

NO-A182 654

INTEGRATED INFORMATION SUPPORT SYSTEM (IIS) VOLUME 8

1/1

USER INTERFACE SUBS. (U) GENERAL ELECTRIC CO

SCHENECTADY NY PRODUCTION RESOURCES CONSU.

UNCLASSIFIED

M BUTTERNORTH 01 NOV 85 UTP-620144502

F/G 12/5

NL

END
8 87
DTIC



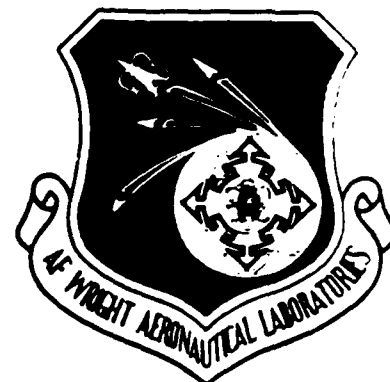
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

016 FILE COPY

(2)

AFWAL-TR-86-4006
Volume VIII
Part 27

AD-A182 654



INTEGRATED INFORMATION
SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 27 - Rapid Application Generator Unit Test Plan

General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345

DTIC
ELECTE
S JUL 20 1987 D
CS D

Final Report for Period 22 September 1980 - 31 July 1985
November 1985

Approved for public release; distribution is unlimited.

MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6533

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

A182 654

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE				
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S) AFVAL-TR-86-4006 Vol VIII, Part 27	
6a. NAME OF PERFORMING ORGANIZATION General Electric Company Production Resources Consulting		6b. OFFICE SYMBOL (If applicable) AFVAL/MLTC	7a. NAME OF MONITORING ORGANIZATION AFVAL/MLTC	
6c. ADDRESS (City, State and ZIP Code) 1 River Road Schenectady, NY 12345			7b. ADDRESS (City, State and ZIP Code) WPAFB, OH 45433-6535	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Materials Laboratory Air Force Systems Command, USAF		8b. OFFICE SYMBOL (If applicable) AFVAL/MLTC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-80-C-5155	
8c. ADDRESS (City, State and ZIP Code) Wright-Patterson AFB, Ohio 45435			10. SOURCE OF FUNDING NOS.	
			PROGRAM ELEMENT NO. 78011F	PROJECT NO. 7500
			TASK NO. 62	WORK UNIT NO. 01
11. TITLE (Include Security Classification) (See Reverse)				
12. PERSONAL AUTHOR(S) Butterworth, Manning				
13a. TYPE OF REPORT Final Technical Report		13b. TIME COVERED 22 Sept 1980 - 31 July 1985		14. DATE OF REPORT (Yr., Mo., Day) 1985 November
15. PAGE COUNT 80				
16. SUPPLEMENTARY NOTATION ICAM Project Priority 6201			The computer software contained herein are theoretical and/or references that in no way reflect Air Force-owned or -developed computer software.	
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB GR		
1308	0905			
19. ABSTRACT (Continue on reverse if necessary and identify by block number)				
This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Rapid Application Generator (RAP). The RAP is used to translate application definitions written with any text editor into programs that access data bases via the Common Data Model and manipulate the extracted data in a way determined interactively by the user. <i>Computer program for the manufacturing system analysis.</i>				
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐			21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson			22b. TELEPHONE NUMBER (Include Area Code) 513-255-6976	22c. OFFICE SYMBOL AFVAL/MLTC

DD FORM 1473, 83 APR

EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

11. Title

Integrated Information Support System (IISS)
Vol VIII - User Interface Subsystem
Part 27 - Rapid Application Generator Unit Test Plan

A S D 86 0041
9 Jan 1986



Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

PREFACE

This unit test plan covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Alan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

TASK 4.2

<u>Subcontractors</u>	<u>Role</u>
Boeing Military Aircraft Company (BMAC)	Reviewer.
D. Appleton Company (DACOM)	Responsible for IDEF support, state-of-the-art literature search.
General Dynamics/ Ft. Worth	Responsible for factory view function and information models.

<u>Subcontractors</u>	<u>Role</u>
Illinois Institute of Technology	Responsible for factory view function research (IITRI) and information models of small and medium-size business.
North American Rockwell	Reviewer.
Northrop Corporation	Responsible for factory view function and information models.
Pritsker and Associates	Responsible for IDEF2 support.
SofTech	Responsible for IDEFO support.

TASKS 4.3 - 4.9 (TEST BED)

<u>Subcontractors</u>	<u>Role</u>
Boeing Military Aircraft Company (BMAC)	Responsible for consultation on applications of the technology and on IBM computer technology.
Computer Technology Associates (CTA)	Assisted in the areas of communications systems, system design and integration methodology, and design of the Network Transaction Manager.
Control Data Corporation (CDC)	Responsible for the Common Data Model (CDM) implementation and part of the CDM design (shared with DACOM).
D. Appleton Company (DACOM)	Responsible for the overall CDM Subsystem design integration and test plan, as well as part of the design of the CDM (shared with CDC). DACOM also developed the Integration Methodology and did the schema mappings for the Application Subsystems.

<u>Subcontractors</u>	<u>Role</u>
Digital Equipment Corporation (DEC)	Consulting and support of the performance testing and on DEC software and computer systems operation.
McDonnell Douglas Automation Company (McAuto)	Responsible for the support and enhancements to the Network Transaction Manager Subsystem during 1984/1985 period.
On-Line Software International (OSI)	Responsible for programming the Communications Subsystem on the IBM and for consulting on the IBM.
Rath and Strong Systems Products (RSSP) (In 1985 became McCormack & Dodge)	Responsible for assistance in the implementation and use of the MRP II package (PIOS) that they supplied.
SofTech, Inc.	Responsible for the design and implementation of the Network Transaction Manager (NTM) in 1981/1984 period.
Software Performance Engineering (SPE)	Responsible for directing the work on performance evaluation and analysis.
Structural Dynamics Research Corporation (SDRC)	Responsible for the User Interface and Virtual Terminal Interface Subsystems.

Other prime contractors under other projects who have contributed to Test Bed Technology, their contributing activities and responsible projects are as follows:

<u>Contractors</u>	<u>ICAM Project</u>	<u>Contributing Activities</u>
Boeing Military Aircraft Company (BMAC)	1701, 2201, 2202	Enhancements for IBM node use. Technology Transfer to Integrated Sheet Metal Center (ISMC).

UTP620144502
1 November 1985

<u>Contractors</u>	<u>ICAM Project</u>	<u>Contributing Activities</u>
Control Data Corporation (CDC)	1502, 1701	IISS enhancements to Common Data Model Processor (CDMP).
D. Appleton Company (DACOM)	1502	IISS enhancements to Integration Methodology.
General Electric	1502	Operation of the Test Bed and communications equipment.
Hughes Aircraft Company (HAC)	1701	Test Bed enhancements.
Structural Dynamics Research Corporation (SDRC)	1502, 1701, 1703	IISS enhancements to User Interface/Virtual Terminal Interface (UI/VTI).
Systran	1502	Test Bed enhancements. Operation of Test Bed.

TABLE OF CONTENTS

	<u>Page</u>
SECTION 1.0 GENERAL	1-1
1.1 Purpose	1-1
1.2 Project References	1-1
1.3 Terms and Abbreviations	1-1
SECTION 2.0 DEVELOPMENT ACTIVITY	2-1
2.1 Statement of Pretest Activity	2-1
2.2 Pretest Activity Results	2-1
SECTION 3.0 SYSTEM DESCRIPTION	3-1
3.1 System Description	3-1
3.2 Testing Schedule	3-1
3.3 First Location Testing	3-8
3.4 Subsequent Location Testing	3-8
SECTION 4.0 SPECIFICATIONS AND EVALUATIONS	4-1
4.1 Test Specifications	4-1
4.2 Test Methods and Constraints	4-1
4.3 Test Progression	4-1
4.4 Test Evaluation	4-1
SECTION 5.0 TEST PROCEDURES	5-1
5.1 Test Description	5-1
5.2 Test Control	5-1
5.3 Test Procedures	5-1

APPENDICES

A	TEST APPLICATION TESTAP.FDL	A-1
B	GENERATED C AND COBOL CODE	B-1
C	LIST OF GENERATED BINARY FORM FILES	C-1
D	SCREENS OUTPUT DURING TEST	D-1

FIGURES

3-1	Rapid Application Generator Interfaces	3-2
D-1	Top Level Form	D-1
D-2	Help Form	D-2
D-3	Delete Form	D-3
D-4	Insert Form	D-4
D-5	Modify Form	D-5
D-6	Select Form	D-6

FIGURES (Continued)

D-7	Result of Entry of Y in Verify Item of Delete Form	D-7
D-8	Result of Entry of N in Verify Item of Delete Form	D-8
D-9	Result of Entry of Y in Options Item of Insert Form	D-9
D-10	Result of Entry of X in Rpt45 Item of Testap Form	D-10

SECTION 1

GENERAL

1.1 Purpose

This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Rapid Application Generator known in this document as RAP. The RAP is a configuration item of the Integrated Information Support System (IISS) User Interface (UI).

1.2 Project References

- [1] ICAM Documentation Standards, 15 September 1983, IDS150120000C.
- [2] General Electric Co., System Design Specification, 7 February 1983.
- [3] Structural Dynamics Research Corporation, Application Generator Development Specification, DS 620144502 , 1 November 1985.
- [4] Structural Dynamics Research Corporation, Report Writer Unit Test Plan, UTP620144501 , 1 November 1985.
- [6] Structural Dynamics Research Corporation, Rapid Application Generator Unit Test Plan, UTP621044502 , 1 November 1985.
- [7] Structural Dynamics Research Corporation, Text Editor Unit Test Plan, UTP620144600 , 1 November 1985.
- [8] Structural Dynamics Research Corporation, Form Processor Unit Test Plan, UTP620144200 , 1 November 1985.
- [9] Structural Dynamics Research Corporation, Application Interface Unit Test Plan, UTP621044700 , 1 November 1985.
- [10] Structural Dynamics Research Corporation, Forms Driven Form Editor Unit Test Plan, UTP620144401 , 1 November 1985.

[11] Structural Dynamics Research Corporation, User Interface Services Unit Test Plan, UTP620144100 , 1 November 1985.

[12] Structural Dynamics Research Corporation, Virtual Terminal Unit Test Plan, UTP620144300, 1 November 1985.

1.3 Terms and Abbreviations

Application Definition Language: an extension of the Forms Definition Language that includes retrieval of database information and conditional actions. It is used to define interactive application programs.

