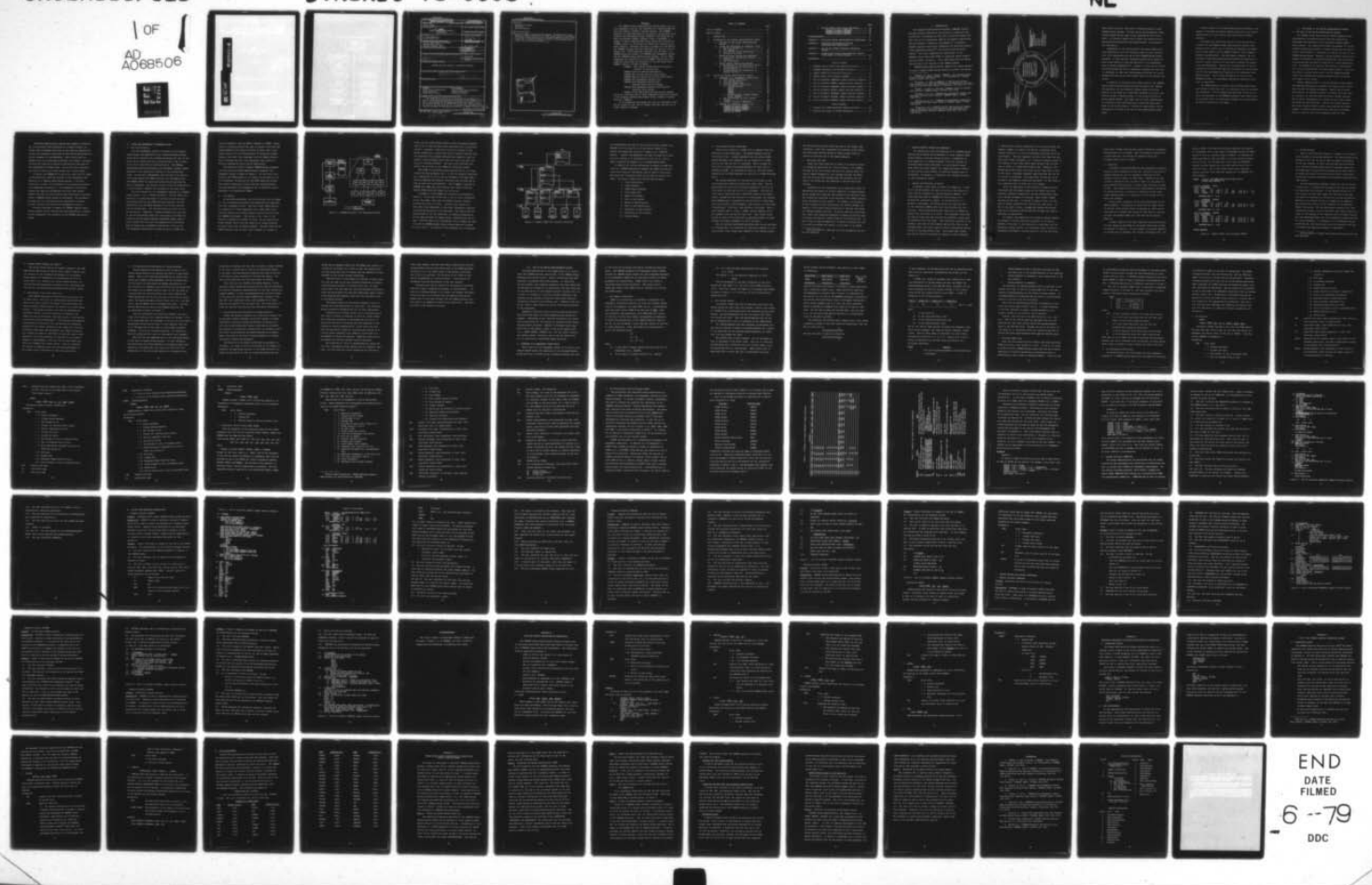
AD-A068 506 DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 13/10
COMRADE DESIGN ADMINISTRATION SYSTEM. USERS MANUAL,(U)
APR 79 C M CHERNICK

