

```
(* %INCLUDE UPARRAY *)
(**)
  PROCEDURE UPARRAY(VAR MESS     : MESSAGE;
                    VAR LBND     : CHAR8;
                    VAR HBND     : CHAR8;
                    VAR ATYPE    : ENTITY_TYPE;
                    VAR ULBD     : CHAR8;
                    VAR UHBD     : CHAR8;
                    VAR UTYPE    : ENTITY_TYPE;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE UPDATE ARRAY MENU                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        LBND           I   THE LOWER BOUND OF THE ARRAY            *)
(*        HBND           I   THE UPPER BOUND OF THE ARRAY            *)
(*        ATYPE          I   THE ARRAY TYPE                          *)
(*        ULBD           O   THE UPDATED LOWER BOUND OF THE ARRAY    *)
(*        UHBD           O   THE UPDATED HIGHER BOUND OF THE ARRAY   *)
(*        UTYPE          O   THE UPDATED ARRAY TYPE                  *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE UPDATE ARRAY PANEL (UPARRAY) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED   *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED       *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.      *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*        NONE                                                         *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
```

```
(* %INCLUDE UPCLASS1 *)
(**)
  PROCEDURE UPCLASS1(VAR MESS    : MESSAGE;
                     VAR NAME    : T_NAME;
                     VAR KNUM    : CHAR8;
                     VAR UNAM    : T_NAME;
                     VAR UNUM    : CHAR8;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*              THIS PROCEDURE :                                           *)
(*                   DISPLAYS THE UPDATE CLASS MENU 1                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        KNUM           I   THE ENTITY KIND NUMBER                       *)
(*        NAME           I   THE ENTITY NAME                              *)
(*        UNAM           O   THE UPDATED ENTITY NAME                      *)
(*        UNUM           O   THE UPDATED ENTITY NUMBER                    *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE CLASS PANEL NUMBER ONE (UPCLASS1) BY          *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO       *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION      *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-       *)
(*        CEDURE.                                                          *)
(*                                                                         *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE UPCLASS2 *)
(**)
  PROCEDURE UPCLASS2(VAR MESS       : MESSAGE;
                     VAR NAME       : T_NAME;
                     VAR NUM        : CHAR8;
                     VAR GROUP      : T_ARRAY23;
                     VAR ARRAY_SIZE : INTEGER;
                     VAR MEMBER     : T_NAME;
                     VAR NEXT_OP    : OPERATIONS;
                     VAR RR         : RET_REC);
      SUBPROGRAM;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*              THIS PROCEDURE :                                      *)
(*                   DISPLAYS THE UPDATE CLASS MENU 2                 *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME           I/O  DESCRIPTION                             *)
(*        ====           ===  ===========                             *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        MEMBER          O   THE MEMBER SELECTED TO BE UPDATED       *)
(*        GROUP           I   THE ARRAY OF MEMBERS TO SELECT FROM     *)
(*        ARRAY_SIZE      I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                            OPERATION                               *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*      NONE                                                          *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*        DDNAMES USED WITH STANDARD FILES:                           *)
(*             NONE                                                   *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                     *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*        DISPLAY THE UPDATE CLASS PANEL NUMBER TWO (UPCLASS2) BY     *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO  *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-  *)
(*        CEDURE.                                                     *)
(*                                                                    *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE UPDEFTYP *)
(**)
  PROCEDURE UPDEFTYP(VAR MESS    : MESSAGE;
                     VAR NAME    : CHAR16;
                     VAR FTYPE   : ENTITY_TYPE;
                     VAR UNAM    : CHAR16;
                     VAR UTYPE   : ENTITY_TYPE;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE UPDATE DEFINED TYPE MENU               *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NAME           I   THE NAME OF THE DEFINED TYPE                 *)
(*        FTYPE          I   THE TYPE OF THE DEFINED TYPE                 *)
(*        UNAM           O   THE UPDATED NAME OF THE DEFINED TYPE         *)
(*        FTYPE          I   THE UPDATED TYPE OF THE DEFINED TYPE         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE CREATE DEFINED TYPE PANEL (UPDEFTYP) BY MAKING       *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN           *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION         *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-       *)
(*        CEDURE.                                                          *)
(*                                                                         *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE UPENTY1 *)
(**)
  PROCEDURE UPENTY1(VAR MESS     : MESSAGE;
                    VAR NAME     : T_NAME;
                    VAR KNUM     : CHAR8;
                    VAR UNAM     : T_NAME;
                    VAR UNUM     : CHAR8;
                    VAR NEXT_OP  : OPERATIONS;
                    VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*              THIS PROCEDURE :                                           *)
(*                   DISPLAYS THE UPDATE ENTITY MENU 1                     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        NAME            I   THE ENTITY NAME                              *)
(*        KNUM            I   THE ENTITY KIND NUMBER                       *)
(*        UNAM            O   THE UPDATED ENTITY NAME                      *)
(*        UNUM            O   THE UPDATED ENTITY KIND NUMBER               *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                            OPERATION                                    *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE ENTITY PANEL NUMBER ONE (UPENTY1) BY          *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO       *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION      *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-       *)
(*        CEDURE.                                                          *)
(*                                                                         *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE UPENTY2 *)
(**)
PROCEDURE UPENTY2(VAR MESS      : MESSAGE;
                  VAR NAME      : T_NAME;
                  VAR KNUM      : CHAR8;
                  VAR MEM_ARRAY : T_ARRAY16;
                  VAR SIZE      : INTEGER;
                  VAR MEMBER    : T_NAME;
                  VAR NEXT_OP   : OPERATIONS;
                  VAR RR        : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*            THIS PROCEDURE :                                       *)
(*                 DISPLAYS THE UPDATE ENTITY MENU 2                 *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ==========                              *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE ENTITY NAME                         *)
(*        KNUM           I   THE ENTITY KIND NUMBER                  *)
(*        MEM_ARRAY      I   THE ARRAY OF MEMBERS TO SELECT FROM     *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        MEMBER         O   THE MEMBER SELECTED                     *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
```

```
(*   $PROCESSING DESCRIPTION:                                          *)
(*        DISPLAY THE UPDATE ENTITY PANEL NUMBER TWO (UPENTY2) BY      *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO   *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION  *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO-   *)
(*        CEDURE.                                                      *)
(*                                                                     *)
(*   $COMMENTS:                                                        *)
(*        NONE                                                         *)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*                                                                     *)
```

```
(* %INCLUDE UPENUM *)
(**)
PROCEDURE UPENUM(VAR MESS    : MESSAGE;
                 VAR MEMBERS : T_ARRAY16;
                 VAR SIZE    : INTEGER;
                 VAR NAME    : T_NAME;
                 VAR NEXT_OP : OPERATIONS;
                 VAR RR      : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                    DISPLAYS THE UPDATE ENUMERATION MENU                 *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MEMBERS        I   THE ARRAY OF MEMBERS TO DISPLAY              *)
(*        SIZE           I   THE SIZE OF THE ARRAY OF NUMBERS             *)
(*        NAME           O   THE MEMBER NAME ENTERED                      *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISLAY THE REVIEW ENUMERATION MENU (UPENUM) BY MAKING            *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN           *)
(*        ENUMERATED TYPE.                                                 *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE UPFIELD *)
(**)

PROCEDURE UPFIELD(VAR MESS     : MESSAGE;
                  VAR NAME     : T_NAME;
                  VAR POS      : CHAR8;
                  VAR PURP     : CHAR8;
                  VAR REQD     : CHAR8;
                  VAR DEPD     : CHAR12;
                  VAR FTYP     : ENTITY_TYPE;
                  VAR FLD_TYP  : T_FIELDTYPE;
                  VAR UNAM     : T_NAME;
                  VAR UPOS     : CHAR8;
                  VAR UPUR     : CHAR8;
                  VAR UREQ     : CHAR8;
                  VAR UDEP     : CHAR8;
                  VAR U_TYP    : ENTITY_TYPE;
                  VAR NEXT_OP  : OPERATIONS;
                  VAR RR       : RET_REC);

SUBPROGRAM;

(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS PROCEDURE :                                                 *)
(*                         DISPLAYS THE UPDATE FIELD PANEL                 *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL    *)
(*        NAME            I   THE NAME OF THE FIELD                       *)
(*        PURP            I   THE PURPOSE OF THE FIELD                    *)
(*        REQD            I   THE REQUIREDNESS OF THE FIELD               *)
(*        DEPD            I   THE DEPENDENCE/INDEPENDENCE OF THE FIELD    *)
(*                            THE FIELD                                    *)
(*        FTYP            I   THE TYPE OF THE FIELD                       *)
(*        FLD_TYP         I   THE TYPE OF FIELD                           *)
(*        UNAM            O   THE UPDATED NAME OF THE FIELD               *)
(*        UPUR            O   THE UPDATED PURPOSE OF THE FIELD            *)
(*        UREQ            O   THE UPDATED REQUIREDNESS OF THE FIELD       *)
(*        UDEP            O   THE UPDATED DEPENDENCE/INDEPENDENCE OF      *)
(*        UTYP            O   THE UPDATED TYPE OF THE FIELD               *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT         *)
(*                            OPERATION                                    *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,     *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
```

```
(*   $COMMONS:                                                          *)
(*      NONE                                                            *)
(*                                                                      *)
(*   $ENVIRONMENT:                                                      *)
(*        LANGUAGE: IBM PASCAL                                          *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                        *)
(*        DDNAMES USED WITH STANDARD FILES:                             *)
(*             NONE                                                     *)
(*                                                                      *)
(*   $EXECUTION PROCEDURE:                                              *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*        DISPLAY THE UPDATE FIELD PANEL (UPFIELD) BY MAKING ISPLNK     *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED    *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED        *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.       *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*        NONE                                                          *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
```

```
(* %INCLUDE UPINT *)
(**)
  PROCEDURE UPINT(VAR MESS     : MESSAGE;
                  VAR PREC     : CHAR8;
                  VAR UPREC    : CHAR8;
                  VAR NEXT_OP  : OPERATIONS;
                  VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE UPDATE INTEGER MENU                    *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        PREC           I   THE PRECISION OF THE INTEGER                 *)
(*        UPREC          O   THE UPDATED PRECISION OF THE INTEGER         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE INTEGER PANEL (UPINT) BY MAKING ISPLNK        *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED           *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.          *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE UPLIST *)
(**)
  PROCEDURE UPLIST(VAR MESS     : MESSAGE;
                   VAR MIN      : CHAR8;
                   VAR MAX      : CHAR8;
                   VAR ATYPE    : ENTITY_TYPE;
                   VAR UMIN     : CHAR8;
                   VAR UMAX     : CHAR8;
                   VAR UTYPE    : ENTITY_TYPE;
                   VAR NEXT_OP  : OPERATIONS;
                   VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                     DISPLAYS THE UPDATE LIST MENU                       *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MIN            I   THE MINIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        MAX            I   THE MAXIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           LIST                                          *)
(*        ATYPE          I   THE LIST TYPE                                *)
(*        UMIN           O   THE UPDATED MINIMUM NUMBER OF OCCUR-         *)
(*                           RENCES IN THE LIST                            *)
(*        UMAX           O   THE UPDATED MAXIMUM NUMBER OF OCCUR-         *)
(*                           RENCES IN THE LIST                            *)
(*        UTYPE          O   THE UPDATED LIST TYPE                        *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
```

```
(*   $EXECUTION PROCEDURE:                                              *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                       *)
(*                                                                      *)
(*   $PROCESSING DESCRIPTION:                                           *)
(*        DISPLAY THE UPDATE LIST PANEL (UPLIST) BY MAKING ISPLNK       *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED    *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED        *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.       *)
(*                                                                      *)