Application Generator: (AG), subset of the IISS User Interface that consists of software modules that generate IISS application code and associated form definitions based on a language input. The part of the AG that generates report programs is called the Report Writer. The part of the AG that generates interactive applications is called the Rapid Application Generator.

Application Interface: (AI), subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The AI enables applications to be hosted on computers other than the host of the User Interface.

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Attribute: field characteristic such as blinking, highlighted, black, etc. and various other combinations. Background attributes are defined for forms or windows only. Foreground attributes are defined for items. Attributes may be permanent, i.e., they remain the same unless changed by the application program, or they may be temporary, i.e., they remain in effect until the window is redisplayed.

Common Data Model: (CDM), IISS subsystem that describes common data application process formats, form definitions, etc. of the IISS and includes conceptual schema, external schemas, internal schemas, and schema transformation operators.

Communication Services: allows on host interprocess communication and inter-host communication between the various Test Bed subsystems.

Communication Subsystem: (COMM), IISS subsystem that provides communication services to the Test Bed and subsystems.

Computer Program Configuration Item: (CPCI), an aggregation of computer programs or any of their discrete portions, which satisfies an end-use function.

Conceptual Schema: (CS), the standard definition used for all data in the CDM. It is based on IDEF1 information modelling.

Cursor Position: the position of the cursor after any command is issued.

Device Drivers: (DD), software modules written to handle I/O for a specific kind of terminal. The modules map terminal specific commands and data to a neutral format. Device Drivers are part of the UI Virtual Terminal.

Display List: is similar to the open list, except that it contains only those forms that have been added to the screen and are currently displayed on the screen.

External Schema: (ES), an application's view of the CDM's conceptual schema.

Field: two-dimensional space on a terminal screen.

Field Pointer: indicates the ITEM which contains the current cursor position.

Form: structured view which may be imposed on windows or other forms. A form is composed of fields. These fields may be defined as forms, items, and windows.

Form Definition: (FD), forms definition language after compilation. It is read at runtime by the Form Processor.

Forms Definition Language: (FDL), the language in which electronic forms are defined.

Forms Driven Form Editor: (FD FE), subset of the FE which consists of a forms driven application used to create Form Definition files interactively.

Form Editor: (FE), subset of the IISS User Interface that is used to create definitions of forms. The FE consists of the Forms Driven Form Editor and the Forms Language Compiler.

Form Hierarchy: a graphic representation of the way in which forms, items and windows are related to their parent form.

Forms Language Compiler: (FLAN), subset of the FE that consists of a batch process that accepts a series of forms definition language statements and produces form definition files as output.

Form Processor: (FP), subset of the IISS User Interface that consists of a set of callable execution time routines available to an application program for form processing.

Form processor text editor: (fpte), subset of the form Processor that consists of software modules that provide text editing capabilities to all users of applications that use the Form Processor.

IISS Function Screen: the first screen that is displayed after logon. It allows the user to specify the function he wants to access and the device type and device name on which he is working.

Integrated Information Support System: (IISS), a test computing environment used to investigate, demonstrate and test the concepts of information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Item: non-decomposable area of a form in which hard-coded descriptive text may be placed and the only defined areas where user data may be input/output.

Logical Device: a conceptual device which to an application is indistinguishable from a physical device and is then mapped to part or all of a physical device.

Message: descriptive text which may be returned in the standard message line on the terminal screen. They are used to warn of errors or provide other user information.

Message Line: a line on the terminal screen that is used to display messages.

Network Transaction Manager: (NTM). IISS subsystem that performs the coordination, communication and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

Neutral Data Manipulation Language: (NDML). the command language by which the CDM is accessed for the purpose of extracting, deleting, adding, or modifying data.

Open List: a list of all the forms that have been and are currently open for an application process.

Operating System: (OS), software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

Page: instance of forms in windows that are created whenever a form is added to a window.

Paging and Scrolling: a method which allows a form to contain more data than can be displayed with provisions for viewing any portion of the data buffer.

Physical Device: a hardware terminal.

Presentation Schema: (PS), may be equivalent to a form. It is the view presented to the user of the application.

Qualified Name: the name of a form, item or window preceded by the hierarchy path so that it is uniquely identified.

Rapid Application Generator: (RAP), part of the Application Generator that generates source code for interactive programs based on a language input.

Subform: a form that is used within another form.

Text Editor: (TE), subset of the IISS User Interface that consists of a file editor that is based on the text editing functions built into the Form Processor.

User Data: data which is either input by the user or output by the application programs to items.

User Interface: (UI), IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System (UIDS) and the User Interface Management System (UIMS).

User Interface Development System: (UIDS), collection of IISS User Interface subsystems that are used by applications programmers as they develop IISS applications. The UIDS includes the Form Editor and the Application Generator.

User Interface Management System: (UIMS), the runtime UI. It consists of the Form Processor, Virtual Terminal, Application Interface, the User Interface Services and the Text Editor.

User Interface Monitor: (UIM), part of the Form Processor that handles messaging between the NTM and the UI. It also provides authorization checks and initiates applications.

User Interface Services: (UIS), subset of the IISS User Interface that consists of a package of routines that aid users in controlling their environment. It includes message management, change password, and application definition services.

User Interface/Virtual Terminal Interface: (UI/VTI), another name for the User Interface.

Virtual Terminal: (VT), subset of the IISS User Interface that performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by the UI software which constitutes the virtual terminal definition. Specific terminals are then mapped against the virtual terminal software by specific software modules written for each type of real terminal supported.

UTP620144502
1 November 1985

Virtual Terminal Interface: (VTI), the callable interface to the VT.

Window: dynamic area of a terminal screen on which predefined forms may be placed at run time.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 2

DEVELOPMENT ACTIVITY

2.1 Statement of Pretest Activity

During system development, the Rapid Application Generator was tested progressively. Functionality was incrementally tested and as errors were discovered by this testing, the software was corrected.

This testing was conducted by the program developers in a manual mode. Any errors were noted by the developers and corrections to the program were then made after a testing session.

2.2 Pretest Activity Results

Testing of the RAP revealed numerous flaws which were then corrected and retesting proved successful. Testing included exceptional conditions and error conditions for the language.

SECTION 3

SYSTEM DESCRIPTION

3.1 System Description

The Rapid Application Generator is used to translate application definitions written with any text editor into programs that access data bases via the CDM and manipulate the extracted data in a way determined interactively by the user.

The Application Definition Language in which the application definitions are expressed includes the Forms Definition Language and other statement types. The syntax of the Application Definition Language accepted as input to FLAN is modelled after the Forms Definition Language and the Neutral Data Manipulation Language.

The application definition is input to the RAP which accesses metadata in the CDM to determine the schema definitions of the referenced table field values. The results are a generated C file and Cobol file and separate binary form files for each form definition within the application. The Cobol code output by the RAP is constrained to be compatible with statement forms expected by the CDM precompiler. The C file contains the control logic of the application as well as the interface to the Forms Processor software. This file calls the Cobol procedure which accesses the data in the CDM.

When the Cobol file is precompiled to translate the input SELECT statements to database calls, several procedures are generated each of which must be compiled. In the simplest case, two of the generated Cobol procedures comprise a separate process; the other two after compilation become linked into the application program.

The generated application communicates via the NTM with the generated request process that in turn accesses the CDM. The course of this access is determined by the user's interaction with the application program.

The interface block diagram for the Rapid Application Generator is shown in Figure 3-1.

UTP620144502
1 November 1985

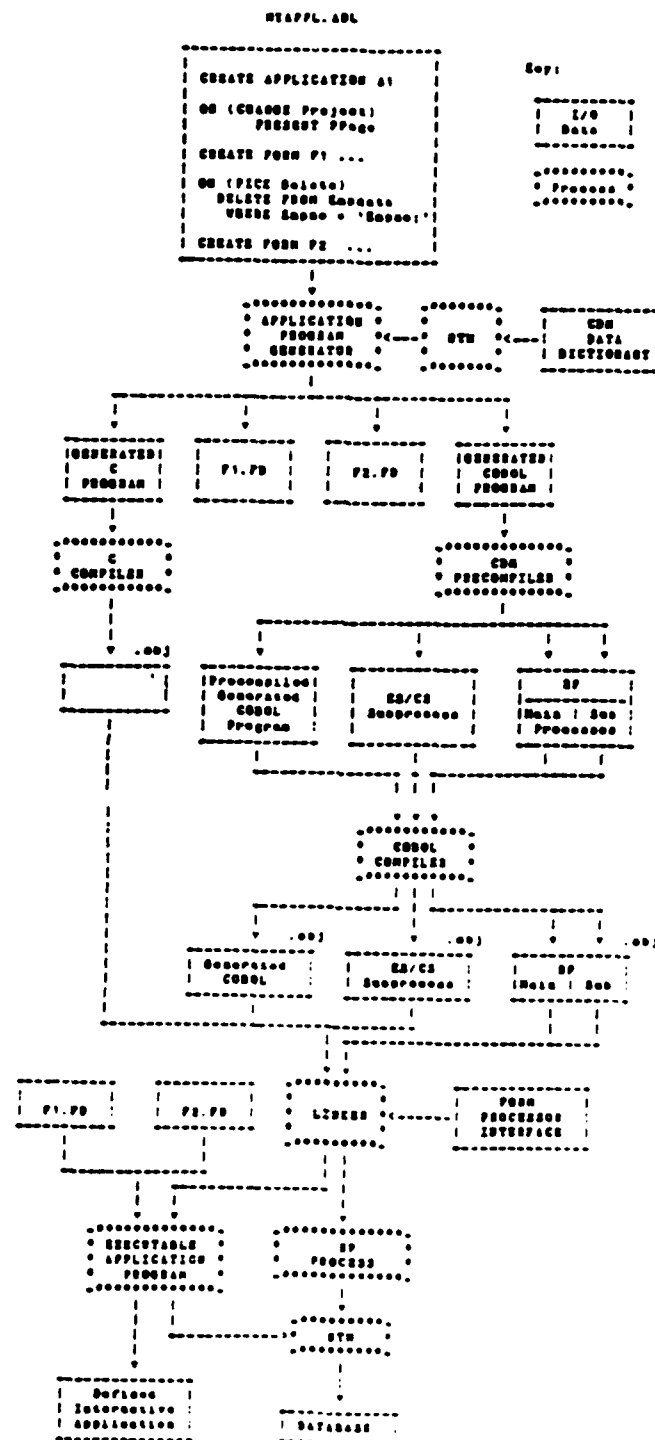


Figure 3-1 Rapid Application Generator Interfaces

3.2 Testing Schedule

The execution of the Rapid Application Generator is dependent upon the CDM and NTM subsystems of IISS and testing of the RAP must be done only after the CDM and NTM have themselves been successfully tested. Since COBOL code generated by the RAP must be precompiled, the precompiler must also be tested prior to testing of the RAP. Finally within the UI subsystem, the RAP uses the Forms Processor and Flan and therefore must be tested only after their successful tests.