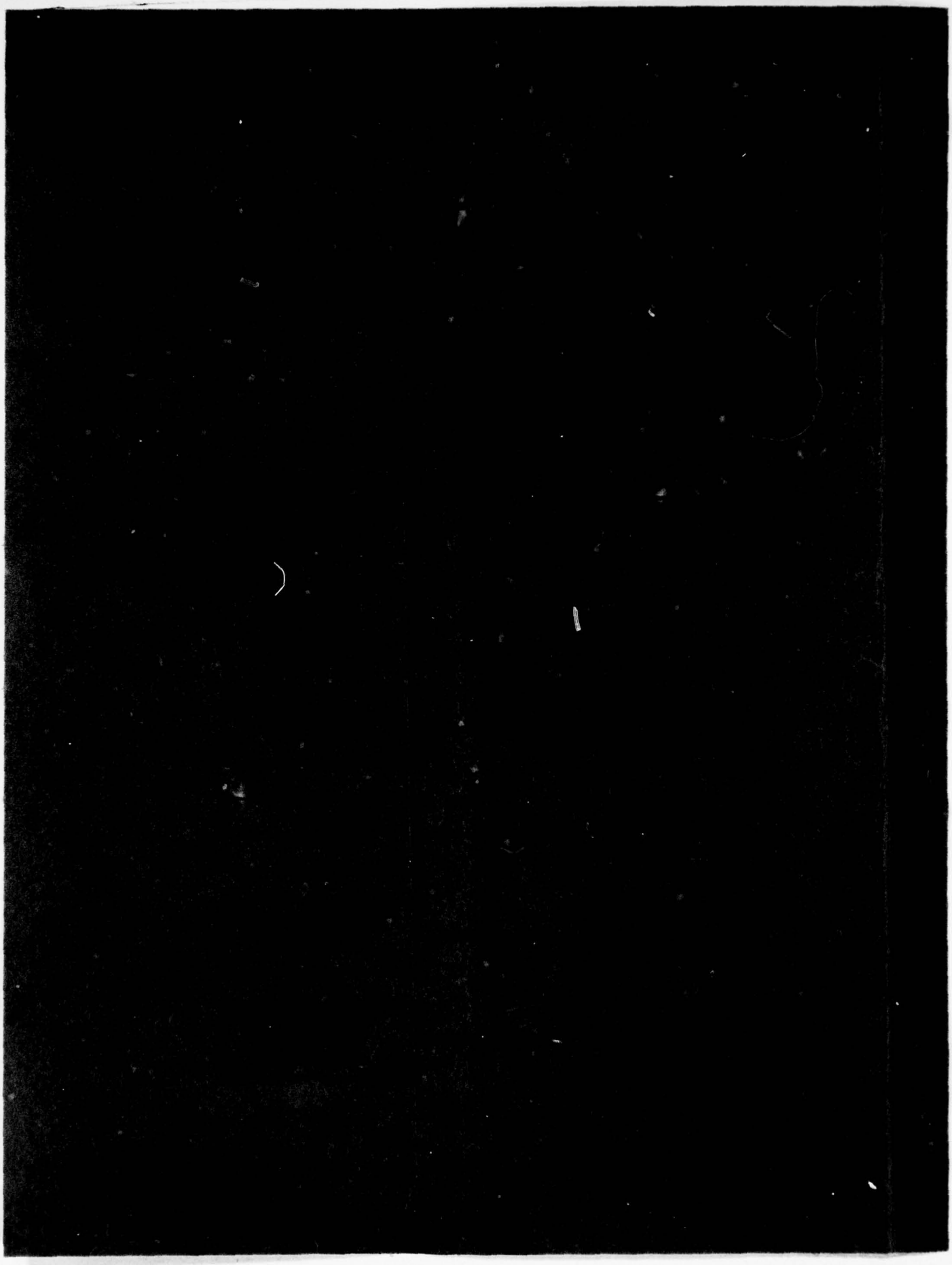
UNCLASSIFIED DTNSRDC-76-0008 NL



END
DATE FILMED
6 -79
DDC

DDC FILE COPY

AD A068506



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: Report 76-0008	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): (6) COMRADE DESIGN ADMINISTRATION SYSTEM USERS MANUAL	5. TYPE OF REPORT & PERIOD COVERED / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): (10) C. Michael Chernick	8. CONTRACT OR GRANT NUMBER(s): (12) 95p.
9. PERFORMING ORGANIZATION NAME AND ADDRESS: David W. Taylor Naval Ship Research and Development Center, Bethesda, Maryland 20084	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (See reverse side)
11. CONTROLLING OFFICE NAME AND ADDRESS	(11) 12. REPORT DATE: April 1979 / 13. NUMBER OF PAGES: 94
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (14) DTNSRDC-76-0008	15. SECURITY CLASS. (of this report): UNCLASSIFIED / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(16) SO331H (17) 14525

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

COMRADE
Design Project Administration
Design Monitoring
Design Project Control
File Management
Computer-Aided Project Management
Computer-Aided Design
File Access Control

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

The COMRADE Design Administration System (DA) was developed at the David W. Taylor Naval Ship Research and Development Center (DTNSRDC) in support of the Computer-Aided Ship Design and Construction (CASDAC) Project. The DA system operates on the DTNSRDC Control Data 6700 computing facility and serves as the focal point for all the organizational and administrative functions necessary for development and operation of COMRADE (Computer-Aided

(Continued on reverse side)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

387 682

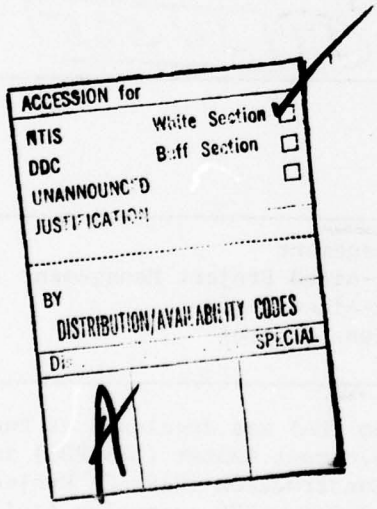
JOB

(Block 10)

63509N Project S0331H
Task 14525
Work Unit 1850-030

(Block 20 continued)

Design Environment) integrated design systems. DA enables design system managers to control system usage, to report design progress, to initialize and terminate design projects, and to efficiently manage and protect design system files. This report provides an overview of the DA system and describes the use of the DA capabilities.