(*   $COMMENTS:                                                         *)
(*        NONE                                                          *)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*                                                                      *)
(*    REVISED: MM/DD/YY CCRR      I. M. THECHANGER           GROUP_ID   *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                      *)
(*    REVISED: MM/DD/YY CCZZ      I. M. THEPROGRAMMER        GROUP_ID   *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                      *)
(*    REVISED: MM/DD/YY           I. M. APERSON              GROUP_ID   *)
(*     A DESCRIPTION OF THE CHANGE MADE.                                *)
(*                                                                      *)
(*    ORIGINATED: 08/13/87        C. H. MOHME                DBMA       *)
(*                                                                      *)
(*----------------------------------------------------------------------*)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE UPLIST *)
```

```
(* %INCLUDE UPPNTR *)
(**)
  PROCEDURE UPPNTR(VAR MESS       : MESSAGE;
                   VAR GROUP      : T_ARRAY23;
                   VAR ARRAY_SIZE : INTEGER;
                   VAR MEMBER     : T_NAME;
                   VAR NEXT_OP    : OPERATIONS;
                   VAR RR         : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*              THIS PROCEDURE :                                           *)
(*                   DISPLAYS THE UPDATE POINTER MENU                      *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL    *)
(*        GROUP           I   THE ARRAY OF MEMBERS TO SELECT FROM         *)
(*        ARRAY_SIZE      I   THE SIZE OF THE ARRAY OF MEMBERS            *)
(*        MEMBER          O   THE MEMBER SELECTED TO BE UPDATED           *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT         *)
(*                            OPERATION                                    *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND,     *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE POINTER PANEL (UPPNTR) BY MAKING ISPLNK       *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.                                                            *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE UPREAL *)
(**)
  PROCEDURE UPREAL(VAR MESS     : MESSAGE;
                   VAR PREC     : CHAR8;
                   VAR UPREC    : CHAR8;
                   VAR NEXT_OP  : OPERATIONS;
                   VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE UPDATE REAL MENU                  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        PREC           I   THE REAL SIZE IN DECIMAL DIGIT FORM      *)
(*        UPREC          O   THE UPDATED REAL SIZE IN DECIMAL DIGIT   *)
(*                           FORM                                    *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT      *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,  *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN  *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE UPDATE REAL PANEL (UPREAL) BY MAKING ISPLNK    *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED     *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.    *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE UPSET *)
(**)
  PROCEDURE UPSET(VAR MESS     : MESSAGE;
                  VAR MIN      : CHAR8;
                  VAR MAX      : CHAR8;
                  VAR ATYPE    : ENTITY_TYPE;
                  VAR UMIN     : CHAR8;
                  VAR UMAX     : CHAR8;
                  VAR UTYPE    : ENTITY_TYPE;
                  VAR NEXT_OP  : OPERATIONS;
                  VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*      THIS FUNCTION:                                                    *)
(*                    DISPLAYS THE UPDATE SET MENU                        *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME          I/O  DESCRIPTION                                  *)
(*        ====          ===  ===========                                  *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL     *)
(*        MIN            I   THE MINIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           SET                                          *)
(*        MAX            I   THE MAXIMUM NUMBER OF OCCURRENCES IN THE     *)
(*                           SET                                          *)
(*        ATYPE          I   THE SET TYPE                                 *)
(*        UMIN           O   THE UPDATED MINIMUM NUMBER OF OCCUR-         *)
(*                           RENCES IN THE SET                            *)
(*        UMAX           O   THE UPDATED MAXIMUM NUMBER OF OCCUR-         *)
(*                           RENCES IN THE SET                            *)
(*        UTYPE          O   THE UPDATED SET TYPE                         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                    *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN      *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*      NONE                                                              *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*        DDNAMES USED WITH STANDARD FILES:                               *)
(*             NONE                                                       *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                        *)
```

```
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE SET PANEL (UPSET) BY MAKING ISPLNK            *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED       *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED           *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.          *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCRR      I. M. THECHANGER            GROUP_ID     *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                  *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCZZ      I. M. THEPROGRAMMER         GROUP_ID     *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE               *)
(*     NARATION ON THE NEXT LINE.                                          *)
(*                                                                         *)
(*    REVISED: MM/DD/YY           I. M. APERSON               GROUP_ID     *)
(*     A DESCRIPTION OF THE CHANGE MADE.                                   *)
(*                                                                         *)
(*    ORIGINATED: 08/13/87        C. H. MOHME                 DBMA         *)
(*                                                                         *)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*END--------------------------------------------------------------------*)
(* END %INCLUDE UPSET *)
```

```
(* %INCLUDE UPSTRING *)
(**)
  PROCEDURE UPSTRING(VAR MESS     : MESSAGE;
                     VAR LEN      : CHAR8;
                     VAR ULEN     : CHAR8;
                     VAR NEXT_OP  : OPERATIONS;
                     VAR RR       : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      THIS FUNCTION:                                               *)
(*                    DISPLAYS THE UPDATE STRING MENU                *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        LEN            I   THE LENGTH OF THE STRING IN BYTES       *)
(*        ULEN           O   THE UPDATED LENGTH OF THE STRING IN     *)
(*                           BYTES                                   *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE UPDATE STRING PANEL (UPSTRING) BY MAKING ISPLNK *)
(*        CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN ENUMERATED  *)
(*        TYPE.  THIS DATA AS WELL AS OTHER INFORMATION GATHERED      *)
(*        FROM THE PANEL IS PASSED BACK TO THE CALLING PROCEDURE.     *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(* %INCLUDE UPSTRUC *)
(**)
  PROCEDURE UPSTRUC(VAR MESS       : MESSAGE;
                    VAR CLAS       : T_ARRAY16;
                    VAR ARRAY_SIZE : INTEGER;
                    VAR MEMBER     : T_NAME;
                    VAR NEXT_OP    : OPERATIONS;
                    VAR RR         : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*    $FUNCTION:                                                           *)
(*              THIS PROCEDURE :                                           *)
(*                   DISPLAYS THE UPDATE STRUCTURE MENU                    *)
(*                   RECEIVES THE NAME OF A STRUCTURE TO BE UPDATED        *)
(*                                                                         *)
(*    $DESCRIPTION OF ARGUMENTS:                                           *)
(*         NAME          I/O  DESCRIPTION                                  *)
(*         ====          ===  ===========                                  *)
(*         MESS           I   THE ERROR MESSAGE RECEIVED FROM MAINLINE     *)
(*         CLAS           I   THE ARRAY OF FIELDS                          *)
(*         ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS             *)
(*         MEMBER         O   THE MEMBER SELECTED                          *)
(*         NEXT_OP        O   TELLS THE MAINLINE WHAT PANEL TO CALL        *)
(*                            NEXT                                         *)
(*         RR             O   TELLS THE MAINLINE IF THERE IS AN ERROR      *)
(*                            AND IN WHAT ROUTINE IT OCCURS                *)
(*                                                                         *)
(*    $COMMONS:                                                            *)
(*       NONE                                                              *)
(*                                                                         *)
(*    $ENVIRONMENT:                                                        *)
(*         LANGUAGE: IBM PASCAL                                            *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*         DDNAMES USED WITH STANDARD FILES:                               *)
(*              NONE                                                       *)
(*                                                                         *)
(*    $EXECUTION PROCEDURE:                                                *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                         *)
(*                                                                         *)
(*    $PROCESSING DESCRIPTION:                                             *)
(*         DISPLAY THE UPDATE STRUCTURE PANEL (UPSTRUC) BY MAKING          *)
(*         ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN          *)
(*         ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION        *)
(*         GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING           *)
(*         PROCEDURE.                                                      *)
(*                                                                         *)
(*    $COMMENTS:                                                           *)
(*                                                                         *)
(*    $CHANGE CONTROL:                                                     *)
(*                                                                         *)
```

```
(* %INCLUDE UPSUB1 *)
(**)
  PROCEDURE UPSUB1(VAR MESS    : MESSAGE;
                   VAR NAME    : T_NAME;
                   VAR UNAM    : T_NAME;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*      THIS FUNCTION:                                                     *)
(*                    DISPLAYS THE UPDATE SUBSCHEMA PANEL 1                *)
(*                    INPUT OF THE NAME OF A SUBSCHEMA                     *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME          I/O  DESCRIPTION                                   *)
(*        ====          ===  ===========                                   *)
(*        MESS           I   THE ERROR MESSAGE RECEIVED FROM MAINLINE     *)
(*        NAME           I   THE SUBSCHEMA NAME                           *)
(*        UNAM           O   THE UPDATED SUBSCHEMA NAME                   *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT          *)
(*                           OPERATION                                     *)
(*        XRC            O   INDICATES IF AN ERROR HAS OCCURRED AND,      *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN       *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*        DDNAMES USED WITH STANDARD FILES:                                *)
(*             NONE                                                        *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                          *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        DISPLAY THE UPDATE SUBSCHEMA PANEL (UPSUB1) BY MAKING            *)
(*        ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN           *)
(*        ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION         *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING            *)
(*        PROCEDURE.                                                       *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
```

```
(* %INCLUDE UPSUB2 *)
(**)
  PROCEDURE UPSUB2(VAR MESS       : MESSAGE;
                   VAR NAME       : T_NAME;
                   VAR MEMBERS    : T_ARRAY23;
                   VAR ARRAY_SIZE : INTEGER;
                   VAR MEMBER     : T_NAME;
                   VAR NEXT_OP    : OPERATIONS;
                   VAR RR         : RET_REC);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    $FUNCTION:                                                          *)
(*       THIS FUNCTION:                                                   *)
(*                     DISPLAYS THE REVIEW SUBSCHEMA PANEL 2              *)
(*                                                                        *)
(*    $DESCRIPTION OF ARGUMENTS:                                          *)
(*         NAME          I/O  DESCRIPTION                                 *)
(*         ====          ===  ===========                                 *)
(*         MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL   *)
(*         NAME           I   THE NAME OF THE SUBSCHEMA TO BE UPDATED    *)
(*         MEMBERS        I   THE ARRAY OF MEMBERS TO SELECT FROM        *)
(*         ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS           *)
(*         MEMBER         O   THE MEMBER SELECTED                        *)
(*         NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT        *)
(*                            OPERATION                                   *)
(*         RR             O   INDICATES IF AN ERROR HAS OCCURRED AND,    *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN     *)
(*                                                                        *)
(*    $COMMONS:                                                           *)
(*       NONE                                                             *)
(*                                                                        *)
(*    $ENVIRONMENT:                                                       *)
(*         LANGUAGE: IBM PASCAL                                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*         DDNAMES USED WITH STANDARD FILES:                              *)
(*              NONE                                                      *)
(*                                                                        *)
(*    $EXECUTION PROCEDURE:                                               *)
(*         SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                        *)
(*                                                                        *)
(*    $PROCESSING DESCRIPTION:                                            *)
(*         DISPLAY THE UPDATE SUBSCHEMA PANEL NUMBER TWO BY MAKING        *)
(*         ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO AN         *)
(*         ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION       *)
(*         GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING          *)