3.3 First Location Testing

These tests of the RAP require the following:

Equipment: Air Force VAX, terminals supported by the Virtual Terminal as listed in the UI Terminal Operator Guide.

Support Software: the Integrated Information Support System, Oracle database management system, C compiler, and Cobol compiler.

Personnel: One integrator familiar with the IISS.

Training: The Rapid Application Generator User Manual has been provided with the current release.

Deliverables: The Rapid Application Generator subsystem of the IISS UI/VTI.

Test Materials: This test can be performed using the supplied script file as outlined in this test plan.

Security considerations: None.

3.4 Subsequent Location Testing

The requirements as listed in the previous section must be met. In subsequent tests it will not be necessary to update the UI and NTM databases if the same application names are used. This Unit Test Plan was written for IISS release 2.0 and may become obsolete for future releases.

SECTION 4

SPECIFICATIONS AND EVALUATIONS

4.1 Test Specification

The following functional areas are demonstrated by the outlined tests:

Functional Requirements	Test Activity													
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Switching Between Functions		X	X	X	X									
Change of Field Value														
Overflow of Form										X				
Occurrence of Field Value		X				X	X							
Nonoccurrence of Field Value										X				
Function Key Selection		X	X	X	X									
Menu Picking										X				
Cursor Location											X	X	X	X
Database Delete Operation		not implemented in version 2.0												
Database Insert Operation		not implemented in version 2.0												
Database Modify Operation		not implemented in version 2.0												
Database Select Operation										X				
Record Data Display										X				
Chart Data Display		not implemented in version 2.0												
Icon Selection		not implemented in version 2.0												
Stretchy Lines		not implemented in version 2.0												
Security and Access Control														
Present Form Action		X	X	X	X	X								
Present Conditional Form										X				
Set Field Value Action		X				X	X							
Help String/Help Form Action		X												
Exit Action													X	

- A - entry of Y in Newuser item of top level form (Figs D-1,2)
 B - selection of 'Delete' key on top level form (Fig D-3)
 C - selection of 'Insert' key on top level form (Fig D-4)
 D - selection of 'Modify' key on top level form (Fig D-5)
 E - selection of 'Query' key on top level form (Fig D-6)
 F - entry of Y in Verify item of delete form (Fig D-7)
 G - entry of N in Verify item of delete form (Fig D-8)

H - entry of Y in Options item of insert form (Fig D-9)
I - entry of X in Rpt45 item of query form (Fig D-10)
J - selection of 'Return' key on delete form (Fig D-1)
K - selection of 'Return' key on insert form (Fig D-1)
L - selection of 'Return' key on modify form (Fig D-1)
M - selection of 'Return' key on query form (Fig D-1)
N - selection of 'Quit' key

The preceding chart and the file in Appendix A show the direct correspondence between the test and the functional requirements given above.

4.2 Testing Methods and Constraints

The required input is stated for each test. This testing tests the normal mode of operation of these functions and does not completely exercise all the error combinations that a user of the RAP might create by faulty entry of field information. These tests have been done, however, through the normal testing done by the developer of these functions. No additional constraints are placed on this unit test besides those listed in Section 3.3 of this document.

4.3 Test Progression

The progression of testing of the RAP is fully outlined in Section 5.3 of this unit test plan. This progression should be followed exactly to insure the successful testing of this IISS configuration item.

4.4 Test Evaluation

The complete Rapid Application Generator test consists of many stages each having its associated output. The first stage is the input and processing of the application definition by the generator. The outputs are the generated C and COBOL files which are listed in Appendix B and the binary form files named in Appendix C. The code created during the test should be the same as that appearing in Appendix B. The test should also create all of the binary form files named in Appendix C.

The second stage is the precompilation of the COBOL file. This should successfully produce four COBOL procedures. The names of these procedures and the names of the files containing them are constructed at generation time. The files' names as

well as the success or failure of the precompilation are reported to the test evaluator in another file named according to his choice. The procedure names must be found by looking within the procedures themselves.

The third stage is the compilation and linking of the code which has been created in stages one and two. The respective compilers and linker will report the success or failure of the steps comprising this stage of the test.

The fourth and final stage of the test is the execution of the generated application. The success of this stage will depend upon the successful operation of the NTM, the CDM, and the Form Processor. This stage is automated by the use of a script file supplied with the current release. It is named TESTAP.SCP and is under IISS Configuration Management. Also supplied is the output created by running this script file. This appears under the name of TESTAP.SAV under configuration management.

This output may be compared with that produced when the steps for testing given in Section 5.3 are followed. This stage of the test will be considered successful if this comparison reveals no differences other than differences of dates and times and if the screens output to the terminal are the same (apart from dates and times) as those appearing in Appendix D. (Note: the data in the RPT45 query may not be exactly the same since the contents of the CDM may change between tests; however, the format of the screens should be the same.)

SECTION 5

TEST PROCEDURES

5.1 Test Description

A general description of this unit test was provided in Section 3.

5.2 Test Control

As outlined, this unit test is a semiautomated test which may be done by anyone. Once startup of the NTM has been initiated, no further operator intervention is necessary for that terminal and process. The remaining steps in stages one, two, and three as described in Section 4.4 above are largely manual. Once stage 4 has been initiated, the execution of the test is automatically controlled by the script file. The comparison of the results of stage 4 is manual.

5.3 Test Procedures

To run the unit test plan on a VAX, one must be logged on to an IISS account. The NTM must be up and running and the UI group logical names IISSFLIB, IISSULIB and IISSMLIB must be set properly. IISSFLIB points to the directory containing form definitions (FD files). IISSULIB points to the NTM environment directory since the Application Generator writes the FD files out to the directory and in subsequent running of the application, these FD files are used. IISSMLIB points to the directory containing error messages (MSG files). The steps involved in the RAP test are outlined in detail below.

Steps for Executing the Rapid Application Generator.

Below is an example of how the Rapid Application Generator may be invoked in the VAX/VMS environment. This example requires the use of two terminals. In normal usage of the RAP if the NTM is already running, only one terminal is needed. The steps are numbered sequentially and those that are executed on the first terminal are indicated here with an "A", those on the second terminal are labeled with a "B". The following convention is used to document the procedure.

- Text in angle brackets is to be replaced with appropriate information by the user.
- Single upper case words enclosed in angle brackets represent terminal keys (e.g. <ENTER>).
- Text in upper case is to be entered as shown.

Stage 1 ---

- 1-A Logon on terminal A
2-A \$ SET DEF <to directory containing your NTM environment>
3-A \$ @IISS This brings up the NTM
- 4-B Logon on terminal B
5-B \$ SET DEF <to directory containing your NTM environment>
6-B \$ VT100 This starts up the VT100 device driver. If the User Interface system has been installed at your site with a different device driver, then this step is amended as is appropriate.
- 7-B Fill in the items on the IISS logon screen as follows:
Username: <MORENC>
Password: <STANLEY>
Role : <MANAGER>
Press <ENTER>
- 8-B Fill in the function item on the function screen as follows:
Function: SDAPPGENER
Press <ENTER>
- 9-B Fill in the field on the Rapid Application Generator screen as follows, where [appdir] is the directory where testap.fdl resides:
Application Name: [appdir]testap.fdl
This compiles the application definition, creates the .FD files, and produces both the C code (testapC.C) and the COBOL code (testapX.PRC).

Press <QUIT>

Returns you to the operating system.

Stage 2 ---

10-B \$ @[NDML_dir]NDML

This invokes the NDML precompiler on the test bed and produces as output four or more COBOL files named Xxxxx.TMP:

- database application
- one or more RP-SUB process files
- one or more CS-ES (conceptual_to_external_schema) subroutines
- RP-Main process name

Inside these files are the Yxxxx procedure names.

This step requires an input file containing precompiler directives. Please see the precompiler documentation for the format of this file.

Stage 3 ---

11-B \$ @[NDML_dir]COMPANS (one of the Yxxxx files)

Repeat this step with each of the Xxxxx files that were generated. These steps compile each COBOL file.

12-B \$ LIB/REP GENLIB.OLB *.OBJ

This step inserts the generated object modules in a library named genlib.olb in the user's directory.

13-B \$ CC testapC

This compiles the .C code.

14-B \$ @[NDML_dir]LNKORP <RP-Main process name>

This step links the RP-Main to the RP-Sub to produce an executable.

NOTE: This RP process name is the Yxxxx name.

15-B \$ @LNKAP testap

This step links the compiled .C and .COB modules with the RAP library to produce an executable.

Stage 4 ---

16-B \$ DELETE SEL*.DAT;*

When executed, the generated application creates files named SELn.DAT. These remain after the application terminates. To avoid confusion it is recommended that they be deleted.

17-B \$ UFI

Update the UI database of known applications.

username: <username>

password: <password>

Enter the following line:

INSERT INTO ROLAPP VALUES

(' <role in capital letters> ', 'SDTESTAPZZ');

EXIT

The following steps update the NTM database.

18-B \$ EDIT/EDT ACTTBL.DAT Insert new lines as follows:

TESTAPZZ1

<RP-Main process name padded to 8 characters with Z's>1

19-B \$ EDIT/EDT APITBL.DAT

Insert new lines as follows:

SDTESTAPZZT1V1

GR<RP-Main process name padded to 8 characters with Z's>T1V1

20-B \$ EDIT/EDT APTTBL.DAT

Insert new lines as follows:

TESTAPZZ0599010320000010

RP-Main name padded to 8 characters with
Z's 9999010120001130

21-B \$ VT100

Start up the VT100 device driver.
If the User Interface system has
been installed at your site with
a different device driver, then
this step is amended as is
appropriate.

22-B

Fill in the items on the IISS
logon screen as follows:

Username: <MORENC>
Password: <STANLEY>
Role : <MANAGER>
Press <ENTER>

23-B

Fill in the items on the FUNCTION
screen as follows:

Function: SDDEFINEAP
Press <ENTER>

24-B

Fill in the items on the screen
that is displayed as follows:

report: SDTESTAPZZ
description: APPLICATION GENERATOR
host : VAX
Press <ENTER>
Press <QUIT>

25-B \$ VT100 -RTESTAP.SCP -STESTAP.CMP

Start up the VT100 device driver
with a source script file as
input and a target script file as
output. If the User Interface
system has been installed at your
site with a different device
driver, then this step is amended
as is appropriate.

26-A \$SD

This begins the shutdown of the
NTM.

APPENDIX A

TEST APPLICATION TESTAP.FDL

The following is the file TESTAP.FDL which is the source file for the test application TESTAP. (Due to editing constraints on this document some text strings appear continued on a second line. The actual application definition under IISS Configuration Management does not contain the carriage return embedded within the string.)

CREATE APPLICATION TESTAP

ON (STARTUP)

SET 'Topform.Newuser;' = " "
PRESENT Topform

ON (PICK RETURN & CURSOR IN 'Delform;') PRESENT Topform
ON (PICK RETURN & CURSOR IN 'Insform;') PRESENT Topform
ON (PICK RETURN & CURSOR IN 'Modform;') PRESENT Topform
ON (PICK RETURN & CURSOR IN 'TESTAP;') PRESENT Topform

ON (PICK DELETE) PRESENT Delform
ON (PICK INSERT) PRESENT Insform
ON (PICK MODIFY) PRESENT Modform
ON (PICK QUERY) PRESENT TESTAP

ON ('Delform.Verify;' = "Y")
SET 'Delform.Verify;' = " "
SET 'Delform.Vermmsg;' = "Verification On"
PRESENT Delform

ON ('Delform.Verify;' = "N")
SET 'Delform.Verify;' = " "
SET 'Delform.Vermmsg;' = "Verification Off"
PRESENT Delform

ON ('Insform.Options;' = "Y")PRESENT IOPTS IN 'INSWIN;'

ON ('Topform.Newuser;' = "Y")
SET 'Topform.Newuser;' = " "
HELP TOPHELP

ON (PICK QUIT)EXIT

CREATE FORM Topform

PROMPT CENTER AT ROW 1 AND COL 40

"Structural Dynamics Research Corporation"

PROMPT CENTER AT ROW 2 AND COL 40

"IISS Application Generator Release 2.0"

KEYPAD (KEY 4 = QUIT

KEY 5 = DELETE

KEY 6 = INSERT

KEY 7 = MODIFY

KEY 8 = QUERY)

ITEM Date AT ROW 4 AND COLUMN 12 SIZE 9 VALUE '._DATE'

PROMPT AT LEFT "DATE: " DISPLAY AS TEXT

ITEM Time AT ROW 4 AND COLUMN 62 SIZE 9 VALUE '._TIME'

PROMPT AT LEFT "TIME: " DISPLAY AS TEXT

ITEM NEWUSER AT ROW 6 AND COL 45 SIZE 1 BY 1 DOMAIN

(UPPER) DISPLAY AS INPUT

PROMPT AT LEFT "Do you wish help getting started? (Y/N)"

CREATE FORM TOPHELP

PROMPT AT ROW 6 AND COL 2

"This example illustrates the functionality of the Rapid
Application"

PROMPT AT ROW 7 AND COL 2

"Generator. Only query capability is available in this
release. "

PROMPT AT ROW 9 AND COL 7

". Hit pf4 to return to the top level form"

PROMPT AT ROW 10 AND COL 7

". Hit pf5 to display a DELETE form"

PROMPT AT ROW 11 AND COL 7

". Hit pf6 to display an INSERT form"

PROMPT AT ROW 12 AND COL 7

". Hit pf7 to display a MODIFY form"

PROMPT AT ROW 13 AND COL 7

". Hit pf8 to display a QUERY form"

PROMPT AT ROW 21 AND COL 2

"Hit <enter> to return to the top level form."

CREATE FORM Delform

PROMPT CENTER AT ROW 2 AND COL 40 "Application Generator"
PROMPT AT ROW 3 AND COL 2 "Delete Record Form"
PROMPT AT ROW 4 AND COL 4

-----"
KEYPAD (KEY 4 = RETURN)

ITEM Verify AT ROW 6 AND COL 70 SIZE 1 BY 1
DISPLAY AS INPUT DOMAIN (MUST ENTER UPPER)
PROMPT AT LEFT

"Do you wish to verify before DELETE operation is
committed? (Y/N)"

ITEM Vermsg AT ROW 7 AND COL 5 SIZE 17 DISPLAY AS OUTPUT
PROMPT AT ROW 9 AND COL 38 "Records Retrieved"
FORM Drecords AT ROW 10 AND COL 14 SIZE 64 BY 10
ITEM Delpick(10 VERTICAL WITH 0 SPACES)
AT ROW 10 AND COL 13 SIZE 1 BY 1 DISPLAY AS INPUT

CREATE FORM Drecords
BACKGROUND WHITE

CREATE FORM Insform

PROMPT CENTER AT ROW 2 AND COL 40 "Application Generator"
PROMPT AT ROW 3 AND COL 2 "Insert Record Form"
PROMPT AT ROW 4 AND COL 4

-----"
KEYPAD (KEY 4 = RETURN)

PROMPT CENTER AT ROW 6 AND COL 40 "Records To Be Inserted."
ITEM Options AT ROW 7 AND COL 20 SIZE 1 BY 1
DISPLAY AS INPUT
DOMAIN (UPPER) PROMPT AT LEFT "Options: (Y/N) "
FORM Irecords AT ROW 8 AND COL 2 SIZE 72 BY 10
WINDOW INSWIN AT ROW 20 AND COL 2 SIZE 72 BY 2
BACKGROUND BLACK

CREATE FORM Irecords
BACKGROUND WHITE

CREATE FORM IOPTS CONDITIONAL

ITEM LRECL AT ROW 1 AND COL 20 SIZE 3 BY 1
DISPLAY AS INPUT
PROMPT AT LEFT "Record Length: "
ITEM BLNKS AT ROW 1 AND COL 60 SIZE 1 BY 1
DISPLAY AS INPUT DOMAIN (UPPER)
PROMPT AT LEFT "Strip trailing blanks? (Y/N)"

UTP620144502
1 November 1985

```
CREATE FORM Modform
  PROMPT CENTER AT ROW 2 AND COL 40 "Application Generator"
  PROMPT AT ROW 3 AND COL 2 "Modify Record Form"
  PROMPT AT ROW 4 AND COL 4
"-----"
  KEYPAD (KEY 4 = RETURN)
  PROMPT AT ROW 6 AND COL 38 "Records To Be Modified"
  FORM Mrecords AT ROW 8 AND COL 2 SIZE 72 BY 10
```

```
CREATE FORM Mrecords
  BACKGROUND WHITE
```

```
/*\
***** REPORTS *****/
CREATE FORM TESTAP
  PROMPT CENTER AT ROW 2 AND COL 40 "Application Generator"
  PROMPT AT ROW 3 AND COL 2 "Select Record Form"
  PROMPT AT ROW 4 AND COL 4
"-----"
  KEYPAD (KEY 4 = RETURN)
  PROMPT CENTER AT ROW 5 AND COL 15 "Available Reports"
  ITEM Rpt45 AT 7 20 SIZE 1 PROMPT AT LEFT "RPT45"
    DISPLAY AS INPUT
  ITEM Selrpt AT 8 20 SIZE 1 PROMPT AT LEFT "SELRPT"
    DISPLAY AS INPUT
  PROMPT AT ROW 10 AND COL 25
    "Enter any non-blank character to select a report."
```

```
/*\
*      RPT45
\*/
```

```
ON ('Rpt45;' != " ")
SELECT 'Stype.Dbname;' = DB.DB_NAME
      'Stype.Setid;'   = SM.SET_ID
      'Stype.Rtownid;' = RS.RT_ID_OF_OWNER
      'Stype.Rtmemid;' = SM.RT_ID_OF_MEMBER
      'Stype.Reqopt;'  = SM.REQ_MEM_IND
FROM DATA_BASE DB,
      RECORD_TYPE RT,
      RECORD_SET RS,
      SET_TYPE_MEMBER SM
```

```
WHERE DB.DB_ID = RT.DB_ID AND
      RT.DB_ID = RS.DB_ID AND
      RT.RT_ID = RS.RT_ID OF OWNER AND
      RS.DB_ID = SM.DB_ID AND
      RS.SET_ID = SM.SET_ID
```

/*

```
ORDER BY DB.DB_NAME
        SM.SET_ID
        RS.RT_ID OF OWNER
        SM.RT_ID OF MEMBER
        SM.REQ_MEM_IND
```

*/

PRESENT Page45

```
ON (OVERFLOW BY 'Page45.stype(1)')
    HELP "Press <enter> to continue, <quit> to quit."
```

ON (PICK NTER & CURSOR IN 'Page45;') PRESENT Page45

ON (PICK QUIT & CURSOR IN 'Page45;') PRESENT Testap

CREATE FORM PAGE45

SIZE 80 BY 23

KEYPAD (KEY 0 = NTER KEY 4 = QUIT)

PROMPT AT 1 35 "REPORT OF CODASYL SET TYPES"

PROMPT AT 3 2 "DB_NAME"

PROMPT AT 4 2 "_____"

PROMPT AT 3 13 "SET_ID"

PROMPT AT 4 13 "_____"

PROMPT AT 3 34 "RT_ID_OF_OWNER"

PROMPT AT 4 34 "_____"

PROMPT AT 3 55 "RT_ID_OF_MEMBER"

PROMPT AT 4 55 "_____"

PROMPT AT 3 76 "R/O"

PROMPT AT 4 76 "_____"

ITEM Dummy AT 2 2 SIZE 1 BY 1 DISPLAY AS INPUT

ITEM Pdate DISPLAY AS TEXT AT 1 2 SIZE 10

VALUE ' . TIME'

ITEM Ppage DISPLAY AS TEXT AT 1 70 VALUE ' . PAGENO;'

FORM STYLE (* VERTICAL 0) AT 5 2 SIZE 76 BY 1

UTP620144502
1 November 1985

CREATE FORM Stype

ITEM Ddbname DISPLAY AS OUTPUT AT 1 2 SIZE 10
ITEM Setid DISPLAY AS OUTPUT AT 1 13 SIZE 20
ITEM Rtownid DISPLAY AS OUTPUT AT 1 34 SIZE 20
ITEM Rtmemid DISPLAY AS OUTPUT AT 1 55 SIZE 20
ITEM Reqopt DISPLAY AS OUTPUT AT 1 76 SIZE 1

ON ('Selrpt;' != " ")

SELECT 'Esitem(1).Esname;' = VW.VIEW_NAME
'Esitem(1).Diname;' = VW.DI_ID
'Esitem(1).Dimr;' = VW.ES_TYPE
'Esitem(1).Disize;' = VW.ES_SIZE
'Esitem(1).Dind;' = VW.ES_ND
FROM VIEW META DATA VW
/* ORDER BY VW.VIEW_NAME VW.DI_ID */
PRESENT Esrpt

ON (Overflow BY 'Esrpt.es.esitem(1)')

HELP "Press <enter> to continue, <quit> to quit."