(*         PROCEDURE.                                                     *)
(*                                                                        *)
```

```
(*    $COMMENTS:                                                    *)
(*         NONE                                                     *)
(*                                                                  *)
(*    $CHANGE CONTROL:                                              *)
(*                                                                  *)
```

```
(* %INCLUDE UPSUPER *)
(**)
  PROCEDURE UPSUPER(VAR MESS             : MESSAGE;
                    VAR NAME             : T_NAME;
                    VAR UNAM             : T_NAME;
                    VAR REMOVE_SUPERTYPE : BOOLEAN;
                    VAR NEXT_OP          : OPERATIONS;
                    VAR RR               : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*             THIS PROCEDURE :                                      *)
(*                  DISPLAYS THE UPDATE SUPERTYPE MENU               *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE ENTITY NAME                         *)
(*        KNUM           I   THE ENTITY KIND NUMBER                  *)
(*        UNAM           O   THE UPDATED ENTITY NAME                 *)
(*        UNUM           O   THE UPDATED ENTITY KIND NUMBER          *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*        DISPLAY THE UPDATE ENTITY PANEL NUMBER ONE (UPSUPER) BY    *)
(*        MAKING ISPLNK CALLS.  THE OPTION CHOSEN IS TRANSLATED INTO *)
(*        AN ENUMERATED TYPE.  THIS DATA AS WELL AS OTHER INFORMATION *)
(*        GATHERED FROM THE PANEL IS PASSED BACK TO THE CALLING PRO- *)
(*        CEDURE.                                                    *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                *)
(*                                                                       *)
(*    ORIGINATED: 09/28/87       C. H. MOHME                 DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE UPSUPER *)
```

```
(* %INCLUDE XATTDATA *)
(**)
  PROCEDURE XATTDATA(VAR IRC      : RET_REC;
                     VAR XREFFILE : TEXT);
       SUBPROGRAM;
(**)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*   $FUNCTION:                                                       *)
(*        THIS ROUTINE GENERATES A LIST OF ALL "ENTITIES" CONTAINING  *)
(*        A PARTICULAR "ENTITY."                                      *)
(*                                                                    *)
(*   $DESCRIPTION OF ARGUMENTS:                                       *)
(*        NAME          I/O  DESCRIPTION                            *)
(*        ====          ===  ===========                            *)
(*        IRC            O   RETURN CODE                            *)
(*        XREFFILE       O   CROSS REFERENCE REPORT FILE            *)
(*                                                                    *)
(*   $COMMONS:                                                        *)
(*        NONE                                                        *)
(*                                                                    *)
(*   $ENVIRONMENT:                                                    *)
(*        LANGUAGE: IBM PASCAL                                        *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                      *)
(*                                                                    *)
(*   $EXECUTION PROCEDURE:                                            *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                 *)
(*                                                                    *)
(*   $PROCESSING DESCRIPTION:                                         *)
(*      DISPLAY THE CROSS REFERENCE MENU.                             *)
(*      DETERMINE WHICH OPTION WAS CHOSEN AND CALL THE APPROPRIATE    *)
(*        ROUTINE.                                                    *)
(*        INTEGER DATA TYPE.  DISPLAY MENU TO OBTAIN DESIRED          *)
(*          PRECISION.                                                *)
(*        REAL DATA TYPE.  DISPLAY MENU TO OBTAIN DESIRED PRECISION.  *)
(*        STRING DATA TYPE.  DISPLAY MENU TO OBTAIN DESIRED LENGTH.   *)
(*        OTHER DATA TYPES.                                           *)
(*        ARRAY, LIST, OR SET DATA TYPE.                              *)
(*        THE LIST AND SET DATA TYPES ARE IMPLEMENTED AS ARRAYS.      *)
(*          THEREFORE, SPECIAL CHECKING MUST BE PERFORMED.            *)
(*      IF A PRECISION WAS SPECIFIED FOR THE INTEGER, REAL, OR STRING *)
(*        DATA TYPE, THEN SELECT FROM THE DATA TYPE LIST THOSE        *)
(*        ENTITIES WITH THE SPECIFIED PRECISION.                      *)
(*      MAKE A LIST OF THE USERS OF THE DATA TYPE AND CALL THE        *)
(*        ROUTINE TO DISPLAY THE ATTRIBUTES AND USER ENTITIES.        *)
(*                                                                    *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*    ORIGINATED: 11/16/87       C. H. MOHME                 DBMA     *)
(*                                                                       *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XATTDATA *)
```

```
(* %INCLUDE XATTNAME *)
(**)
  PROCEDURE XATTNAME(VAR IRC      : RET_REC;
                     VAR XREFFILE : TEXT);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GENERATES A LIST OF ALL ENTITIES HAVING AN    *)
(*        ATTRIBUTE WITH A SPECIFIED NAME.                           *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        IRC            O   RETURN CODE                           *)
(*        XREFFILE       O   THE CROSS REFERENCE REPORT FILE        *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*      DETERMINE THE ATTRIBUTE NAME.                                *)
(*      DETERMINE IF A VALID IDENTIFIER WAS ENTERED.                 *)
(*        MAKE A LIST OF ATTRIBUTES OF THE SPECIFIED NAME.           *)
(*        MAKE A LIST OF THE ENTITY USERS.                           *)
(*        PREPARE TO DISPLAY THE RESULTS.                            *)
(*        DISPLAY THE RESULTS.                                       *)
(*        RETURN TO MAIN MENU.                                       *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID  *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER       GROUP_ID  *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE         *)
(*     NARATION ON THE NEXT LINE.                                    *)
(*                                                                   *)
(*    ORIGINATED: 11/19/87       C. H. MOHME               DBMA      *)
(*                                                                   *)
(*END----------------------------------------------------------------*)
(* END %INCLUDE XATTNAME *)
```

```
(* %INCLUDE XATTRES *)
(**)
  PROCEDURE XATTRES(VAR IRC              : RET_REC;
                    VAR ATTRIBUTE_LIST   : LISTKEY;
                    VAR DATA_TYPE_KIND   : INTEGER;
                    VAR PRECISION        : INTEGER;
                    VAR COM1             : CHAR50;
                    VAR COM2             : CHAR50;
                    VAR XREFFILE         : TEXT;
                    VAR OPTION           : OPERATIONS);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        DISPLAYS CROSS REFERENCE REPORT RESULTS                          *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        IRC             O   RETURN CODE                                  *)
(*     ATTRIBUTE_LIST     I   THE LIST OF ATTRIBUTES                       *)
(*     DATA_TYPE_KIND     I   THE DATA TYPE KIND TO BE FOUND IN THE        *)
(*                              ATTRIBUTE LIST                             *)
(*       PRECISION        I   THE INTEGER OR REAL PRECISION, OR THE        *)
(*                              STRING LENGTH                              *)
(*        COM1            I   THE MENU COMMENTS                            *)
(*        COM2            I   THE MENU COMMENTS                            *)
(*        XREFFILE       I/O  THE CROSS REFERENCE REPORT FILE              *)
(*        OPTION          O   THE OPTION SELECTED                          *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*        NONE                                                             *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*     READ EACH ATTRIBUTE OFF OF THE LIST OF ATTRIBUTES                   *)
(*     FROM THE ATTRIBUTE ADB OBTAIN THE ATTRIBUTE NAME AND PUT IT         *)
(*       INTO AN ARRAY.                                                    *)
(*     FIND THE LIST OF USERS OF THE ATTRIBUTE.  THIS LIST WILL            *)
(*       ALWAYS CONSIST OF ONE AND ONLY ONE MEMBER.                        *)
```

```
(*     FROM THE USER ADB, OBTAIN THE USER NAME AND PUT IT INTO AN     *)
(*       ARRAY.  IF THE USER IS A GLOBAL ATTRIBUTE, PUT THE KEYWORD   *)
(*       "GLOBAL" INTO THE ARRAY.                                     *)
(*     DEFINE THE COMMENTS FOR THE CROSS REFERENCE REPORT             *)
(*     DISPLAY THE CROSS REFERENCE REPORT RESULT MENU                 *)
(*                                                                    *)
(*   $COMMENTS:                                                       *)
(*                                                                    *)
(*   $CHANGE CONTROL:                                                 *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                             *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/19/87       C. H. MOHME                DBMA     *)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XATTRES *)
```

```
(* %INCLUDE XEXPREC *)
(**)
  PROCEDURE XEXPREC(VAR   IRC      : RET_REC;
                    CONST ENT_KIND : INTEGER;
                    VAR   XREFFILE : TEXT;
                    VAR   MSG      : MESSAGE);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE GENERATES A LIST OF ALL EXISTING PRECISIONS        *)
(*        FOR THE INTEGER, REAL, OR STRING DATA TYPE.                     *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC             O   RETURN CODE                                 *)
(*        ENT_KIND        O   THE ENTITY KIND                             *)
(*        XREFFILE        O   THE CROSS REFERENCE REPORT FILE             *)
(*        MSG             O   PANEL MESSAGE                               *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*      ESTABLISH THE MAXIMUM PRECISION AND THE DISPLAY MESSAGE.          *)
(*      THE EXISTS ARRAY IS USED TO HOLD BOOLEAN VALUES.  EXISTS(.X.)     *)
(*        IS TRUE IF THE X PRECISION EXISTS FOR THE INTEGER, REAL, OR     *)
(*        STRING DATA TYPE IN THE SCHEMA MODEL.                           *)
(*      MAKE A LIST OF THE INTEGER, REAL, OR STRING DATA TYPES AND        *)
(*        DETERMINE THE EXISTING PRECISIONS.                              *)
(*      PREPARE TO DISPLAY THE RESULTS.                                   *)
(*      DISPLAY THE RESULTS.                                              *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID     *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                 *)
(*                                                                        *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER          GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                      *)
(*    ORIGINATED: 11/20/87       C. H. MOHME                  DBMA     *)
(*                                                                      *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XEXPREC *)
```

```
(* %INCLUDE XFNDARY *)
(**)
  PROCEDURE XFNDARY(CONST ENTITY_KEY : ENTKEY;
                    VAR   ADB        : ENTBLOCK;
                    VAR   DATA       : BLKDATA;
                    VAR   RRC        : EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE FINDS ALL ARRAY DATA TYPES OF SET, LIST, OR   *)
(*        ARRAY AND PUTS THEM ON A LIST.                             *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                           *)
(*        ====          ===  ===========                           *)
(*        ENTITY_KEY     I   KEY TO THE ENTITY                     *)
(*        ADB            O   DATA STORED IN THE ADB                *)
(*        DATA          I/O  THE USER DEFINED DATA STRUCTURE USED  *)
(*                           TO PASS DATA INTO THIS PROCEDURE AND  *)
(*                           TO GET THE DESIRED OUTPUT FROM THE    *)
(*                           PROCEDURE                             *)
(*        RRC            O   EXTERNAL RETURN CODE                  *)
(*                           = 0  OK                               *)
(*                           >= 10  ERROR                          *)
(*                                                                 *)
(*   $COMMONS:                                                     *)
(*      NONE                                                       *)
(*                                                                 *)
(*   $ENVIRONMENT:                                                 *)
(*        LANGUAGE: IBM PASCAL                                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                 *)
(*   $EXECUTION PROCEDURE:                                         *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE              *)
(*                                                                 *)
(*   $PROCESSING DESCRIPTION:                                      *)
(*        THIS ROUTINE IS CALLED BY A MAS EXECUTE ROUTINE.  ALL    *)
(*        ARRAY DATA TYPES OF SET, LIST, OR ARRAY ARE PUT ONTO A   *)
(*        LIST.                                                    *)
(*                                                                 *)
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*                                                                 *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                          *)
(*                                                                 *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/24/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XFNDARY *)
```

```
(* %INCLUDE XFNDKEY *)
(**)
  PROCEDURE XFNDKEY(VAR IRC                : RET_REC;
                    VAR CHAR_STRING        : T_NAME;
                    VAR BAD_NAME           : BOOLEAN;
                    VAR BAD_KIND_NUMBER    : BOOLEAN;
                    VAR NUMBER             : INTEGER;
                    CONST ENT_KIND         : INTEGER);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*    $FUNCTION:                                                          *)
(*         THIS ROUTINE DETERMINES THE KEY FOR A GIVEN ENTITY NAME        *)
(*         OR NUMBER                                                      *)
(*                                                                        *)
(*    $DESCRIPTION OF ARGUMENTS:                                          *)
(*         NAME           I/O  DESCRIPTION                               *)
(*         ====           ===  ===========                               *)
(*         IRC             O   RETURN CODE                               *)
(*         CHAR_STRING     I   THE GIVEN CHARACTER STRING                *)
(*         BAD_NAME        O   INDICATES IF THE GIVEN CHARACTER STRING    *)
(*                               IS A NAME                                *)
(*     BAD_KIND_NUMBER     O   INDICATES IF THE GIVEN CHARACTER STRING    *)
(*                               IS A NUMBER                              *)
(*         NUMBER          O   THE NUMBER CONTAINED IN THE CHARACTER      *)
(*                               STRING                                   *)
(*         DATA_REC        O   THE RECORD USED BY MAKXEQ                 *)
(*         ENT_KIND        I   THE ENTITY KIND                           *)
(*                                                                        *)
(*    $COMMONS:                                                           *)