ON (PICK NTER & CURSOR IN 'Esrpt;') PRESENT ESRPT

ON (PICK QUIT & CURSOR IN 'Esrpt;') PRESENT TESTAP

CREATE FORM Esrpt

KEYPAD (KEY 0 = NTER KEY 4 = QUIT)

PROMPT CENTER AT 1 20 "External Schemas in CDM"

PROMPT CENTER AT 2 20 "-----"

ITEM Dummy AT 1 2 SIZE 1 BY 1 DISPLAY AS INPUT

ITEM Pagenum AT 1 70 DISPLAY AS TEXT VALUE '._PAGENO;'

ITEM Today RIGHT AT 10 LEFT OF Pagenum DISPLAY AS TEXT
VALUE '._DATE;'

FORM Es AT 4 2 SIZE 79 BY 20

CREATE FORM ES

PROMPT AT 2 2 "External Schema"

PROMPT AT 2 35 "Data Item"

PROMPT AT 2 67 "MR"

PROMPT AT 2 70 "SIZE"

PROMPT AT 2 75 "ND"

FORM Esitem (* V 0) AT 3 2 SIZE 78 BY 1

UTP620144502
1 November 1985

CREATE FORM ESITEM

ITEM Esname DISPLAY AS OUTPUT AT 1 2 SIZE 30
ITEM Diname DISPLAY AS OUTPUT AT 1 35 SIZE 30
ITEM Dimr DISPLAY AS OUTPUT AT 1 67 SIZE 1
ITEM Disize DISPLAY AS OUTPUT AT 1 70 SIZE 4
ITEM Dind DISPLAY AS OUTPUT AT 1 75 SIZE 4

UTP620144502
1 November 1985

APPENDIX B

GENERATED C AND COBOL CODE

```
#include <stdtyp.h>
#include <stdio.h>
#include <fpcode.h>
#include <ntm.h>
#include <ctlchr.h>
#define RCODE_LEN 5
#define NDMLOK "00000"
typedef struct acttyp
{
    struct acttyp *nxtact;
    int actno;
    int priority;
} ACTION;
ACTION *firstact = NULL;
ACTION **addptr;
int i, pfkey;
static int page;
int selno;
char rcode[5];
char cpath[121];
char systate[1];
bool flddsp[53];
int winfrm[53];
int scrnfrm;
char myname[] = "SDTESTAP";
char form_name[12];
#ifndef FRM46
    struct frm46
    {
        char ESNAME[30];
        char DINAME[30];
        char DIMR[1];
        char DISIZE[4];
        char DIND[4];
    } frm46; /* ESITEM */
#define FRM46
#endif
#ifndef FRM38
    struct frm38
    {
        char DUMMY[1];
        char PAGENUM[11];
    }
```

UTP620144502
1 November 1985

```
        char TODAY[9];
        } frm38; /* ESRPT */
#define FRM38
#endif
#ifndef FRM32
    struct frm32
    {
        char DBNAME[10];
        char SETID[20];
        char RTOWNID[20];
        char RTMEMID[20];
        char REQOPT[1];
    } frm32; /* STYPE */
#define FRM32
#endif
#ifndef FRM26
    struct frm26
    {
        char DUMMY[1];
        char PDATE[10];
        char PPAGE[11];
    } frm26; /* PAGE45 */
#define FRM26
#endif
#ifndef FRM17
    struct frm17
    {
        char LRECL[3];
        char BLNKS[1];
    } frm17; /* IOPTS */
#define FRM17
#endif
#ifndef FRM23
    struct frm23
    {
        char RPT45[1];
        char SELRPT[1];
    } frm23; /* TESTAP */
#define FRM23
#endif
#ifndef FRM12
    struct frm12
    {
        char OPTIONS[1];
    } frm12; /* INSFORM */
#define FRM12
#endif
```

UTP620144502
1 November 1985

```
#ifndef FRM5
    struct frm5
    {
        char VERIFY[1];
        char VERMSG[17];
        char DELPICK[10][1];
    } frm5; /* DELFORM */
#define FRM5
#endif
#ifndef FRM0
    struct frm0
    {
        char DATE[9];
        char TIME[9];
        char NEWUSER[1];
    } frm0; /* TOPFORM */
#define FRM0
#endif

    struct frm38 frm38c, frm38p;
    struct frm46 frm46c, frm46p;
    struct frm26 frm26c, frm26p;
    struct frm32 frm32c, frm32p;
    struct frm17 frm17c, frm17p;
    struct frm23 frm23c, frm23p;
    struct frm12 frm12c, frm12p;
    struct frm5 frm5c, frm5p;
    struct frm0 frm0c, frm0p;
    char path0[120] = "TOPFORM";
    char path4[120] = "TOPHELP";
    char path5[120] = "DELFORM";
    char path11[120] = "DRECORDS";
    char path12[120] = "INSFORM";
    char path16[120] = "IRECORDS";
    char path17[120] = "IOPTS";
    char path20[120] = "MODFORM";
    char path22[120] = "MRECORDS";
    char path23[120] = "TESTAP";
    char path26[120] = "PAGE45";
    char path32[120] = "STYPE";
    char path38[120] = "ESRPT";
    char path43[120] = "ES";
    char path46[120] = "ESITEM";
    struct
```

UTP620144502
1 November 1985

```
{
char DBNAME[10];
char SETID[20];
char RTOWNID[20];
char RTMEMID[20];
char REQOPT[1];
char cr;
} dbr0;
struct
{
char ESNAME[30];
char DINAME[30];
char DIMR[1];
char DISIZE[4];
char DIND[4];
char cr;
} dbr1;
FILE *selfil[2] =
{
NULL,
NULL,
};
int dbcode[2];
char filnam[2][10] =
{
"sel0.dat",
"sell.dat",
};
struct
{
char dummy;
} whrstruct;
#include <fpparm.h>
main ()
{
int page = 1;
int code;
static int currnt = CURRNT;
char *strchr();
char rcode[5];
char ntm_buffer[4096];
static char ntm_buffer_size[]={"4096"};
char systate[1];
memset(&frm38c, ' ', sizeof frm38c);
memset(&frm46c, ' ', sizeof frm46c);
memset(&frm26c, ' ', sizeof frm26c);
memset(&frm32c, ' ', sizeof frm32c);
```

UTP620144502
1 November 1985

```
memset(&frm17c, ' ', sizeof frm17c);
memset(&frm23c, ' ', sizeof frm23c);
memset(&frm12c, ' ', sizeof frm12c);
memset(&frm5c, ' ', sizeof frm5c);
memset(&frm0c, ' ', sizeof frm0c);
for (i = 0; i < 52; i++)
{
    flddsp[i] = FALSE;
    winfrm[i] = -1;
}
scrnfrm = -1;
inital(ntm_buffer, ntm_buffer_size, systate, rcode);
if (memcmp(rcode, NTMGODRET, RCODE_LEN) != 0)
{
    pmsglc(rcode);
    newpag("screen:", rcode);
    goto error;
}
initfp();
addal(0);
while (firstact)
{
    i = firstact->actno;
    free(firstact);
    firstact = firstact->nxtact;
    doact(i);
}
while (TRUE)
{
    if (flddsp[38])
        pdata(path38, &frm38c, rcode);
    if (flddsp[26])
        pdata(path26, &frm26c, rcode);
    if (flddsp[17])
        pdata(path17, &frm17c, rcode);
    if (flddsp[23])
        pdata(path23, &frm23c, rcode);
    if (flddsp[12])
        pdata(path12, &frm12c, rcode);
    if (flddsp[5])
        pdata(path5, &frm5c, rcode);
    if (flddsp[0])
        pdata(path0, &frm0c, rcode);
    oisr("screen", &pfkey, rcode);
    if (flddsp[38])
```

```
{
  memcpy(&frm38p, &frm38c, sizeof(frm38c));
  gdata(&currnt, path38, &frm38c, rcode);
}
if (flddsp[26])
{
  memcpy(&frm26p, &frm26c, sizeof(frm26c));
  gdata(&currnt, path26, &frm26c, rcode);
}
if (flddsp[17])
{
  memcpy(&frm17p, &frm17c, sizeof(frm17c));
  gdata(&currnt, path17, &frm17c, rcode);
}
if (flddsp[23])
{
  memcpy(&frm23p, &frm23c, sizeof(frm23c));
  gdata(&currnt, path23, &frm23c, rcode);
}
if (flddsp[12])
{
  memcpy(&frm12p, &frm12c, sizeof(frm12c));
  gdata(&currnt, path12, &frm12c, rcode);
}
if (flddsp[5])
{
  memcpy(&frm5p, &frm5c, sizeof(frm5c));
  gdata(&currnt, path5, &frm5c, rcode);
}
if (flddsp[0])
{
  memcpy(&frm0p, &frm0c, sizeof(frm0c));
  gdata(&currnt, path0, &frm0c, rcode);
}
trig1() ||
trig2() ||
trig3() ||
trig4() ||
trig5() ||
trig6() ||
trig7() ||
trig8() ||
trig9() ||
trig10() ||
trig11() ||
trig12() ||
trig13() ||
```

```
    trig14() ||
    trig16() ||
    trig17() ||
    trig18() ||
    trig20() ||
    trig21() ||
    chkcng();

    while (firstact)
    {
        i = firstact->actno;
        free(firstact);
        firstact = firstact->nxtact;
        doact(i);
    }
}
error:
termfp();
systate[0] = '1';
trmndml(systate);
}
bool trig1()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
    *strchr();
    char tmpnam[128];
    int path_len, chk_len, row, col, i, j;
    if (pfkey != 4 || !flddsp[5])
        return FALSE;
    getcur(path_name, &fld_type, &row, &col, rcode);
    path_len = strchr(path_name, ':') - path_name + 1;
    j = 0;
    sprintf(tmpnam, "DELFORM");
    chk_len = strlen(tmpnam);
    while ((i = index(&path_name[j], tmpnam)) != 0 &&
        i + j + chk_len <= path_len)
    {
        if (path_name[i + j - 1] == '.' &&
            strchr(".(;", path_name[i + j + chk_len]))
        {
            cond = TRUE;
            page = 1;
            rmvpag("screen;", &page, rcode);
            addfrm("screen", "TOPFORM", &page, rcode);
            strcpy(form_name, "TOPFORM");
            pdata(path0, &frm0c, rcode);
        }
    }
}
```

```

        flddsp[0] = TRUE;
        if (scrnfrm == 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 0;
    }
    j += i + 1;
}
return cond;
}
bool trig2()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
    *strchr();
    char tmpnam[128];
    int path_len, chck_len, row, col, i, j;
    if (pfkey == 4 || !flddsp[12])
        return FALSE;
    getcur(path_name, &fld_type, &row, &col, rcode);
    path_len = strchr(path_name, ';') - path_name + 1;
    j = 0;
    sprintf(tmpnam, "INSFORM");
    chck_len = strlen(tmpnam);
    while ((i = index(&path_name[j], tmpnam)) != 0 &&
        i + j + chck_len == path_len)
    {
        if (path_name[i + j - 1] == '.' &&
            strchr(".(;", path_name[i + j + chck_len]))
        {
            cond = TRUE;
            page = 1;
            rmvpag("screen:", &page, rcode);
            addfrm("screen", "TOPFORM", &page, rcode);
            strcpy(form_name, "TOPFORM");
            pdata(path0, &frm0c, rcode);
            flddsp[0] = TRUE;
            if (scrnfrm == 0) flddsp[scrnfrm] = FALSE;
            scrnfrm = 0;
        }
        j += i + 1;
    }
    return cond;
}
bool trig3()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
    *strchr();

```

```
char tmpnam[128];
int path_len, chck_len, row, col, i, j;
if (pfkey != 4 || !flddsp[20])
    return FALSE;
getcur(path_name, &fld_type, &row, &col, rcode);
path_len = strchr(path_name, ';') - path_name + 1;
j = 0;
sprintf(tmpnam, "MODFORM");
chck_len = strlen(tmpnam);
while ((i = index(&path_name[j], tmpnam)) != 0 &&
        i + j + chck_len <= path_len)
{
    if (path_name[i + j - 1] == '.' &&
        strchr(".(;<", path_name[i + j + chck_len]))
    {
        cond = TRUE;
        page = 1;
        rmvpag("screen;", &page, rcode);
        addfrm("screen", "TOPFORM", &page, rcode);
        strcpy(form_name, "TOPFORM");
        pdata(path0, &frm0c, rcode);
        flddsp[0] = TRUE;
        if (scrnfrm != 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 0;
    }
    j += i + 1;
}
return cond;
}
bool trig4()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
        *strchr();
    char tmpnam[128];
    int path_len, chck_len, row, col, i, j;
    if (pfkey != 4 || !flddsp[23])
        return FALSE;
    getcur(path_name, &fld_type, &row, &col, rcode);
    path_len = strchr(path_name, ';') - path_name + 1;
    j = 0;
    sprintf(tmpnam, "TESTAP");
    chck_len = strlen(tmpnam);
    while ((i = index(&path_name[j], tmpnam)) != 0 &&
            i + j + chck_len <= path_len)
```

```
{
  if (path_name[i + j - 1] == '.' &&
      strchr("(<;", path_name[i + j + chck_len]))
  {
    cond = TRUE;
    page = 1;
    rmvpag("screen;", &page, rcode);
    addfrm("screen", "TOPFORM", &page, rcode);
    strcpy(form_name, "TOPFORM");
    pdata(path0, &frm0c, rcode);
    flddsp[0] = TRUE;
    if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
    scrnfrm = 0;
  }
  j += i + 1;
}
return cond;
}
bool trig5()
{
  bool cond = FALSE;
  if (pfkey != 5 || !flddsp[0])
    return FALSE;
  page = 1;
  rmvpag("screen;", &page, rcode);
  addfrm("screen", "DELFORM", &page, rcode);
  strcpy(form_name, "DELFORM");
  pdata(path5, &frm5c, rcode);
  flddsp[5] = TRUE;
  if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
  scrnfrm = 5;
  return TRUE;
}
bool trig6()
{
  bool cond = FALSE;
  if (pfkey != 6 || !flddsp[0])
    return FALSE;
  page = 1;
  rmvpag("screen;", &page, rcode);
  addfrm("screen", "INSFORM", &page, rcode);
  strcpy(form_name, "INSFORM");
  pdata(path12, &frm12c, rcode);
  flddsp[12] = TRUE;
  if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
  scrnfrm = 12;
  return TRUE;
}
```

```
    }  
    bool trig7()  
    {  
        bool cond = FALSE;  
        if (pfkey != 7 || !fiddsp[0])  
            return FALSE;  
        page = 1;  
        rmvpag("screen;", @page, rcode);  
        addfrm("screen", "MODFORM", @page, rcode);  
        strcpy(form_name, "MODFORM");  
        fiddsp[20] = TRUE;  
        if (scrnfrm >= 0) fiddsp[scrnfrm] = FALSE;  
        scrnfrm = 20;  
        return TRUE;  
    }  
    bool trig8()  
    {  
        bool cond = FALSE;  
        if (pfkey != 8 || !fiddsp[0])  
            return FALSE;  
        page = 1;  
        rmvpag("screen;", @page, rcode);  
        addfrm("screen", "TESTAP", @page, rcode);  
        strcpy(form_name, "TESTAP");  
        pdata(path23, @frm23c, rcode);  
        fiddsp[23] = TRUE;  
        if (scrnfrm >= 0) fiddsp[scrnfrm] = FALSE;  
        scrnfrm = 23;  
        return TRUE;  
    }  
    bool trig9()  
    {  
        bool cond = FALSE;  
        char tempstr[128];  
        if (!fiddsp[5]) return FALSE;  
        if (memcmp(frm5c.VERIFY,  
            "Y",  
            sizeof(frm5c.VERIFY)) == 0)  
        {  
            cond = TRUE;  
            memcpy(frm5c.VERIFY, " ", sizeof frm5c.VERIFY);  
            pdata(path5, @frm5c, rcode);  
            memcpy(frm5c.VERMSG, "Verification On ", sizeof  
                frm5c.VERMSG);  
            pdata(path5, @frm5c, rcode);  
            page = 1;  
            rmvpag("screen;", @page, rcode);  
        }  
    }
```

UTP620144502
1 November 1985

```
        addfrm("screen", "DELFORM", &page, rcode);
        strcpy(form_name, "DELFORM");
        pdata(path5, &frm5c, rcode);
        flddsp[5] = TRUE;
        if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 5;
    }
    return cond;
}
bool trig10()
{
    bool cond = FALSE;
    char tempstr[128];
    if (!flddsp[5]) return FALSE;
    if (memcmp(frm5c.VERIFY,
               "N",
               sizeof(frm5c.VERIFY)) == 0)
    {
        cond = TRUE;
        memcpy(frm5c.VERIFY, " ", sizeof frm5c.VERIFY);
        pdata(path5, &frm5c, rcode);
        memcpy(frm5c.VERMSG, "Verification Off ", sizeof
               frm5c.VERMSG);
        pdata(path5, &frm5c, rcode);
        page = 1;
        rmvpag("screen;", &page, rcode);
        addfrm("screen", "DELFORM", &page, rcode);
        strcpy(form_name, "DELFORM");
        pdata(path5, &frm5c, rcode);
        flddsp[5] = TRUE;
        if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 5;
    }
    return cond;
}
bool trig11()
{
    bool cond = FALSE;
    char tempstr[128];
    if (!flddsp[12]) return FALSE;
    if (memcmp(frml2c.OPTIONS,
               "Y",
               sizeof(frml2c.OPTIONS)) == 0)
    {
        cond = TRUE;
        page = 1;
        rmvpag("INSWIN;", &page, rcode);
```

```
    addfrm("INSWIN;", "IOPTS", &page, rcode);
    pdata(path17, &frm17c, rcode);
    flddsp[17] = TRUE;
    if (winfrm[16] >= 0) flddsp[winfrm[16]] = FALSE;
    winfrm[16] = 17;
}
return cond;
}
bool  trig12()
{
    bool cond = FALSE;
    char  tempstr[128];
    if (!flddsp[0]) return FALSE;
    if (memcmp(frm0c.NEWUSER,
        "Y",
        sizeof(frm0c.NEWUSER)) == 0)
    {
        cond = TRUE;
        memcpy(frm0c.NEWUSER, " ", sizeof frm0c.NEWUSER);
        pdata(path0, &frm0c, rcode);
        addfrm("screen", "TOPHELP", &page, rcode);
        pmsgls("Press <enter> to continue.");
        oisclr("screen", &pfkey, rcode);
        rmvpag("screen", &page, rcode);
    }
    return cond;
}
bool  trig13()
{
    bool cond = FALSE;
    if (pfkey != 4 || !flddsp[0])
        return FALSE;
    termfp();
    systate[0] = '1';
    trmndml(systate);
    return TRUE;
}
bool  trig14()
{
    bool cond = FALSE;
    char  tempstr[128];
    if (!flddsp[23]) return FALSE;
    if (memcmp(frm23c.RPT45,
        " ",
        sizeof(frm23c.RPT45)) != 0)
```