(*         NONE                                                           *)
(*                                                                        *)
(*    $ENVIRONMENT:                                                       *)
(*         LANGUAGE: IBM PASCAL                                           *)
(*         HARDWARE SYSTEM: IBM 360/370/4341/4381                         *)
(*                                                                        *)
(*    $EXECUTION PROCEDURE:                                               *)
(*         INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                    *)
(*                                                                        *)
(*    $PROCESSING DESCRIPTION:                                            *)
(*         INITIALIZE THE RECORD USED BY MAS IN MAKXEQ.  THIS RECORD      *)
(*         CONTAINS AN ENTITY NAME OR KIND NUMBER.  THE MAKXEQ ROUTINE    *)
(*         WILL RETURN THE ENTITY KEY, IF THE ENTITY EXISTS IN THE        *)
(*         MODEL.                                                         *)
(*                                                                        *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*     NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*    ORIGINATED: 11/17/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XFNDKEY *)
```

```
(* %INCLUDE XFNDNAME *)
(**)
  PROCEDURE XFNDNAME(CONST ENTITY_KEY : ENTKEY;
                     VAR   ADB        : ENTBLOCK;
                     VAR   DATA       : BLKDATA;
                     VAR   RRC        : EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE FINDS ALL ATTRIBUTES WITH THE GIVEN NAME AND  *)
(*        PUTS THEM ON A LIST.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        ENTITY_KEY      I   KEY TO THE ENTITY                     *)
(*        ADB             O   DATA STORED IN THE ADB                *)
(*        DATA           I/O  THE USER DEFINED DATA STRUCTURE USED  *)
(*                            TO PASS DATA INTO THIS PROCEDURE AND  *)
(*                            TO GET THE DESIRED OUTPUT FROM THE    *)
(*                            PROCEDURE                             *)
(*        RRC             O   EXTERNAL RETURN CODE                  *)
(*                            = 0  OK                               *)
(*                            >= 10  ERROR                          *)
(*                                                                  *)
(*   $COMMONS:                                                      *)
(*      NONE                                                        *)
(*                                                                  *)
(*   $ENVIRONMENT:                                                  *)
(*        LANGUAGE: IBM PASCAL                                      *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                    *)
(*                                                                  *)
(*   $EXECUTION PROCEDURE:                                          *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE               *)
(*                                                                  *)
(*   $PROCESSING DESCRIPTION:                                       *)
(*        THIS ROUTINE IS CALLED BY A MAS EXECUTE ROUTINE.  ALL     *)
(*        ATTRIBUTES WITH THE GIVEN NAME ARE RETURNED IN A LIST.    *)
(*                                                                  *)
(*   $COMMENTS:                                                     *)
(*                                                                  *)
(*   $CHANGE CONTROL:                                               *)
(*                                                                  *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                           *)
(*                                                                  *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/19/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XFNDNAME *)
```

```
(* %INCLUDE XFNDPREC *)
(**)
  PROCEDURE XFNDPREC(CONST ENTITY_KEY : ENTKEY;
                     VAR   ADB        : ENTBLOCK;
                     VAR   DATA       : BLKDATA;
                     VAR   RRC        : EXT_RET_CODE);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------------*)
(*                                                                         *)
(*   $FUNCTION:                                                            *)
(*        THIS ROUTINE FINDS ALL INTEGERS, REALS, OR STRINGS WITH          *)
(*        THE SPECIFIED PRECISION AND PUTS THEM ON A LIST.                 *)
(*                                                                         *)
(*   $DESCRIPTION OF ARGUMENTS:                                            *)
(*        NAME           I/O  DESCRIPTION                                  *)
(*        ====           ===  ===========                                  *)
(*        ENTITY_KEY      I   KEY TO THE ENTITY                           *)
(*        ADB             O   DATA STORED IN THE ADB                      *)
(*        DATA           I/O  THE USER DEFINED DATA STRUCTURE USED        *)
(*                            TO PASS DATA INTO THIS PROCEDURE AND         *)
(*                            TO GET THE DESIRED OUTPUT FROM THE           *)
(*                            PROCEDURE                                    *)
(*        RRC             O   EXTERNAL RETURN CODE                        *)
(*                            = 0  OK                                      *)
(*                            >= 10  ERROR                                 *)
(*                                                                         *)
(*   $COMMONS:                                                             *)
(*      NONE                                                               *)
(*                                                                         *)
(*   $ENVIRONMENT:                                                         *)
(*        LANGUAGE: IBM PASCAL                                             *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                           *)
(*                                                                         *)
(*   $EXECUTION PROCEDURE:                                                 *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                      *)
(*                                                                         *)
(*   $PROCESSING DESCRIPTION:                                              *)
(*        THIS ROUTINE IS CALLED BY A MAS EXECUTE ROUTINE.  ALL            *)
(*        INTEGERS, REALS, OR STRINGS WITH THE SPECIFIED PRECISION         *)
(*        ARE FOUND AND PUT ONTO A LIST.                                   *)
(*                                                                         *)
(*   $COMMENTS:                                                            *)
(*                                                                         *)
(*   $CHANGE CONTROL:                                                      *)
(*                                                                         *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID      *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                  *)
(*                                                                         *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/19/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*--------------------------------------------------------------------*)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XFNDPREC *)
```

```
(* %INCLUDE XLISTENT *)
(**)
  PROCEDURE XLISTENT(VAR IRC                  : RET_REC;
                     CONST SPECIFIED_KEY_KIND : INTEGER;
                     CONST RESULT_LIST_KIND   : INTEGER;
                     VAR   XREFFILE           : TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE GENERATES A LIST OF ALL "ENTITIES" CONTAINING  *)
(*        A PARTICULAR ENTITY.                                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                            *)
(*        ====           ===  ===========                            *)
(*        IRC             O   RETURN CODE                            *)
(*   SPECIFIED_KEY_KIND   I   THE ENTITY KIND OF THE PARTICULAR ENTITY *)
(*                              TO BE FOUND.                         *)
(*   RESULT_LIST_KIND     I   THE ENTITY KIND OF THE RESULTING LIST    *)
(*        XREFFILE        O   THE CROSS REFERENCE REPORT FILE         *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*     DISPLAY THE CROSS REFERENCE MENU                              *)
(*     DEFINE THE MENU COMMENTS TO DISPLAY FOR THE USER'S INFORMATION *)
(*     DISPLAY THE SPECIFICATION MENU, IF APPROPRIATE                *)
(*     DETERMINE WHICH OPTION WAS CHOSEN AND CALL THE APPROPRIATE    *)
(*       ROUTINE                                                     *)
(*     DETERMINE IF A VALID NAME OR NUMBER WAS ENTERED               *)
(*     DETERMINE THE KEY OF THE NAME OR NUMBER SPECIFIED             *)
(*     CREATE THE DESIRED LIST AND DEFINE THE CROSS REFERENCE REPORT *)
(*       COMMENTS                                                    *)
(*     FIND ALL ATTRIBUTES OF THE SPECIFIED DEFINED TYPE             *)
(*     FIND ALL CLASSES CONTAINING THE SPECIFIED ENTITY              *)
(*     FIND ALL SUBSCHEMAS CONTAINING THE SPECIFIED ENTITY           *)
(*     FIND ALL SUBSCHEMAS CONTAINING THE SPECIFIED CLASS            *)
(*     DISPLAY THE RESULTS                                           *)
(*     RETURN TO MAIN MENU                                           *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                              *)
(*                                                                       *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER        GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                       *)
(*    ORIGINATED: 11/16/87       C. H. MOHME                DBMA     *)
(*                                                                       *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XLISTENT *)
```

```
(* %INCLUDE XMAIN *)
(**)
  PROCEDURE XMAIN(VAR IRC      : RET_REC;
                  VAR XREFFILE : TEXT);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DETERMINES THE CROSS REFERENCE OPTION DESIRED  *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        IRC            O   RETURN CODE                            *)
(*        XREFFILE       O   CROSS REFERENCE REPORT FILE            *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*      DISPLAY THE CROSS REFERENCE MAIN MENU.                       *)
(*      DETERMINE WHICH OPTION WAS CHOSEN AND CALL THE APPROPRIATE   *)
(*        ROUTINE.                                                   *)
(*        LIST ALL ATTRIBUTES OF A PARTICULAR DATA TYPE (THIS LIST OF *)
(*          ATTRIBUTES ALSO INCLUDES THE ATTRIBUTES' USER ENTITY).   *)
(*        LIST ALL ENTITIES HAVING AN ATTRIBUTE OF A PARTICULAR      *)
(*          DEFINED TYPE.                                            *)
(*        LIST ALL ENTITIES HAVING AN ATTRIBUTE WITH A PARTICULAR    *)
(*          NAME.                                                    *)
(*        LIST ALL CLASSES CONTAINING A PARTICULAR ENTITY.           *)
(*        LIST ALL SUBSCHEMAS CONTAINING A PARTICULAR ENTITY.        *)
(*        LIST ALL SUBSCEHMAS CONTAINING A PARTICULAR CLASS.         *)
(*        LIST ALL EXISTING PRECISIONS FOR THE INTEGER DATA TYPE.    *)
(*        LIST ALL EXISTING PRECISIONS FOR THE REAL DATA TYPE.       *)
(*        LIST ALL EXISTING LENGTHS FOR THE STRING DATA TYPE.        *)
(*        RETURN TO MAIN MENU                                        *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
```

```
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER            GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                             *)
(*                                                                    *)
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/16/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XMAIN *)
```

```
(* %INCLUDE XMATTRES *)
(**)
PROCEDURE XMATTRES(VAR MESS       : MESSAGE;
                   VAR CLAS       : T_ARRAYRV;
                   VAR ARRAY_SIZE : INTEGER;
                   VAR COM1       : CHAR50;
                   VAR COM2       : CHAR50;
                   VAR XREFFILE   : TEXT;
                   VAR NEXT_OP    : OPERATIONS;
                   VAR RR         : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE MENU                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        CLAS           I   THE ARRAY OF MEMBERS                    *)
(*        ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        COM1           I   THE MENU COMMENTS                       *)
(*        COM2           I   THE MENU COMMENTS                       *)
(*        XREFFILE      I/O  THE CROSS REFERENCE REPORT FILE         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*     INITIALIZE THE VARIABLES                                      *)
(*     PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM  *)
(*       VARIABLES                                                   *)
(*     CREATE THE TABLE                                              *)
(*     PREPARE TO DISPLAY THE CROSS REFERENCE DISPLAY MENU           *)
(*     LOAD THE TABLE AND WRITE THE CROSS REFERENCE REPORT RESULTS TO *)
(*       FILE                                                        *)
```

```
(*     DISPLAY THE CROSS REFERENCE DISPLAY MENU                          *)
(*     DETERMINE THE ACTION SELECTED FROM THE MENU                       *)
(*     REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM         *)
(*       VARIABLES                                                       *)
(*     REMOVE THE TABLE                                                  *)
(*                                                                       *)
(*    $COMMENTS:                                                         *)
(*         NONE                                                          *)
(*                                                                       *)
(*    $CHANGE CONTROL:                                                   *)
(*                                                                       *)
(*     REVISED: MM/DD/YY CCRR      I. M. THECHANGER            GROUP_ID *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                               *)
(*                                                                       *)
(*     REVISED: MM/DD/YY CCZZ      I. M. THEPROGRAMMER         GROUP_ID *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE            *)
(*      NARATION ON THE NEXT LINE.                                       *)
(*                                                                       *)
(*     ORIGINATED: 11/19/87        C. H. MOHME                 DBMA     *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE CRDISP *)
```

```
(* %INCLUDE XMMAIN *)
(**)
  PROCEDURE XMMAIN(VAR MESS    : MESSAGE;
                   VAR NEXT_OP : OPERATIONS;
                   VAR RR      : RET_REC);
       SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE MENU                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*     INITIALIZE THE VARIABLES                                      *)
(*     PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM  *)
(*       VARIABLES                                                   *)
(*     DISPLAY THE CROSS REFERENCE MENU                              *)
(*     DETERMINE THE ACTION SELECTED FROM THE MENU                   *)
(*     REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM     *)
(*       VARIABLES                                                   *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID  *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
```

```
(*    ORIGINATED: 11/16/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE XMMAIN *)
```

```
(* %INCLUDE XMNAMSPE *)