UTP620144502
1 November 1985

```
{
cond = TRUE;
selno = 0;
TESTAPX(&selno, &whrstruct,
        filnam[0],
        filnam[1],
        rcode);
if (memcmp(rcode, NDMLOK, RCODE_LEN) != 0)
pmsgls(rcode);
if (selfil[0]) fclose(selfil[0]);
selfil[0] = fopen(filnam[0], "r");
nextdb(0, &dbr0, sizeof dbr0);
STRASN(frm32p, frm32c);
page = 1;
rmvpag("screen;", &page, rcode);
addfrm("screen", "PAGE45", &page, rcode);
strcpy(form name, "PAGE45");
pdata(path26, &frm26c, rcode);
flddsp[26] = TRUE;
if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
scrnfrm = 26;
PAGE45();
}
return cond;
}
bool trig16()
{
bool cond = FALSE;
char path_name[128], rcode[RCODE_LEN], fld_type,
*strchr();
char tmpnam[128];
int path_len, chck_len, row, col, i, j;
if (pfkey != 0 || !flddsp[26])
return FALSE;
getcur(path_name, &fld_type, &row, &col, rcode);
path_len = strchr(path_name, ';') - path_name + 1;
j = 0;
sprintf(tmpnam, "PAGE45");
chck_len = strlen(tmpnam);
while ((i = index(&path_name[j], tmpnam)) >= 0 &&
        i + j + chck_len <= path_len)
{
if (path_name[i + j - 1] == '.' &&
        strchr(".(;", path_name[i + j + chck_len]))
```

```
{
    cond = TRUE;
    page = 1;
    rmvpag("screen;", @page, rcode);
    addfrm("screen", "PAGE45", @page, rcode);
    strcpy(form_name, "PAGE45");
    pdata(path26, @frm26c, rcode);
    flddsp[26] = TRUE;
    if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
    scrnfrm = 26;
    PAGE45();
}
j += i + 1;
}
return cond;
}
bool trig17()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
    *strchr();
    char tmpnam[128];
    int path_len, chk_len, row, col, i, j;
    if (pfkey != 4 || !flddsp[26])
        return FALSE;
    getcur(path_name, @fld_type, @row, @col, rcode);
    path_len = strchr(path_name, ';') - path_name + 1;
    j = 0;
    sprintf(tmpnam, "PAGE45");
    chk_len = strlen(tmpnam);
    while ((i = index(@path_name[j], tmpnam)) != 0 &&
        i + j + chk_len <= path_len)
    {
        if (path_name[i + j - 1] == '.' &&
            strchr(".(;<", path_name[i + j + chk_len]))
        {
            cond = TRUE;
            page = 1;
            rmvpag("screen;", @page, rcode);
            addfrm("screen", "TESTAP", @page, rcode);
            strcpy(form_name, "TESTAP");
            pdata(path23, @frm23c, rcode);
            flddsp[23] = TRUE;
            if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
            scrnfrm = 23;
        }
    }
}
```

```
        j += i + 1;
    }
    return cond;
}
bool trig18()
{
    bool cond = FALSE;
    char tempstr[128];
    if (!flddsp[23]) return FALSE;
    if (memcmp(frm23c.SELRPT,
        " ",
        sizeof(frm23c.SELRPT)) != 0)
    {
        cond = TRUE;
        selno = 1;
        TESTAPX(&selno, &whrstruct,
            filnam[0],
            filnam[1],
            rcode);
        if (memcmp(rcode, NDMLOK, RCODE_LEN) != 0)
            pmsgls(rcode);
        if (selfil[1]) fclose(selfil[1]);
        selfil[1] = fopen(filnam[1], "r");
        nextdb(1, &dbrl, sizeof dbrl);
        STRASN(frm46p, frm46c);
        page = 1;
        rmvpag("screen;", &page, rcode);
        addfrm("screen", "ESRPT", &page, rcode);
        strcpy(form_name, "ESRPT");
        pdata(path38, &frm38c, rcode);
        flddsp[38] = TRUE;
        if (scrnfrm != 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 38;
        ESRPT();
    }
    return cond;
}
bool trig20()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
        *strchr();
    char tmpnam[128];
    int path_len, chck_len, row, col, i, j;
    if (pfkey != 0 || !flddsp[38])
        return FALSE;
    getcur(path_name, &fld_type, &row, &col, rcode);
```

```

path_len = strchr(path_name, ';') - path_name + 1;
j = 0;
sprintf(tmpnam, "ESRPT");
chk_len = strlen(tmpnam);
while ((i = index(&path_name[j], tmpnam)) != 0 &&
        i + j + chk_len <= path_len)
{
    if (path_name[i + j - 1] == '.' &&
        strchr(".(;<", path_name[i + j + chk_len]))
    {
        cond = TRUE;
        page = 1;
        rmvpag("screen:", &page, rcode);
        addfrm("screen", "ESRPT", &page, rcode);
        strcpy(form_name, "ESRPT");
        pdata(path38, &frm38c, rcode);
        flddsp[38] = TRUE;
        if (scrnfrm != 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 38;
        ESRPT();
    }
    j += i + 1;
}
return cond;
}
bool trig21()
{
    bool cond = FALSE;
    char path_name[128], rcode[RCODE_LEN], fld_type,
    *strchr();
    char tmpnam[128];
    int path_len, chk_len, row, col, i, j;
    if (pfkey != 4 || !flddsp[38])
        return FALSE;
    getcur(path_name, &fld_type, &row, &col, rcode);
    path_len = strchr(path_name, ';') - path_name + 1;
    j = 0;
    sprintf(tmpnam, "ESRPT");
    chk_len = strlen(tmpnam);
    while ((i = index(&path_name[j], tmpnam)) != 0 &&
            i + j + chk_len <= path_len)
    {
        if (path_name[i + j - 1] == '.' &&
            strchr(".(;<", path_name[i + j + chk_len]))
        {

```

```
    {
        cond = TRUE;
        page = 1;
        rmvpag("screen;", @page, rcode);
        addfrm("screen", "TESTAP", @page, rcode);
        strcpy(form_name, "TESTAP");
        pdata(path23, @frm23c, rcode);
        flddsp[23] = TRUE;
        if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
        scrnfrm = 23;
    }
    j += i + 1;
}
return cond;
}
ESRPT () /* form number 38 */
{
    char rcode[5];
    char cpath[120];
    int element;
    int donel;
    donel = dbcode[1];
    while (!donel)
    {
        if (!donel)
        {
            sprintf(cpath, "%s.ESITEM", path38);
            addelm(cpath, @element, rcode);
            if (memcmp(rcode, OVRFLW, 5) == 0)
            {
                addal(19);
                donel = 2;
            }
        }
        else
        {
            sprintf(cpath, "%s(%d)", cpath, element);
            pdata(cpath, @frm46c, rcode);
            donel |= nextdb(1, @dbrl, sizeof dbrl);
            donel |= chkeng();
        }
    }
    if (dbcode[1] == 2) donel = nextdb(1, @dbrl, sizeof dbrl);
    return donel;
}
PAGE45 () /* form number 26 */
```

UTP620144502
1 November 1985

```
{
char rcode[5];
char cpath[120];
int element;
int donel;
donel = dbcode[0];
while (!donel)
{
    if (!donel)
    {
        sprintf(cpath, "%s.STYPE", path26);
        addelm(cpath, &element, rcode);
        if (memcmp(rcode, OVRFLW, 5) == 0)
        {
            addal(15);
            donel = 2;
        }
        else
        {
            sprintf(cpath, "%s(%d)", cpath, element);
            pdata(cpath, &frm32c, rcode);
            donel |= nextdb(0, &dbr0, sizeof dbr0);
            donel |= chkeng();
        }
    }
    if (dbcode[0] == 2) donel = nextdb(0, &dbr0, sizeof dbr0);
    return donel;
}

nextdb(selno, string, length)
int selno;
char string[];
int length;
{
    if (fread(string, length, 1, selfil[selno]))
    {
        if (string[0] == GS_CH)
        {
            dbcode[selno] = 2;
            switch (selno)
            {
                case 0 :
                {
                    break;
                }
                case 1 :
            }
        }
    }
}
```

```
        {  
        }  
        break;  
    }  
    }  
    else  
    {  
        dbcode[selno] = 0;  
        switch (selno)  
        {  
            case 0 :  
            {  
                STRASN(frm32p, frm32c);  
                memcpy(frm32c.DBNAME, dbr0.DBNAME, 10);  
                memcpy(frm32c.SETID, dbr0.SETID, 20);  
                memcpy(frm32c.RTOWNID, dbr0.RTOWNID, 20);  
                memcpy(frm32c.RTMEMID, dbr0.RTMEMID, 20);  
                memcpy(frm32c.REQOPT, dbr0.REQOPT, 1);  
            }  
            break;  
            case 1 :  
            {  
                STRASN(frm46p, frm46c);  
                memcpy(frm46c.ESNAME, dbr1.ESNAME, 30);  
                memcpy(frm46c.DINAME, dbr1.DINAME, 30);  
                memcpy(frm46c.DIMR, dbr1.DIMR, 1);  
                memcpy(frm46c.DISIZE, dbr1.DISIZE, 4);  
                memcpy(frm46c.DIND, dbr1.DIND, 4);  
            }  
            break;  
        }  
    }  
    }  
    else dbcode[selno] = 1;  
    return dbcode[selno];  
}  
chkcng()  
{  
    int code;  
    code = 0;  
    return code;  
}  
addal(trgno)  
int trgno;  
{  
    addptr = &firstact;  
    switch (trgno)
```

UTP620144502
1 November 1985

```
{
case -2: addact(-2, 1); break;
case 0 :
    {
        addact(0, 3);
        addact(1, 2);
    }
break;
case 1 :
    {
        addact(2, 2);
    }
break;
case 2 :
    {
        addact(3, 2);
    }
break;
case 3 :
    {
        addact(4, 2);
    }
break;
case 4 :
    {
        addact(5, 2);
    }
break;
case 5 :
    {
        addact(6, 2);
    }
break;
case 6 :
    {
        addact(7, 2);
    }
break;
case 7 :
    {
        addact(8, 2);
    }
break;
case 8 :
    {
        addact(9, 2);
    }
}
```

UTP620144502
1 November 1985

```
break;
case 9 :
{
    addact(10, 3);
    addact(11, 3);
    addact(12, 2);
}
break;
case 10 :
{
    addact(13, 3);
    addact(14, 3);
    addact(15, 2);
}
break;
case 11 :
{
    addact(16, 2);
}
break;
case 12 :
{
    addact(17, 3);
    addact(18, 6);
}
break;
case 13 :
{
    addact(19, 0);
}
break;
case 14 :
{
    addact(20, 1);
    addact(21, 2);
}
break;
case 15 :
{
    addact(22, 6);
}
break;
case 16 :
{
    addact(23, 2);
}
```

UTP620144502
1 November 1985

```
break;
case 17 :
{
    addact(24, 2);
}
break;
case 18 :
{
    addact(25, 1);
    addact(26, 2);
}
break;
case 19 :
{
    addact(27, 6);
}
break;
case 20 :
{
    addact(28, 2);
}
break;
case 21 :
{
    addact(29, 2);
}
break;
}
}
addact(actno, priority)
int actno;
int priority;
{
    ACTION *ap;
    for (, *addptr @@ (*addptr)->priority = priority;
        addptr = @(*addptr)->nxtact);
    for (, *addptr @@ (*addptr)->priority == priority;
        addptr = @(*addptr)->nxtact);
    ap = (ACTION *)malloc(sizeof (ACTION));
    ap->actno = actno;
    ap->priority = priority;
    ap->nxtact = *addptr;
    *addptr = ap;
    addptr = @ap->nxtact;
}
```

UTP620144502
1 November 1985

```
doact(actno)
int actno;
{
    switch (actno)
    {
        case 0 :
        {
            memcpy(frm0c.NEWUSER, " ", sizeof frm0c.NEWUSER);
            pdata(path0, &frm0c, rcode);
        }
        break;
        case 1 :
        {
            page = 1;
            rmvpag("screen;", &page, rcode);
            addfrm("screen", "TOPFORM", &page, rcode);
            strcpy(form_name, "TOPFORM");
            pdata(path0, &frm0c, rcode);
            flddsp[0] = TRUE;
            if (scrnfrm >= 0) flddsp[scrnfrm] = FALSE;
            scrnfrm = 0;
        }
        break;
        case 22 :
        {
            pmsgls("Press <enter> to continue, <quit> to
quit.");
        }
        break;
        case 27 :
        {
            pmsgls("Press <enter> to continue, <quit> to
quit.");
        }
        break;
    }
}

newpag(win, rcode)
char win[], rcode[];
{
    char pagstr[4];
    static int pagcnt = 0;
    sprintf(pagstr, "%3d", ++pagcnt);
    pdata("._pageno:", pagstr, rcode);
    if (memcmp(rcode, OK, RCODE_LEN) == 0) outscr(win, rcode);
}
```