(**)
  PROCEDURE XMNAMSPE(VAR MESS    : MESSAGE;
                     VAR NAME    : T_NAME;
                     VAR COM1    : CHAR50;
                     VAR COM2    : CHAR50;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE REPORT MENU                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ==========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NAME           I   THE ENTITY NAME                         *)
(*        COM1           I   THE MENU COMMENTS                       *)
(*        COM2           I   THE MENU COMMENTS                       *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*     INITIALIZE THE VARIABLES                                      *)
(*     PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM  *)
(*       VARIABLES                                                   *)
```

```
(*     DISPLAY THE ENTITY SPECIFICATION MENU                              *)
(*     DETERMINE THE ACTION SELECTED FROM THE MENU                        *)
(*     REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM          *)
(*       VARIABLES                                                        *)
(*                                                                        *)
(*    $COMMENTS:                                                          *)
(*         NONE                                                           *)
(*                                                                        *)
(*    $CHANGE CONTROL:                                                    *)
(*                                                                        *)
(*     REVISED: MM/DD/YY              NAME                      GROUP     *)
(*       COMMENTS                                                         *)
(*                                                                        *)
(*     REVISED: MM/DD/YY              NAME                      GROUP     *)
(*       COMMENTS                                                         *)
(*                                                                        *)
(*     REVISED: MM/DD/YY              NAME                      GROUP     *)
(*       COMMENTS                                                         *)
(*                                                                        *)
(*     ORIGINATED: 11/17/87       C. H. MOHME                   DBMA      *)
(*                                                                        *)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*END-------------------------------------------------------------------*)
(* END %INCLUDE XMNAMSPE *)
```

```
(* %INCLUDE XMPRESPE *)
(**)
  PROCEDURE XMPRESPE(VAR MESS           : MESSAGE;
                     VAR DATA_TYPE_KIND : INTEGER;
                     VAR SIZE           : CHAR8;
                     VAR NEXT_OP        : OPERATIONS;
                     VAR RR             : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE REPORT MENU                       *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                           *)
(*        ====           ===  ===========                           *)
(*        MESS            I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        DATA_TYPE_KIND  I   THE DATA TYPE KIND                     *)
(*        SIZE            O   THE PRECISION OF THE REAL ENTERED      *)
(*        NEXT_OP         O   ENUMERATED TYPE INDICATING THE NEXT    *)
(*                            OPERATION                              *)
(*        RR              O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                            IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*      INITIALIZE THE VARIABLES                                     *)
(*      PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM *)
(*        VARIABLES                                                  *)
(*      DISPLAY THE CROSS REFERENCE MENU                             *)
(*      DETERMINE THE ACTION SELECTED FROM THE MENU                  *)
(*      REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM    *)
(*        VARIABLES                                                  *)
(*                                                                   *)
```

```
(*   $COMMENTS:                                                          *)
(*        NONE                                                           *)
(*                                                                       *)
(*   $CHANGE CONTROL:                                                    *)
(*                                                                       *)
(*    ORIGINATED: 11/19/87       C. H. MOHME                 DBMA       *)
(*                                                                       *)
(*-----------------------------------------------------------------------*)
(*                                                                       *)
(*END------------------------------------------------------------------*)
(* END %INCLUDE XMPRESPE*)
```

```
(* %INCLUDE XMRESULT *)
(**)
PROCEDURE XMRESULT(VAR MESS       : MESSAGE;
                   VAR CLAS       : T_ARRAYTV;
                   VAR ARRAY_SIZE : INTEGER;
                   VAR COM1       : CHAR50;
                   VAR COM2       : CHAR50;
                   VAR XREFFILE   : TEXT;
                   VAR NEXT_OP    : OPERATIONS;
                   VAR RR         : RET_REC);
SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE MENU                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        CLAS           I   THE ARRAY OF MEMBERS                    *)
(*        ARRAY_SIZE     I   THE SIZE OF THE ARRAY OF MEMBERS        *)
(*        COM1           I   MENU COMMENTS                           *)
(*        COM2           I   MENU COMMENTS                           *)
(*        XREFFILE      I/O  THE CROSS REFERENCE REPORT FILE         *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*      INITIALIZE THE VARIABLES                                     *)
(*      PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM *)
(*        VARIABLES.                                                 *)
```

```
(*      CREATE THE TABLE                                                    *)
(*      PREPARE TO DISPLAY THE CROSS REFERENCE DISPLAY MENU                 *)
(*      LOAD THE TABLE AND WRITE THE CROSS REFERENCE REPORT RESULTS         *)
(*        TO FILE                                                           *)
(*      DISPLAY THE CROSS REFERENCE DISPLAY MENU                            *)
(*      DETERMINE THE ACTION SELECTED FROM THE MENU                         *)
(*      REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM           *)
(*        VARIABLES                                                         *)
(*      REMOVE THE TABLE                                                    *)
(*                                                                          *)
(*    $COMMENTS:                                                            *)
(*         NONE                                                             *)
(*                                                                          *)
(*    $CHANGE CONTROL:                                                      *)
(*                                                                          *)
(*     REVISED: MM/DD/YY CCRR      I. M. THECHANGER            GROUP_ID    *)
(*      DESCRIPTION OF LATEST CHANGE MADE.                                  *)
(*                                                                          *)
(*     REVISED: MM/DD/YY CCZZ      I. M. THEPROGRAMMER         GROUP_ID    *)
(*      DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE               *)
(*      NARATION ON THE NEXT LINE.                                          *)
(*                                                                          *)
(*     ORIGINATED: 11/17/87        C. H. MOHME                 DBMA        *)
(*                                                                          *)
(*--------------------------------------------------------------------------*)
(*                                                                          *)
(*END-----------------------------------------------------------------------*)
(* END %INCLUDE CRDISP *)
```

```
(* %INCLUDE XMTYPSPE *)
(**)
  PROCEDURE XMTYPSPE(VAR MESS    : MESSAGE;
                     VAR NEXT_OP : OPERATIONS;
                     VAR RR      : RET_REC);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*      DISPLAYS A CROSS REFERENCE MENU                              *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME          I/O  DESCRIPTION                             *)
(*        ====          ===  ===========                             *)
(*        MESS           I   THE ERROR MESSAGE DISPLAYED ON THE PANEL *)
(*        NEXT_OP        O   ENUMERATED TYPE INDICATING THE NEXT     *)
(*                           OPERATION                               *)
(*        RR             O   INDICATES IF AN ERROR HAS OCCURRED AND, *)
(*                           IF ONE HAS, WHAT ROUTINE IT OCCURRED IN *)
(*                                                                   *)
(*   $COMMONS:                                                       *)
(*      NONE                                                         *)
(*                                                                   *)
(*   $ENVIRONMENT:                                                   *)
(*        LANGUAGE: IBM PASCAL                                       *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                     *)
(*        DDNAMES USED WITH STANDARD FILES:                          *)
(*             NONE                                                  *)
(*                                                                   *)
(*   $EXECUTION PROCEDURE:                                           *)
(*        SCHEMA EXECUTIVE MENU INTERFACE ROUTINE                    *)
(*                                                                   *)
(*   $PROCESSING DESCRIPTION:                                        *)
(*      INITIALIZE THE VARIABLES                                     *)
(*      PERMIT THE COMMUNICATION BETWEEN PANEL VARIABLES AND PROGRAM *)
(*        VARIABLES                                                  *)
(*      DISPLAY THE CROSS REFERENCE MENU                             *)
(*      DETERMINE THE ACTION SELECTED FROM THE MENU                  *)
(*      REMOVE CORRESPONDENCE BETWEEN PANEL VARIABLES AND PROGRAM    *)
(*        VARIABLES                                                  *)
(*                                                                   *)
(*   $COMMENTS:                                                      *)
(*        NONE                                                       *)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*                                                                   *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID  *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                            *)
(*                                                                   *)
```

```
(*    ORIGINATED: 11/19/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*---------------------------------------------------------------------*)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XMTYPSPE *)
```

```
(* %INCLUDE XNAMENUM *)
(**)
  PROCEDURE XNAMENUM(VAR IRC             : RET_REC;
                     VAR CHAR_STRING     : T_NAME;
                     VAR BAD_NAME        : BOOLEAN;
                     VAR BAD_KIND_NUMBER : BOOLEAN;
                     VAR NUMBER          : INTEGER);
      SUBPROGRAM;
(**)
(*------------------------------------------------------------------------*)
(*                                                                        *)
(*   $FUNCTION:                                                           *)
(*        THIS ROUTINE DETERMINES IF A CHARACTER STRING IS A NAME         *)
(*        OR NUMBER                                                       *)
(*                                                                        *)
(*   $DESCRIPTION OF ARGUMENTS:                                           *)
(*        NAME           I/O  DESCRIPTION                                 *)
(*        ====           ===  ===========                                 *)
(*        IRC             O   RETURN CODE                                 *)
(*        CHAR_STRING     I   THE GIVEN CHARACTER STRING                  *)
(*        BAD_NAME        O   INDICATES IF THE GIVEN CHARACTER STRING     *)
(*                              IS A NAME                                 *)
(*    BAD_KIND_NUMBER     O   INDICATES IF THE GIVEN CHARACTER STRING     *)
(*                              IS A NUMBER                               *)
(*        NUMBER          O   THE NUMBER CONTAINED IN THE CHARACTER       *)
(*                              STRING                                    *)
(*                                                                        *)
(*   $COMMONS:                                                            *)
(*        NONE                                                            *)
(*                                                                        *)
(*   $ENVIRONMENT:                                                        *)
(*        LANGUAGE: IBM PASCAL                                            *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                          *)
(*                                                                        *)
(*   $EXECUTION PROCEDURE:                                                *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE                     *)
(*                                                                        *)
(*   $PROCESSING DESCRIPTION:                                             *)
(*        DETERMINE IF THE CHARACTER STRING (CHAR_STRING) IS A VALID      *)
(*        NAME OR KIND NUMBER.                                            *)
(*                                                                        *)
(*   $COMMENTS:                                                           *)
(*                                                                        *)
(*   $CHANGE CONTROL:                                                     *)
(*                                                                        *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER           GROUP_ID     *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                                 *)
(*                                                                        *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER          GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE           *)
(*     NARATION ON THE NEXT LINE.                                      *)
(*                                                                     *)
(*    ORIGINATED: 11/17/87       C. H. MOHME                  DBMA     *)
(*                                                                     *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XNAMENUM *)
```

```
(* %INCLUDE XRESULT *)
(**)
  PROCEDURE XRESULT(VAR IRC              : RET_REC;
                    VAR RESULT_LIST      : LISTKEY;
                    VAR COM1             : CHAR50;
                    VAR COM2             : CHAR50;
                    VAR XREFFILE         : TEXT;
                    VAR OPTION           : OPERATIONS);
      SUBPROGRAM;
(**)
(*-------------------------------------------------------------------*)
(*                                                                   *)
(*   $FUNCTION:                                                      *)
(*        THIS ROUTINE DISPLAYS THE CROSS REFERENCE REPORT RESULTS   *)
(*                                                                   *)
(*   $DESCRIPTION OF ARGUMENTS:                                      *)
(*        NAME           I/O  DESCRIPTION                          *)
(*        ====           ===  ===========                          *)
(*        IRC             O   RETURN CODE                          *)
(*        RESULT_LIST     I   THE CROSS REFERENCE RESULT LIST      *)
(*        COM1            I   MENU COMMENTS DESCRIBING THE OUTPUT  *)
(*        COM2            I   MENU COMMENTS DESCRIBING THE OUTPUT  *)
(*        XREFFILE       I/O  CROSS REFERENCE REPORT FILE          *)
(*        OPTION          O   OPTION SELECTED                      *)
(*                                                                 *)
(*   $COMMONS:                                                     *)
(*        NONE                                                     *)
(*                                                                 *)
(*   $ENVIRONMENT:                                                 *)
(*        LANGUAGE: IBM PASCAL                                     *)
(*        HARDWARE SYSTEM: IBM 360/370/4341/4381                   *)
(*                                                                 *)
(*   $EXECUTION PROCEDURE:                                         *)
(*        INTERNAL PROCEDURE FOR THE SCHEMA EXECUTIVE              *)
(*                                                                 *)
(*   $PROCESSING DESCRIPTION:                                      *)
(*      READ EACH KEY OFF OF THE RESULT LIST.                      *)
(*      FROM EACH ENTITY ADB OBTAIN THE ENTITY NAME (AND KIND NUMBER, *)
(*        IF APPLICABLE).                                          *)
(*      FILL UP THE TABLE OF NAMES AND KIND NUMBERS FOR DISPLAY ON *)
(*        THE CROSS REFERENCE MENU.                                *)
(*      DISPLAY THE CROSS REFERENCE REPORT RESULTS MENU.           *)
(*                                                                 *)
(*   $COMMENTS:                                                    *)
(*                                                                 *)
(*   $CHANGE CONTROL:                                              *)
(*                                                                 *)
(*    REVISED: MM/DD/YY CCRR     I. M. THECHANGER          GROUP_ID *)
(*     DESCRIPTION OF LATEST CHANGE MADE.                          *)
(*                                                                 *)
```

```
(*    REVISED: MM/DD/YY CCZZ     I. M. THEPROGRAMMER         GROUP_ID *)
(*     DESCRIPTION OF CHANGE MADE.  IF LENGTHY, CONTINUE THE          *)
(*     NARATION ON THE NEXT LINE.                                     *)
(*                                                                    *)
(*    ORIGINATED: 11/17/87       C. H. MOHME                 DBMA     *)
(*                                                                    *)
(*END-----------------------------------------------------------------*)
(* END %INCLUDE XRESULT *)
```

APPENDIX J

SCHEMA MANAGER DATA DICTIONARY

This section provides the data structures for the Schema Manager. The following index provides an alphabetic listing of the functions:

BALTYP - Boundary Alignment Types
BATTYP - Batch Interface Types and Constants
DDTYP - Data Dictionary Constants & Types
LEXTYPE - Batch Interface Types and Constants
NVITYP - NVI Constants & Types
RTSTYP - Run-Time Subschema Constants & Types
SCECON - Schema Manager Constants
SCETYP - Schema Manager Types

```
(* (BALTYP) ************************************************************)
(*                                                                      *)
(*   $CHANGE CONTROL:                                                   *)
(*        ORIGINATED: 17 SEPTEMBER 1986, M. H. CHOI, DBMA               *)
(*                                                                      *)
(************************************************************************)
TYPE
(*                                                                      *)
(*                                                                      *)
   T_DW_POINTER = @T_DW_FRAME;
   T_DW_FRAME = RECORD
      NEXT            : T_DW_POINTER;
      NAME            : T_NAME;
      SIZE            : INTEGER;
      END;
(*                                                                      *)
   T_FW_POINTER = @T_FW_FRAME;
   T_FW_FRAME = RECORD
      NEXT            : T_FW_POINTER;
      NAME            : T_NAME;
      SIZE            : INTEGER;
      END;
(*                                                                      *)
   T_HW_POINTER = @T_HW_FRAME;
   T_HW_FRAME = RECORD
      NEXT            : T_HW_POINTER;
      NAME            : T_NAME;
      SIZE            : INTEGER;
      END;
(*                                                                      *)
   T_BY_POINTER = @T_BY_FRAME;
   T_BY_FRAME = RECORD
      NEXT            : T_BY_POINTER;
      NAME            : T_NAME;
      SIZE            : INTEGER;
      END;
(*                                                                      *)
   T_PNTR_POINTER = @T_PNTR_FRAME;
   T_PNTR_FRAME = RECORD
      NEXT            : T_PNTR_POINTER;
      NAME            : T_NAME;
      SIZE            : INTEGER;
      END;
(*                                                                      *)
   T_LIST_BOUNDARY = RECORD
      NAME         : T_NAME;
      OFFSET       : INTEGER;
      SIZE         : INTEGER;
      END;
```

```
(*                                                                    *)
   T_OFFSET_LIST = ARRAY (. 1..100 .) OF T_LIST_BOUNDARY;
(*                                                                    *)
   T_HIGH_BOUND = ARRAY (. 1..100 .) OF INTEGER;
   T_LOW_BOUND = ARRAY (. 1..100 .) OF INTEGER;
(*                                                                   **)
   T_OPTIONAL_GLOBAL = RECORD
      KEY          : ENTKEY;
      POSITION     : INTEGER;
      END;
(*                                                                   **)
   T_GLOBAL_FIELD = ARRAY (. 1..100 .) OF T_OPTIONAL_GLOBAL;
(*                                                                   **)
   T_ALIGN = ( DW, FW, HW, BY, PNTR, NONE );
(*                                                                   **)
   T_S_Info_Pointer  = @T_S_Info_Frame;
   T_S_Info_Frame = Record
      Name          : T_Name;
      Offset        : Integer;
      Size          : Integer;
      Next          : T_S_Info_Pointer;
      End;
(*                                                                    *)
   T_Struc_Pointer  = @T_Struc_Frame;
   T_Struc_Frame = Record
      Next          : T_Struc_pointer;
      Name          : T_Name;
      Defn          : T_S_Info_Pointer;
      End;
(*                                                                   **)
(* END %INCLUDE BALTYP ************************************************)
```

```
(* BEGIN %INCLUDE BATTYPE ********************************************)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 20 MARCH 1987, C. H. MOHME                     *)
(*                                                                   *)
(*********************************************************************)
CONST
   FIRST                    = 1;

TYPE
   BLKDATA = RECORD
     NAME        :  T_NAME;
     KIND        :  INTEGER;
     NAME_FOUND  :  BOOLEAN;
     KIND_FOUND  :  BOOLEAN;
     KEY         :  ENTKEY;
   END;

   ENTITY_LIST_PTR = @ENTITY_INFORMATION;

   ENTITY_INFORMATION = RECORD
     NAME   :  T_NAME;
     NUMBER :  INTEGER;
     NEXT   :  ENTITY_LIST_PTR;
   END;

   LEXSTRING = STRING(MAX_LENGTH_VALUE);

   T_EXPECTED       = RECORD
      ENTRIES          : INTEGER;
      TOKEN_VALUE      : ARRAY(.1..20.) OF T_TOKEN_VALUE
      END;

(*                                                                   *)
(* END   %INCLUDE BATTYPE ********************************************)
```

```
(* (DDTYP) DATA DICTIONARY CONSTANTS AND TYPES ************************)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        REVISED   : 15 MAY 1987, M. H. CHOI, DBMA                    *)
(*                    ADDED T_ADB_RECORD AND T_CL_RECORD TO RETURN     *)
(*                    THE DEFINITIONS IN PHYSICAL SCHEMA ORDER         *)
(*        ORIGINATED: 19 MARCH 1987, M. H. CHOI, DBMA                  *)
(*                                                                     *)
(***********************************************************************)
CONST
   CONTINUATION_FLAG    = 'X';
   PHYSICAL_ORDER       = 'P';
   MAX_ARRAY_REC        = 100;
(* RETURN CODE VALUES:                                                *)
   KIND_NOT_IN_DATA_DICTIONARY = 1;
   ACTUAL_SIZE_GT_SPACE_AVAIL = -1;
(*                                                                    *)
TYPE
   T_INX_RECORD = PACKED ARRAY (. 1..80 .) OF CHAR;
(*                                                                    *)
   T_USER_ARRAY = ARRAY (. 1..46 .) OF
                     PACKED ARRAY (. 1..80 .) OF CHAR;
(*                                                                    *)
   T_BOUNDS = RECORD
              LBOUND : PACKED ARRAY (. 1..4 .) OF CHAR;
              UBOUND : PACKED ARRAY (. 1..4 .) OF CHAR;
              END;
(*                                                                    *)
   T_ADB_RECORD = RECORD
                  ADB_KEY    : INTEGER;
                  POSITION   : INTEGER;
                  NO_OF_REC  : INTEGER;
                  END;
(*                                                                    *)
   T_CL_RECORD  = RECORD
                  POSITION   : INTEGER;
                  NO_OF_REC  : INTEGER;
                  END;
(*                                                                    *)
   T_VARIANT_RECORD = RECORD
      CASE INTEGER OF
        0 : ( BASIC_RECORD   : PACKED ARRAY (. 1..80 .) OF CHAR );
        1 : ( BOUNDS         : ARRAY (. 1..9 .) OF T_BOUNDS );
        2 : ( C_FLAG         : CHAR;
              NO_OF_KINDS    : PACKED ARRAY (. 1..2 .) OF CHAR;
              KINDS          : ARRAY (. 1..12 .) OF
                                  PACKED ARRAY (. 1..6 .) OF CHAR );
```

```
        3 : ( E_FLAG          : CHAR;
              NO_OF_VALUES    : PACKED ARRAY (. 1..2 .) OF CHAR;
              VALUES          : ARRAY (. 1..4 .) OF
                                    PACKED ARRAY (. 1..17 .) OF CHAR );
      END;
(*                                                                    *)
   T_ADB_ARRAY = ARRAY (. 1..MAX_ARRAY_REC .) OF T_ADB_RECORD;
(*                                                                    *)
   T_CL_ARRAY = ARRAY (. 1..MAX_ARRAY_REC .) OF T_CL_RECORD;
(*                                                                    *)
   T_FILE_VARIANT = FILE OF T_VARIANT_RECORD;
(*                                                                    *)
   T_INX_FILE = FILE OF T_INX_RECORD;
(*                                                                    *)
(* END %INCLUDE DDTYP ************************************************)
```

```
(* BEGIN %INCLUDE LEXTYPE ********************************************)
(*                                                                   *)
(*   $CHANGE CONTROL:                                                *)
(*        ORIGINATED: 19 MARCH 1987, G. A. WHITE                     *)
(*                                                                   *)
(*********************************************************************)
CONST
   Final_Token      = 'Final     ';
   Found_Addop      = 'Addop     ';
   Found_Assignment = 'Assignment';
   Found_Comment    = 'Comment   ';
   Found_Delimiter  = 'Delimiter ';
   Found_Error      = 'Error     ';
   Found_Identifier = 'Identifier';
   Found_Integer    = 'Integer   ';
   Found_Keyword    = 'Keyword   ';
   Found_Literal    = 'Literal   ';
   Found_Mulop      = 'Mulop     ';
   Found_Pound_Sign = 'Pound_Sign';
   Found_Real       = 'Real      ';
   Found_Relop      = 'Relop     ';
   Found_Unary      = 'Unary     ';
   Initial_Token    = 'Initial   ';
(* ARBITRARY SIZE PARAMETERS:                                        *)
   Max_Length_Token = 10;
   Max_Length_Value = 80;
Type
   T_Token          = String(Max_Length_Token);
   T_Token_Value    = String(Max_Length_Value);
(*                                                                   *)
(* END   %INCLUDE LEXTYPE ********************************************)
```

```
(* (NVITYP) NAME_VALUE INTERFACE CONSTANTS AND TYPES ******************)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        REVISED :  15 JULY 1987, M. H. CHOI, DBMA                    *)
(*                   ADDED COMPARISON VALUES                           *)
(*        REVISED :  16 SEPTEMBER 1986, M. H. CHOI, DBMA               *)
(*                   CHANGED STRUCTURE OF THE SCHEMA INSTANCE COLLECTOR *)
(*                   BECAUSE STRUCTURE CHANGED IN MAS TO HANDLE NEW    *)
(*                   DELETE AND COMPRESS RULES                         *)
(*        REVISED :  12 SEPTEMBER 1986, M. H. CHOI, DBMA               *)
(*                   ADDED FIELD TO T_INT_ITEM FOR THE COLL_ADB AND    *)
(*                   MAPROB2 BECAUSE T_INT_ITEM STRUCTURE CHANGED IN MAS*)
(*        REVISED :  04 JUNE 1986, M. H. CHOI, DBMA                    *)
(*                   ADDED RETURN CODE VALUES FOR FAILED_IN_MAL AND    *)
(*                   FAILURE.                                          *)
(*        ORIGINATED: 13 MAY 1986, G. A. WHITE, DBMA                   *)
(*                                                                     *)
(*********************************************************************)
CONST
(* ARBITRARY SIZE PARAMETERS:                                          *)
   MAX_ARRAY = 100;
   MAX_ATTRIBUTE_NAME = 1000;
   MAX_ATTRIBUTE_VALUE = 1000;
   MAX_CHARS         = 1000;
   MAX_GROUP         = 8;
   MAX_LIST          = 4000000;
   MAX_RDB_SIZE      = 65535;
   MAX_WORDS         = MAX_CHARS DIV 4;
   SCHEMA_NAME_SIZE = 16;
(* RETURN CODE VALUES:                                                 *)
   INVALID_ARRAY_ENTITY = 7;
   INVALID_SCALAR_VALUE = 6;
   FAILED_IN_MAL = 5;
   FAILED_IN_MAEXEQ = 4;
   NIL_ENTITY_KEY   = 3;
   ATTRIBUTE_NOT_IN_ENTITY = 2;
   KIND_NOT_IN_RUNTIME_SUBSCHEMA = 1;
   SUCCESS           = 0;
   WARNING           = -1;
   FAILURE           = -2;
(* COMPARISON VALUES:                                                  *)
   EQ = 1;
   LT = 2;
   GT = 3;
   NE = 4;
   LE = 5;
   GE = 6;
(* ATTRIBUTE NAME STRING DELIMITERS:                                   *)
   END_OF_SEGMENT = '.';
   END_OF_STRING  = '00'XC;
```

```
(* DIMENSION VALUE DELIMITERS:                                          *)
   BEGIN_OF_ARRAY = '(';
   END_OF_ARRAY   = ')';
TYPE
   T_ATTRIBUTE_NAME = ARRAY(. 1..MAX_ATTRIBUTE_NAME .) OF CHAR;
   T_DIMEN_VALUE    = ARRAY(. 1..MAX_ARRAY .) OF INTEGER;
   T_DISPLAY_WORD   = PACKED ARRAY(. 1..8 .) OF CHAR;
   T_HEX_BYTE       = PACKED 0..255;
   T_HEX_WORD       = ARRAY(. 1..4 .) OF T_HEX_BYTE;
   T_WORD           = ARRAY(. 1..4 .) OF CHAR;
   T_LOCATION       = (IN_ADB, IN_CL, IN_ENUMERATION, IN_STRUCTURE);
   T_SCHEMA_NAME    = PACKED ARRAY (. 1..SCHEMA_NAME_SIZE .) OF CHAR;
   T_SELECTOR       = PACKED 0..9;
   T_INTEGER_1      = PACKED -128..127;
   T_INTEGER_2      = PACKED -32768..32767;
   T_INTEGER_4      = MININT..MAXINT;
   T_VALUE          = ARRAY(. 1..MAX_ATTRIBUTE_VALUE .) OF CHAR;
   T_Array_Value    = ARRAY(. 1..MAX_ATTRIBUTE_VALUE .) OF T_Word;
(*                                                                      *)
   T_ATTRIBUTE_VALUE = RECORD
      CASE INTEGER OF
         0 : ( AS_VARIANT       : T_Value  );
         1 : ( AS_INTEGER_1     : T_Integer_1      );
         2 : ( AS_INTEGER_2     : T_Integer_2      );
         3 : ( AS_INTEGER_4     : T_Integer_4      );
         4 : ( AS_REAL_4        : SHORTREAL        );
         5 : ( AS_REAL_8        : REAL             );
         6 : ( AS_LOGICAL       : BOOLEAN          );
         7 : ( AS_ENUMERATION   : T_Schema_Name    );
         8 : ( AS_Array         : T_Array_Value    );
      END;
(*                                                                      *)
   T_VARIANT_VALUE = RECORD
       CASE INTEGER OF
          0 : ( AS_VARIANT      : T_VALUE    );
          1 : ( AS_INTEGER      : T_INTEGER_4          );
          2 : ( AS_REAL         : REAL                 );
       END;
(*                                                                      *)