Generated COBOL Code

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TESTAP.
AUTHOR. SDRC.
INSTALLATION. SDRC.
DATE-WRITTEN. 03/01/85.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    SYMBOLIC GRPSEP IS 30.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT FCVO ASSIGN "FVARO".
DATA DIVISION.
FILE SECTION.
    FD FCVO
    ACCESS IS SEQUENTIAL
    VALUE OF ID IS FVARO.
01 FRECO.
    03 FILLER PIC X(71).
WORKING-STORAGE SECTION.
01 FVARO PIC X(10).
01 PSREC-0.
    03 DBNAME PIC X(10).
    03 SETID PIC X(20).
    03 RTOWNID PIC X(20).
    03 RTMEMID PIC X(20).
    03 REQOPT PIC X(1).
01 EDATA-BASE-0.
    02 DB-ID-0-0 PIC 9(6).
    02 DBMS-NAME-0-0 PIC X(030).
    02 DB-NAME-0-0 PIC X(030).
    02 HOST-ID-0-0 PIC X(030).
01 CDATA-BASE-0.
    02 CDB-ID PIC -----.
    02 CDBMS-NAME PIC X(030).
    02 CDB-NAME PIC X(030).
    02 CHOST-ID PIC X(030).
01 ERECORD-TYPE-0.
    02 DB-ID-0-1 PIC 9(6).
    02 RT-ID-0-1 PIC X(030).
    02 RT-NO-0-1 PIC 9(6).
```

```
01 CRECORD-TYPE-0.
  02 CDB-ID PIC -----
  02 CRT-ID PIC X(030).
  02 CRT-NO PIC -----
01 ERECORD-SET-0.
  02 DB-ID-0-2 PIC 9(6) .
  02 TOTAL-NUM-MEM-0-2 PIC 9(6) .
  02 RT-ID-OF-OWNER-0-2 PIC X(030).
  02 SET-ID-0-2 PIC X(030).
  02 SET-NO-0-2 PIC 9(6) .
01 CRECORD-SET-0
  02 CDB-ID PIC -----
  02 CTOTAL-NUM-MEM PIC -----
  02 CRT-ID-OF-OWNER PIC X(030).
  02 CSET-ID PIC X(030).
  02 CSET-NO PIC -----
01 ESET-TYPE-MEMBER-0.
  02 DB-ID-0-3 PIC 9(6) .
  02 RT-ID-OF-MEMBER-0-3 PIC X(030).
  02 DB-ID-OF-MEMBER-0-3 PIC 9(6) .
  02 SET-ID-0-3 PIC X(030).
  02 REQ-MEM-IND-0-3 PIC X(001).
01 CSET-TYPE-MEMBER-0.
  02 CDB-ID PIC -----
  02 CRT-ID-OF-MEMBER PIC X(030).
  02 CDB-ID-OF-MEMBER PIC -----
  02 CSET-ID PIC X(030).
  02 CREQ-MEM-IND PIC X(001).
COPY SRVRET OF IISSCLIB
01 MODULE-NAME PIC X(10) VALUE "TESTAP"
01 MSG-DESC PIC X(60)
01 RET-STATUS PIC X(5)
01 REALTEMP COMP-2
01 RC PIC 9999 COMP
LINKAGE SECTION
01 SELECT-NUMBER PIC 9(5) COMP
01 WHERE-STRUCT
  03 DUMMY PIC X(1)
01 PFO PIC X(10)
PROCEDURE DIVISION USING
  SELECT-NUMBER
  WHERE STRUCT
```

PFO

```
START-PROGRAM.
  COMPUTE SELECT-NUMBER = SELECT-NUMBER + 1.
  GO TO
  SELECTO-BEG
  DEPENDING ON SELECT-NUMBER.
MAPO-BEG.
  MOVE DB-NAME-0-0 OF EDATA-BASE-0
    TO DBNAME OF PSREC-0.
  MOVE SET-ID-0-3 OF ESET-TYPE-MEMBER-0
    TO SETID OF PSREC-0.
  MOVE RT-ID-OF-OWNER-0-2 OF ERECORD-SET-0
    TO RTOWNID OF PSREC-0.
  MOVE RT-ID-OF-MEMBER-0-3 OF ESET-TYPE-MEMBER-0
    TO RTMEMID OF PSREC-0.
  MOVE REQ-MEM-IND-0-3 OF ESET-TYPE-MEMBER-0
    TO REQOPT OF PSREC-0.
MAPO-END.
CLOSEO-BEG.
  CLOSE FCVO.
CLOSEO-END.
SELECTO-BEG.
  MOVE PFO TO FVARO.
  OPEN OUTPUT FCVO.
  ** SELECT
  **   :DB-NAME-0-0
  **     = DB.DB_NAME,
  **   :SET-ID-0-3
  **     = SM.SET_ID,
  **   :RT-ID-OF-OWNER-0-2
  **     = RS.RT_ID_OF_OWNER,
  **   :RT-ID-OF-MEMBER-0-3
  **     = SM.RT_ID_OF_MEMBER,
  **   :REQ-MEM-IND-0-3
  **     = SM.REQ_MEM_IND
  ** FROM
  **   DATA_BASE DB,
  **   RECORD_TYPE RT,
  **   RECORD_SET RS,
  **   SET_TYPE_MEMBER SM
  ** WHERE
  **   DB.DB_ID
  **     = RT.DB_ID AND
```

UTP620144502
1 November 1985

```
** RT.DB_ID
**      =      RS.DB_ID AND
** RT.RT_ID
**      =      RS.RT_ID_OF_OWNER AND
** RS.DB_ID
**      =      SM.DB_ID AND
** RS.SET_ID
**      =      SM.SET_ID
**      {
**          PERFORM MAPO-BEG THRU MAPO-END.
**          WRITE FRECO FROM PSREC-O.
**      }
**      IF NOT OK
**          PERFORM CLOSEO-BEG THRU CLOSEO-END
**          GO TO NDML-ERROR
**      END-IF
**      PERFORM CLOSEO-BEG THRU CLOSEO-END.
**      EXIT-PROGRAM.
**      EXIT PROGRAM.
* NDML ERROR PROCESSING
NDML-ERROR.
    MOVE NDML-STATUS TO RET_STATUS.
    MOVE 'NDML-ERROR TRAPPED' TO MESG-DESC.
    PERFORM PROCESS-ERROR.
    GO TO EXIT-PROGRAM.
*
* PROCESS ERROR
COPY ERRPRO OF IISSCLIB.
```

UTP620144502
1 November 1985

APPENDIX C

GENERATED BINARY FORM FILES

TOPFORM.FD
DELFORF.FD
INSFORM.FD
IOPTS.FD
MRECORDS.FD
PAGE45.FD
ESRPT.FD
ESITEM.FD

TOPHELP.FD
DRECORDS.FD
IRECORDS.FD
MODFORM.FD
TESTAP.FD
STYPE.FD
ES.FD

UTP620144502
1 November 1985

APPENDIX D
OUTPUT SCREENS

Structural Dynamics Research Corporation
IIS Application Generator Release 2.0

DATE: 5/30/85 TIME: 16:52:31

Do you wish help getting started? (Y/N) ☐

Msg: ☐ U application

Figure D-1 Top Level Form

This example illustrates the functionality of the Rapid Application Generator. Only query capability is available in this release.

- . Hit pf4 to return to the top level form
- . Hit pf5 to display a DELETE form
- . Hit pf6 to display an INSERT form
- . Hit pf7 to display a MODIFY form
- . Hit pf8 to display a QUERY form

Hit (enter) to return to the top level form.

Msg: application

Figure D-2 Help Form
Obtained by Answering Y in Top Level Form

UTP620144502
1 November 1985

Application Generator

Delete Record Form

Do you wish to verify before DELETE operation is committed? (Y/N) ☐

Records Retrieved

--	--

Msg: ☐ application

Figure D-3 Delete Form
Obtained by Selection of Delete Key on Top Level Form

UTP620144502
1 November 1985

Application Generator

Insert Record Form

Records To Be Inserted.

Options: (Y/N)

Page:

application

Figure D-4 Insert Form
Obtained by Selection of Insert Key on Top Level Form

UTP620144502
1 November 1985

The image shows a terminal window titled 'Application Generator'. Inside the window, the text 'Modify Record Form' is displayed at the top, followed by a horizontal line. Below the line, the text 'Records To Be Modified' is centered. A large, empty rectangular box occupies the middle section of the form. At the bottom left, there is a label 'Msg:' followed by a small square icon. At the bottom right, the word 'Application' is visible.

Figure D-5 Modify Form
Obtained by Selection of Modify Key on Top Level Form

UTP620144502
1 November 1985

Application Generator

Select Record Form

Available Reports

RPT45
SELRPT

Enter any non-blank character to select a report

Page: 0

Application Generator

Figure D-6 Select Form
Obtained by Selection of Query Key on Top Level Form

UTP620144502
1 November 1985

Application Generator

Delete Record Form

Do you wish to verify before DELETE operation is committed? (Y/N) ☐

Verification On

Records Retrieved

--	--

Msg: ☐ 0

application

Figure D-7
Result of Entry of Y in Verify Item of Delete Form

UTP620144502
1 November 1985

Application Generator

Delete Record Form

Do you wish to verify before DELETE operation is committed? (Y/N) ☐

Verification Off

Records Retrieved

--	--

Msg: ☐ 0 application

Figure D-8
Result of Entry of N in Verify Item of Delete Form

UTP620144502
1 November 1985

Application Generator

Insert Record Form

Records To Be Inserted.

Options: (Y/N) ☒ Y

Record Length: Strip trailing blanks? (Y/N) ☐

Reg: 0 application

Figure D-9
Result of Entry of Y in Options Item of Insert Form

UTP620144502
1 November 1985

14:39:09 REPORT OF CODASYL SET TYPES

☐ DB_NAME SET_ID RT_ID_OF_OWNER RT_ID_OF_MEMBER R/O

P108	USNP01-PTTIB-PLT	USNP01-PART-PTTR	USNP01-PLANT-LEC	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R
P108	USNP01-PLTLEC-SN	USNP01-PLANT-LEC	USNP01-IMANT-SIN	R

Msg: ☐ 1 Press (enter) to continue. (quit) to quit. application

Figure D-10
Result of Entry of X in Rpt45 Item of Testap Form

END

8-87

DTIC