   ROUTINE = PACKED ARRAY(. 1..8 .) OF CHAR;
(*                                                                      *)
   ENTDATA = RECORD
      CASE INTEGER OF
         0 : ( CHARS : PACKED ARRAY(. 1..MAX_CHARS .) OF CHAR);
         1 : ( WORDS : ARRAY(. 1..MAX_WORDS .) OF T_HEX_WORD );
         END;
(*                                                                      *)
   EXT_RET_CODE = INTEGER;
(*                                                                      *)
```

```
   LISTINDX = 0..MAX_LIST;
(*                                                                    *)
   LISTPSTN = 0..MAX_LIST;
(*                                                                    *)
   LISTSIZE = 0..MAX_LIST;
(*                                                                    *)
   ORD_KIND = 0..MAXINT;
(*                                                                    *)
   RDBSIZE = PACKED 0..MAX_RDB_SIZE;
(*                                                                    *)
   T_RULE_ELMNTS = ( COMPRESS, DELETE, USER_DELETE, CNST_DELETE );
(*                                                                    *)
   T_RULE = SET OF T_RULE_ELMNTS;
(*                                                                    *)
   T_GROUP = RECORD
     LAST_CNST : RDBSIZE;
     RULE      : T_RULE;
     END;
(*                                                                    *)
   T_GROUP_ARRAY = ARRAY (. 1..MAX_GROUP .) OF T_GROUP;
(*                                                                    *)
   T_SCH_INST_ENT = RECORD
      KIND : ORD_KIND;
      POSITION : LISTPSTN;
      NUM_GROUP: LISTPSTN;
      MIN_CNST : LISTPSTN;
      GROUP    : T_GROUP_ARRAY;
      END;
(*                                                                    *)
   ENTBLOCK = RECORD
      KIND : ORD_KIND;
      SIZE : 0..MAX_CHARS;
      SYSUSE : ARRAY (. 1..4 .) OF BOOLEAN;
      DATA : ENTDATA;
      END;
(*                                                                    *)
   ENTPNTR = @ENTBLOCK;
(*                                                                    *)
   LISTPNTR = @T_SYS_LIST;
(*                                                                    *)
   T_INT_ITEM = RECORD
      RDBEXIST : BOOLEAN;
      MAPROB   : BOOLEAN;
      MAPROB2  : BOOLEAN;
      COLL_ADB : ENTPNTR;
      USERS    : LISTPNTR;
      CNSTS    : LISTPNTR;
      ENPTR    : ENTPNTR;
      END;
(*                                                                    *)
```

```
   ENTITIES = ( NIL_ENT, INT_ROOT, INT_ITEM, APPL_LIST );
(*                                                                  *)
   T_ENTITY = RECORD
      FORM : ENTITIES;
      IIT : T_INT_ITEM;
      END;
(*                                                                  *)
   ANYKEY = RECORD
      P   : @T_ENTITY;
      END;
(*                                                                  *)
   ENTKEY = ANYKEY;
(*                                                                  *)
   LISTKEY = ANYKEY;
(*                                                                  *)
   T_SYS_LIST = RECORD
      SIZE : LISTSIZE;
      LSTLNG : LISTSIZE;
      LIST : ARRAY (. 1..MAX_LIST .) OF ENTKEY;
      END;
(*                                                                  *)
   T_DEFN_POINTER   = @T_DEFN_FRAME;
   T_DEFN_FRAME     = RECORD
      NEXT            : T_DEFN_POINTER;
      KIND            : ORD_KIND;
      DATA_TYPE       : INTEGER;
      CASE INTEGER OF
         1, 2, 3, 4     : ( OFFSET : INTEGER;
                            SIZE   : INTEGER );
         7, 8           : ( POSITION : INTEGER);
         5              : ( SELECTOR_OFFSET : INTEGER ;
                            TABLE_OFFSET    : INTEGER );
         END;
(*                                                                  *)
   T_NAME_POINTER   = @T_NAME_FRAME;
   T_NAME_FRAME     = RECORD
      NAME   : T_SCHEMA_NAME;
      NEXT   : T_NAME_POINTER;
      DEFN   : T_DEFN_POINTER;
      END;
(*                                                                  *)
   T_DATAREC        = RECORD
      NAME_ROOT       : T_NAME_POINTER;
      LIST_ROOT       : LISTKEY;
      ATTRIBUTE_VALUE : T_VARIANT_VALUE;
      DIMEN_VALUE     : T_DIMEN_VALUE;
      NO_OF_DIMENSION : INTEGER;
      END;
(*                                                                  *)
   BLKDATA = T_DATAREC;
(*                                                                  *)
(* END %INCLUDE NVITYP ************************************************)
```

```
(* (RTSTYP) RUN-TIME SUBSCHEMA CONSTANTS AND TYPES          **********)
(*                                                                     *)
(*   $CHANGE CONTROL:                                                  *)
(*        REVISED : 8 SEPTEMBER 1987, M. H. CHOI, DBMA                 *)
(*                  ADDED MINIMUM OCCURENCE FIELD TO T_ATTRIBUTE       *)
(*        ORIGINATED: 17 SEPTEMBER 1986, M. H. CHOI, DBMA              *)
(*                                                                     *)
(***********************************************************************)
CONST
(* ARBITRARY SIZE PARAMETERS:                                         *)
   MAX_ARRAY_POINTER = 100;
   MAX_ATTRIBUTE     = 100;
   MAX_ENUMERATION   = 100;
   MAX_ENUM_INDEX    = 100;
   MAX_ARRAY_LIST    = 100;
   MAX_ARRAY_INDEX   = 100;
   MAX_CL_LIST       = 100;
   MAX_CL_KINDS      = 100;
TYPE
   T_STRING_8        = PACKED ARRAY (.1..8.) OF CHAR;
   T_DATA_TYPE       = ( INTEGER_DT, REAL_DT, STRING_DT, LOGICAL_DT,
                         EUNM_DT, PNTR_DT, ARRAY_DT );
(*                                                                     *)
   T_ATTRIBUTE = RECORD
      NAME              : T_SCHEMA_NAME;
      MIN_OCC           : INTEGER;
      DATA_TYPE         : INTEGER;
      CASE INTEGER OF
         1, 2, 3, 4     : ( OFFSET           : INTEGER;
                            SIZE             : INTEGER);
         7, 8           : ( POSITION         : INTEGER);
         5, 10          : ( SELECTOR_OFFSET  : INTEGER;
                            TABLE_INX_POSITION : INTEGER);
      END;
(*                                                                     *)
   T_ENUM_INDEX = ARRAY (. 1..MAX_ENUM_INDEX .) OF RECORD
      NO_OF_ENTRIES     : INTEGER;
      TABLE_INX_POSITION : INTEGER;
      END;
(*                                                                     *)
   T_ENUMERATION = ARRAY(.1..MAX_ENUMERATION.) OF T_SCHEMA_NAME;
(*                                                                     *)
   T_ARRAY_INDEX = ARRAY (. 1..MAX_ARRAY_INDEX .) OF RECORD
      NO_OF_DIMENS      : INTEGER;
      TABLE_INX_POSITION : INTEGER;
      END;
(*                                                                     *)
   T_ARRAY_LIST = ARRAY (. 1..MAX_ARRAY_LIST .) OF RECORD
      SIZE              : INTEGER;
      LOW_BOUND         : INTEGER;
      END;
```

```
(*                                                                    *)
   T_CL_INDEX = ARRAY (. 1..MAX_CL_LIST .) OF RECORD
      NO_OF_CL_KINDS    : INTEGER;
      TABLE_INX_POSITION : INTEGER;
      END;
(*                                                                    *)
   T_CL_KINDS = ARRAY(. 1..MAX_CL_KINDS .) OF INTEGER;
(*                                                                    *)
   T_SCHEMA_POINTER   = @T_SCHEMA;
   T_SCHEMA           = RECORD
      NAME             : T_SCHEMA_NAME;
      KIND             : INTEGER;
      ATTRIBUTE_COUNT  : INTEGER;
      ENUM_INDEX_OFFSET : INTEGER;
      ENUM_VALUE_OFFSET : INTEGER;
      ARRAY_INDEX_OFFSET : INTEGER;
      ARRAY_LIST_OFFSET : INTEGER;
      CL_INDEX_OFFSET  : INTEGER;
      CL_KINDS_OFFSET  : INTEGER;
      ATTRIBUTE        : ARRAY (. 1..MAX_ATTRIBUTE .) OF T_ATTRIBUTE;
      END;
(*                                                                    *)
   T_RUN_TIME_POINTER = @T_RUN_TIME;
   T_RUN_TIME     = RECORD
      ENTITY      : T_SCHEMA;
      ENUM_INDEX  : T_ENUM_INDEX;
      ENUM_VALUE  : T_ENUMERATION;
      ARRAY_INDEX : T_ARRAY_INDEX;
      ARRAY_LIST  : T_ARRAY_LIST;
      CL_INDEX    : T_CL_INDEX;
      CL_KINDS    : T_CL_KINDS;
      END;
(*                                                                    *)
   T_KIND_ADB_POINTER = @T_KIND_ADB;
   T_KIND_ADB         = RECORD
      SYSTEM_AREA     : T_SCH_INST_ENT;
      RUN_TIME        : T_RUN_TIME;
      END;
(*                                                                    *)
   T_ARRAY_LIST_COMPACTOR = RECORD
      TABLE           : T_ARRAY_LIST;
      TABLE_SIZE      : INTEGER;
      END;
(*                                                                    *)
   T_ARRAY_INX_COMPACTOR = RECORD
      TABLE           : T_ARRAY_INDEX;
      TABLE_SIZE      : INTEGER;
      END;
(*                                                                    *)
```

```
   T_ENUM_INX_COMPACTOR = RECORD
      TABLE          : T_ENUM_INDEX;
      TABLE_SIZE     : INTEGER;
      END;
(*                                                                    *)
   T_ENUM_COMPACTOR = RECORD
      TABLE          : T_ENUMERATION;
      TABLE_SIZE     : INTEGER;
      END;
(*                                                                    *)
   T_CL_INX_COMPACTOR = RECORD
      TABLE          : T_CL_INDEX;
      TABLE_SIZE     : INTEGER;
      END;
(*                                                                    *)
   T_CL_KINDS_COMPACTOR = RECORD
      TABLE          : T_CL_KINDS;
      TABLE_SIZE     : INTEGER;
      END;
(*                                                                    *)
   T_DATA_VALUE = ARRAY (. 1..MAX_ATTRIBUTE_VALUE .) OF CHAR;
(*                                                                    *)
   T_VARIANT_POINTER            = RECORD
      CASE INTEGER OF
         0 : ( AS_ADB_SIZE      :  LISTPSTN          );
         1 : ( AS_INTEGER       :  INTEGER           );
         2 : ( TO_ATTRIBUTE_VALUE : @T_VALUE         );
         3 : ( TO_ENTITY        :  ENTPNTR           );
         4 : ( TO_ENUMERATION   : @T_ENUMERATION     );
         5 : ( TO_SCHEMA        :  T_SCHEMA_POINTER  );
         6 : ( TO_SELECTOR      : @T_SELECTOR        );
         7 : ( TO_ENUM_INDEX    : @T_ENUM_INDEX      );
         8 : ( AS_RUN_TIME_POINTER : T_RUN_TIME_POINTER );
         9 : ( AS_DATA          : @T_DATA_VALUE      );
        10 : ( TO_ARRAY_INDEX   : @T_ARRAY_INDEX     );
        11 : ( TO_ARRAY_LIST    : @T_ARRAY_LIST      );
        12 : ( TO_CL_INDEX      : @T_CL_INDEX        );
        13 : ( TO_CL_LIST       : @T_CL_KINDS        );
        14 : ( TO_ENTKEY        :  ENTKEY            );
        15 : ( TO_ARRAY_VALUE   : @T_Word            );
        16 : ( TO_CNSTKEY       :  LISTPNTR          );
        17 : ( TO_INTEGER_1     : @T_INTEGER_1 );
        18 : ( TO_INTEGER_2     : @T_INTEGER_2 );
        19 : ( TO_INTEGER_4     : @T_INTEGER_4 );
        20 : ( TO_REAL_4        : @SHORTREAL );
        21 : ( TO_REAL_8        : @REAL      );
        22 : ( TO_value         : T_VALUE         );
      END;
(*                                                                    *)
```

```
(*                                                                    *)
CONST
    MAX_BUFFER      = SIZEOF ( T_RUN_TIME );
(*                                                                    *)
(* END %INCLUDE RTSTYP ************************************************)
```

```
%PRINT OFF
(*------------------------------------------------------------------------*)
(*  CONSTANTS USED BY THE SCHEMA MANAGER PACKAGE                          *)
(*------------------------------------------------------------------------*)

CONST

(*------------------------------------------------------------------------*)
(*  BLANK CONSTANTS                                                       *)
(*------------------------------------------------------------------------*)

   BLANK8  = '        ';
   BLANK16 = '                ';
   BLANK30 = '                              ';
   BLANK40 = '                                        ';
   BLANK50 = '                                                  ';
   BLANK_IDENTIFIER = '               ';
   BLANK_TABLE_VARIABLE = '                       ';

(*------------------------------------------------------------------------*)
(*  CONSTANT LENGTHS                                                      *)
(*------------------------------------------------------------------------*)

   IDENTIFIER_LENGTH = 16;
   UNIQUENESS_LENGTH = 14;
   TABLE_VARIABLE_LENGTH = IDENTIFIER_LENGTH + 7;

(*------------------------------------------------------------------------*)
(*  CONSTANT LENGTH OF THE ENTITY ADB                                     *)
(*------------------------------------------------------------------------*)

   ENTITY_ADB_CONST_LENGTH = 28;

(*------------------------------------------------------------------------*)
(*  MINIMUMS AND MAXIMUMS                                                 *)
(*------------------------------------------------------------------------*)

   MINIMUM_INTEGER_PRECISION = 1;
   MINIMUM_REAL_PRECISION    = 1;
   MINIMUM_STRING_LENGTH     = 1;

   MAXIMUM_INTEGER_PRECISION =  9;
   MAXIMUM_REAL_PRECISION    = 16;
   MAXIMUM_STRING_LENGTH     = 1000;

   MINIMUM_ARRAY_BOUND       = -99;
   MAXIMUM_ARRAY_BOUND       = 999;

   MAX_ARRAY_SIZE            = 1000;
```

```
(*-------------------------------------------------------------------*)
(*  ENTITY KIND NUMBERS                                              *)
(*-------------------------------------------------------------------*)

   PRIMITIVE_KIND     = 1000;
   INTEGER_KIND       = 1001;
   REAL_KIND          = 1002;
   STRING_KIND        = 1003;
   LOGICAL_KIND       = 1004;
   ENUMERATION_KIND   = 1005;
   ENUMERITEM_KIND    = 1006;
   POINTER_KIND       = 1008;
   STRUCTURE_KIND     = 1009;
   ARRAY_KIND         = 1010;
   SUPERTYPE_KIND     = 1016;
   CLASS_KIND         = 1017;
   ENTITY_KIND        = 1018;
   FIELD_KIND         = 1019;
   DEFINED_TYPE_KIND  = 1020;
   SUBSCHEMA_KIND     = 1021;
   GLOBAL_FIELD_KIND  = 1022;
   BACKPATCH_KIND     = 1023;
   UNRESOLVED_KIND    = 1024;
   SCHEMA_KIND        = 1089;

(*-------------------------------------------------------------------*)
(*  CONCEPTUAL SCHEMA REPORT CONSTANTS                               *)
(*-------------------------------------------------------------------*)

(*-------------------------------------------------------------------*)
(*  THE MAXIMUM PAGE SIZE                                            *)
(*-------------------------------------------------------------------*)
   MAX_PAGE_SIZE = 55;

(*-------------------------------------------------------------------*)
(*  RUN-TIME SUBSCHEMA CONSTANTS                                     *)
(*-------------------------------------------------------------------*)

   MAX_ATTRIBUTE_VALUE = 1000;
   MAX_CHARS           = 1000;
   MAX_LIST            = 4000000;
   MAX_RDB_SIZE        = 65535;
   MAX_WORDS           = MAX_CHARS DIV 4;
   SCHEMA_NAME_SIZE    = 16;

%PRINT OFF
```

```
%PRINT OFF
(*-------------------------------------------------------------------*)
(*  INCLUDES FOR THE SCHEMA EXECUTIVE                                *)
(*-------------------------------------------------------------------*)

%INCLUDE SCECON;
%INCLUDE APLTYP;

(*-------------------------------------------------------------------*)
(*   TYPES USED BY THE SCHEMA MANAGER PACKAGE                        *)
(*-------------------------------------------------------------------*)

TYPE

(*-------------------------------------------------------------------*)
(*   ENTITY PRIMITIVE TYPES                                          *)
(*-------------------------------------------------------------------*)

   ENTITY_TYPE = (INT,REEL,STRNG,LOGICAL,ARAY,LIS,SETT,ENUM,
                  ENUMITM,POINTR,STRUC,DEF_TYP,CLASS,SUPERTYPE,
                  ENTITY,FIELD,GBLFLD,SUBSCHEMA,SCHEMA,
                  BACK_PATCH,UNRESOLVED);

(*-------------------------------------------------------------------*)
(*   TRANSACTION TYPES                                               *)
(*-------------------------------------------------------------------*)

   TRANS_TYPE = (T_SUBSCMA,    T_CLASS,     T_ENTITY,     T_GBLFLD,
                 T_FIELD,      T_STRUC,     T_ARRAY,  (* T_LIST,
                 T_SET,   *)   T_DEF_TYP,   T_DEF_TYP2,   T_ENUM,
                 T_ENUMITEM,   T_INTEGER,   T_REAL,       T_LOGICAL,
                 T_STRING,     T_POINTER,   T_FIND_KEY,   T_PASS_KEY,
                 T_SUPERTYPE,  T_SUPERTYPE2,  T_END_SUPERTYPE,
                 T_END_STRUC,  T_END_ENUM,  T_END_POINTER,
                 T_END_DEF_TYP, T_END_ENTITY, T_END_CLASS,
                 T_END_SUBSCMA, T_END_GBLFLD, T_END_PROCESSING,
                 T_UNRESOLVED);

(*-------------------------------------------------------------------*)
(*   NAME TYPE FOR THE ENTITIES                                      *)
(*-------------------------------------------------------------------*)

   T_NAME = PACKED ARRAY(.1..16.) OF CHAR;

   CHAR50  = PACKED ARRAY(.1..50.) OF CHAR;

   CHAR150 = PACKED ARRAY(.1..150.) OF CHAR;
```

```
(*-------------------------------------------------------------------*)
(*   ENUMERATED TYPES FOR THE RECORD DECLARATIONS                    *)
(*-------------------------------------------------------------------*)

   T_STATUS = (FROZEN,UNFRZN);

   T_PURPS = (KEY,ROLE);

   T_REQ = (REQ,OPT);

   T_DEP = (DEP,IND);

   F_KEY_REC = RECORD
      REC_KIND : INTEGER;
(*    CASE REC_KIND OF    *)
      END;

(*-------------------------------------------------------------------*)
(*   BACKPATCH RECORD TYPE                                           *)
(*-------------------------------------------------------------------*)

   BACKPATCH_REC = RECORD
      NAME : T_NAME;
      KIND : INTEGER;
      END;

(*-------------------------------------------------------------------*)
(*   UNRESOLVED RECORD TYPE                                          *)
(*-------------------------------------------------------------------*)

   UNRESOLVED_REC = RECORD
      NAME : T_NAME;
      KIND : INTEGER;
      END;

(*-------------------------------------------------------------------*)
(*   SCHEMA RECORD TYPE                                              *)
(*-------------------------------------------------------------------*)

   SCMA_REC = RECORD
      NAME : T_NAME;
      VERSION : INTEGER;
      DATE : ARRAY(.1..9.) OF CHAR;
      STATUS : T_STATUS
      END;
```

```
(*-------------------------------------------------------------------*)
(*   SUBSCHEMA TYPE                                                  *)
(*-------------------------------------------------------------------*)

   SSCMA_REC = RECORD
      NAME     : T_NAME;
      COMMENT  : CHAR150;
      OFFSET   : INTEGER;
      PHYSICAL : BOOLEAN;
      END;

(*-------------------------------------------------------------------*)
(*   CLASS RECORD TYPE                                               *)
(*-------------------------------------------------------------------*)

   CLASS_REC = RECORD
      NAME : T_NAME;
      KIND : INTEGER;
      COMMENT : CHAR150;
      END;

(*-------------------------------------------------------------------*)
(*   ENTITY RECORD TYPE                                              *)
(*-------------------------------------------------------------------*)

   ENTITY_REC = RECORD
      NAME     : T_NAME;
      KIND     : INTEGER;
      COMMENT  : CHAR150;
      PHYSICAL : BOOLEAN;
      END;

(*-------------------------------------------------------------------*)
(*   FIELD RECORD TYPE                                               *)
(*-------------------------------------------------------------------*)

   FIELD_REC = RECORD
      NAME     : T_NAME;
      POS      : INTEGER;
   (* PURPS    : T_PURPS; *)
      REQD     : T_REQ;
   (* DEPND    : T_DEP;   *)
      COMMENT  : CHAR50;
      CASE INTEGER OF
        0 : (OFFSET             : INTEGER;
             SIZE               : INTEGER);
        1 : (POSITION           : INTEGER);
        2 : (SELECTOR_OFFSET    : INTEGER;
             TABLE_INX_POSITION : INTEGER);
      END;
```

```
(*-------------------------------------------------------------------*)
(*   ARRAY RECORD TYPE                                               *)
(*-------------------------------------------------------------------*)

   ARAY_REC = RECORD
      LO_BOUND   : INTEGER;
      HI_BOUND   : INTEGER;
      MIN_OCCUR  : INTEGER;
      ARRAY_TYPE : ENTITY_TYPE;
      END;

(*-------------------------------------------------------------------*)
(*   LIST RECORD TYPE                                                *)
(*-------------------------------------------------------------------*)

(* LIST_REC = RECORD
      MINIMUM : INTEGER;
      MAXIMUM : INTEGER;
      END;                *)

(*-------------------------------------------------------------------*)
(*   SET RECORD TYPE                                                 *)
(*-------------------------------------------------------------------*)

(* SET_REC = RECORD
      MINIMUM : INTEGER;
      MAXIMUM : INTEGER;
      END;                *)

(*-------------------------------------------------------------------*)
(*   ENTITY ATTRIBUTE DATA BLOCK TYPE                                *)
(*-------------------------------------------------------------------*)

   ENTITY_ADB = RECORD
      ENT_KIND  : INTEGER;
      LENGTH    : INTEGER;
      SYSUSE    : INTEGER;
      VERSION   : INTEGER;
      SYS_IDENT : INTEGER;
      IDENT     : INTEGER;
      TYP       : ENTITY_TYPE;
      DUMMY     : ARRAY(.1..3.) OF CHAR;
      CASE TYP  : OF
         INT        :  (I_PRECISION    : INTEGER       );
         REEL       :  (R_PRECISION    : INTEGER       );
         STRNG      :  (S_LENGTH       : INTEGER       );
         LOGICAL    :  ();
         ARAY       :  (ARY            : ARAY_REC      );
    (*   LIS        :  (LST            : LIST_REC      );   *)
```

```
    (*   SETT        :  (SETS           : SET_REC       );   *)
         ENUM        :  ();
         ENUMITM     :  (ENUMITM_NAME   : T_NAME        );
         POINTR      :  ();
         STRUC       :  ();
         DEF_TYP     :  (DEF_TYP_NAME   : T_NAME        );
         CLASS       :  (CLS            : CLASS_REC     );
         SUPERTYPE   :  (SUPERTYPE_NAME : T_NAME        );
         ENTITY      :  (ENT            : ENTITY_REC    );
         FIELD       :  (FLD            : FIELD_REC     );
         GBLFLD      :  ();
         SUBSCHEMA   :  (SSCMA          : SSCMA_REC     );
         SCHEMA      :  (SCMA           : SCMA_REC      );
         BACK_PATCH  :  (BACKPATCH      : BACKPATCH_REC );
         UNRESOLVED  :  (UNRESOLVE      : UNRESOLVED_REC);
      END;

   ENTBLOCK = ENTITY_ADB;

(*-------------------------------------------------------------------------*)
(*   TRANSACTION TYPE                                                      *)
(*-------------------------------------------------------------------------*)

   TRANSACTION = RECORD
      T_TYP : TRANS_TYPE;
      CASE T_TYP : OF
         T_SUBSCMA      :  (SSCMA          : SSCMA_REC     );
         T_CLASS        :  (CLS            : CLASS_REC     );
         T_SUPERTYPE,
         T_SUPERTYPE2   :  (SUPERTYPE_NAME : T_NAME        );
         T_ENTITY       :  (ENT            : ENTITY_REC    );
         T_GBLFLD       :  ();
         T_FIELD        :  (FLD            : FIELD_REC     );
         T_STRUC        :  ();
         T_ARRAY        :  (ARY            : ARAY_REC      );
    (*   T_LIST         :  (LST            : LIST_REC      );   *)
    (*   T_SET          :  (SETS           : SET_REC       );   *)
         T_DEF_TYP,
         T_DEF_TYP2,
         T_END_DEF_TYP  :  (DEFTYP_NAME    : T_NAME        );
         T_ENUM         :  ();
         T_ENUMITEM     :  (ENUMITEM_NAME  : T_NAME        );
         T_INTEGER      :  (I_PRECISION    : INTEGER       );
         T_REAL         :  (R_PRECISION    : INTEGER       );
         T_LOGICAL      :  ();
         T_STRING       :  (S_LENGTH       : INTEGER       );
         T_POINTER      :  ();
         T_FIND_KEY     :  (FNDKEY         : F_KEY_REC     );
         T_PASS_KEY     :  (PASSKEY        : ENTKEY        );
```

```
        T_END_STRUC     :  ();
        T_END_ENUM      :  ();
        T_END_POINTER   :  ();
        T_END_ENTITY    :  ();
        T_END_CLASS     :  ();
        T_END_SUPERTYPE:  ();
        T_END_SUBSCMA   :  ();
        T_END_GBLFLD    :  ();
        T_UNRESOLVED    :  (UNRESOLVE       : UNRESOLVED_REC);
     END;

(*-------------------------------------------------------------------*)
(*   TRANSACTION STACK TYPE                                          *)
(*-------------------------------------------------------------------*)

   TRANSPTR = @T_STACK;

   T_STACK = RECORD
      STACKPTR : TRANSPTR;
      STACKDATA : TRANSACTION
      END;

(*-------------------------------------------------------------------*)
(*   KEY STACK TYPE                                                  *)
(*-------------------------------------------------------------------*)


   KEYPTR = @K_STACK;

   K_STACK = RECORD
      K_STACKPTR : KEYPTR;
      KEY_DATA : LISTKEY
      END;

(*-------------------------------------------------------------------*)
(*   ERROR TYPES                                                     *)
(*-------------------------------------------------------------------*)

   T_SCE_ERROR =
        (OK,
         DIALOG_FAILED,
         TRANSACTION_PROCESSING_ERROR,
         STACK_UNDERFLOW,
         MAS_ROUTINE_ERROR,
         UNKNOWN_TYPE,
         SIZE_OUT_OF_RANGE);
```

```
(*--------------------------------------------------------------------*)
(*   WARNING TYPES                                                    *)
(*--------------------------------------------------------------------*)

   T_SCE_WARNING =
        (OKW,
         INVALID_KEY_RETURNED,
         INVALID_CASE_OPTION,
         MAX_ARRAY_SIZE_EXCEEDED,
         CANNOT_UPDATE,
         NO_ARRAY_ENTITY,
         NOT_ENOUGH_ROOM);

(*--------------------------------------------------------------------*)
(*   ERROR RETURN CODE TYPES                                          *)
(*--------------------------------------------------------------------*)

   RETURN_CODE = RECORD
      ERROR : T_SCE_ERROR;
      WARNING : T_SCE_WARNING;
      END;

   RET_REC = RECORD
      RC : RETURN_CODE;
      ROUT_NAME : STRING(8)
      END;

(*--------------------------------------------------------------------*)
(*   CHARACTER STRING TYPES                                           *)
(*--------------------------------------------------------------------*)

   CHAR8   = PACKED ARRAY(.1..8.) OF CHAR;

   CHAR9   = PACKED ARRAY(.1..9.) OF CHAR;

   CHAR12  = PACKED ARRAY(.1..12.) OF CHAR;

   CHAR16  = PACKED ARRAY(.1..16.) OF CHAR;

   CHAR23  = PACKED ARRAY(.1..23.) OF CHAR;

   CHAR30  = PACKED ARRAY(.1..30.) OF CHAR;

   CHAR40  = PACKED ARRAY(.1..40.) OF CHAR;

   IDCHAR  = PACKED ARRAY (.1..IDENTIFIER_LENGTH.) OF CHAR;

   TABCHAR = PACKED ARRAY (.1..TABLE_VARIABLE_LENGTH.) OF CHAR;

   CRTABCHAR = PACKED ARRAY (.1..40.) OF CHAR;
```

```
(*-------------------------------------------------------------------*)
(*   ARRAY OF ENTITY TYPES                                           *)
(*-------------------------------------------------------------------*)

   T_ARRAY16 = ARRAY(.1..MAX_ARRAY_SIZE.) OF CHAR16;

   T_ARRAY23 = ARRAY(.1..MAX_ARRAY_SIZE.) OF CHAR23;

   T_ARRAYID = ARRAY(.1..MAX_ARRAY_SIZE.) OF IDCHAR;

   T_ARRAYTV = ARRAY(.1..MAX_ARRAY_SIZE.) OF TABCHAR;

   T_ARRAYRV = ARRAY(.1..MAX_ARRAY_SIZE.) OF CRTABCHAR;

(*-------------------------------------------------------------------*)
(*   MESSAGE TYPE                                                    *)
(*-------------------------------------------------------------------*)

   MESSAGE = CHAR8;

(*-------------------------------------------------------------------*)
(*   OPERATION TYPE                                                  *)
(*-------------------------------------------------------------------*)

   OPERATIONS = (EDIT,REPORT,S_FILE,EXIT,S_RETURN,CR_SUB,CR_CLASS,
                 CR_SUPR,CR_ENT,CR_DEF,CR_GBLFLD,UP_SUB,UP_CLASS,UP_ENT,
                 UP_SUPR,UP_GBLFLD,UP_LOCAL,REVIEW,ACCEPT,DISPLAY,
                 LIST,NO_MORE,INCLDS,DATA_DIC,RUNTIME,
                 CONCEPTUAL_SCHEMA,CROSS_REFER,UNDEFINED,
                 ADD,REMOVE,DELETE,UPDATE,MULTIPLE_SEL,SAVE,
                 RETRIEVE,UP_DEF,REV_ENT,REV_DEF,REV_SUPR,REV_GBLFLD,
                 REV_SUB,REV_CLASS,REV_LOCAL,DEFAULT,CR_OPTION_ONE,
                 CR_OPTION_TWO,CR_OPTION_THREE,CR_OPTION_FOUR,
                 CR_OPTION_FIVE,CR_OPTION_SIX,CR_OPTION_SEVEN,
                 CR_OPTION_EIGHT,CR_OPTION_NINE,CR_OPTION_TEN,
                 PHYSICAL_SUBSCHEMA);

(*-------------------------------------------------------------------*)
(*   FIELD TYPE INDICATES THE TYPE OF THE USER OF THE FIELD          *)
(*-------------------------------------------------------------------*)

   T_FIELDTYPE = (ENTITE,GLOBAL,STRUCTURE,SUPR);

(*-------------------------------------------------------------------*)
(*   FIELD DEFINITION TYPE                                           *)
(*-------------------------------------------------------------------*)

   T_FIELD_DEF = (IN_ADB,IN_KEYBLOCK,IN_CNST_REC);
```

```
(*-------------------------------------------------------------------*)
(*   MATCH RECORD INDICATES IF THE NAME AND/OR KIND NUMBER IS        *)
(*   IDENTICAL                                                       *)
(*-------------------------------------------------------------------*)

   MATCH = RECORD
     NAME : BOOLEAN;
     NUMBER : BOOLEAN
   END;

(*-------------------------------------------------------------------*)
(*   WITHIN RECORD INDICATES IF RETURN OR EXIT WAS CHOSEN WITHIN     *)
(*   ANOTHER ROUTINE                                                 *)
(*-------------------------------------------------------------------*)

   WITHIN = RECORD
     ROUTINE : BOOLEAN;
     RET : BOOLEAN;
     XIT : BOOLEAN
   END;

(*-------------------------------------------------------------------*)
(*  TYPES FOR THE CONCEPTUAL SCHEMA REPORT                           *)
(*-------------------------------------------------------------------*)

   PAGES = (CLASS_PAGE,
            DEFINED_TYPE_PAGE,
            SUPERTYPE_PAGE,
            ENTITY_PAGE,
            GLOBAL_FIELD_PAGE,
            SUBSCHEMA_PAGE);

   HEADING_TYPE = (DEFINITION,
                   INDEX);

   PAGE_PTR = @PAGE_REC;

   PAGE_REC = RECORD
     PAGE_NUMBER : INTEGER;
     NEXT_PAGE   : PAGE_PTR
   END;

(*-------------------------------------------------------------------*)
(*  'PAGE_PTR' IS USED TO LINK TOGETHER THE PAGE NUMBER OF THOSE     *)
(*  PAGES THAT BEGIN A NEW ENTITY, CLASS, OR SUBSCHEMA.  WHEN THE    *)
(*  INDICES ARE PRINTED OUT, THE CHAIN OF PAGE NUMBERS CREATED IS    *)
(*  USED.                                                            *)
(*-------------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------------*)
(*  TYPES FOR THE RUN-TIME SUBSCHEMA                                 *)
(*-------------------------------------------------------------------*)

   T_SCHEMA_NAME     = PACKED ARRAY (. 1..SCHEMA_NAME_SIZE .) OF CHAR;
   T_ATTRIBUTE_VALUE = ARRAY (. 1..MAX_ATTRIBUTE_VALUE .) OF CHAR;
   T_PS_ORDER        = ARRAY (. 1..100 .) OF INTEGER;
   T_HEX_BYTE        = PACKED 0..255;
   T_HEX_WORD        = ARRAY (. 1..4 .) OF T_HEX_BYTE;
   T_SELECTOR        = PACKED 0..9;
   T_INTEGER_1       = PACKED -128..127;
   T_INTEGER_2       = PACKED -32768..32767;
   T_INTEGER_4       = MININT..MAXINT;
   T_VALUE           = ARRAY(.1..MAX_ATTRIBUTE_VALUE.) OF CHAR;
   T_WORD            = ARRAY(.1..4.) OF CHAR;

   ENTPNTR = @ENTBLOCK;

   LISTPNTR = @T_SYS_LIST;

   RDBSIZE = PACKED 0..MAX_RDB_SIZE;

   T_SYS_LIST = RECORD
      SIZE   : LISTSIZE;
      LSTLNG : LISTSIZE;
      LIST   : ARRAY (. 1..MAX_LIST .) OF ENTKEY;
      END;

   T_SCH_INST_ENT = RECORD
      KIND     : ORD_KIND;
      POSITION : LISTPSTN;
      NUM_GROUP: LISTPSTN;
      MIN_CNST : LISTPSTN;
      END;

   T_ENTITY_INDEX = ARRAY (. 1..1000 .) OF RECORD
      NAME      : T_SCHEMA_NAME;
      KIND      : INTEGER;
      END;
```