FOREWORD

The COMRADE Design Administration System serves as a project management tool for the comprehensive Computer-Aided Design Environment (COMRADE) software system. The COMRADE software has been developed at the David W. Taylor Naval Ship Research and Development Center (DTNSRDC) and funded by the Naval Sea Systems Command (NAVSEA, formerly NAVSHIPS) in support of their Computer-Aided Ship Design and Construction (CASDAC) effort to facilitate the design and construction of Naval vessels. Although conceived in support of ship design, the COMRADE software has been developed for use as a general design tool. Three separate components are included: the COMRADE Executive System; the COMRADE Data Management System; and the COMRADE Design Administration System. All of these components are operational on the DTNSRDC Control Data 6700 computing system and are documented in the following eight DTNSRDC reports (76-0001--76-0008):

COMRADE - The Computer-Aided Design Environment Project, An Introduction

COMRADE Executive System Users Manual

COMRADE Data Storage Facility Users Manual

COMRADE Absolute Subroutine Utility Users Manual

COMRADE Data Management System Primer

COMRADE Data Management System Host-Language-Interface Users Manual

COMRADE Data Management System Conversational-Interface Users Manual

COMRADE Design Administration System Users Manual

Inquiries concerning any of these components should be directed to the Computer Systems Development Group, Computer Sciences Division, DTNSRDC.

It is understood and agreed that the U.S. Government shall not be liable for any loss or damage incurred from the use of these computer programs.

TABLE OF CONTENTS

		Page
LIST OF FIGURES		vi
LIST OF TABLES		vi
I.	INTRODUCTION	1
II.	PHILOSOPHY OF THE DESIGN ADMINISTRATION SYSTEM	5
	A. THE ROLE OF THE DESIGN ADMINISTRATION SYSTEM	5
	B. ACCESS AND MANAGEMENT OF PERMANENT FILES	7
	1. The File Directory	7
	2. File Naming	8
	3. File Access Control Mechanisms	13
	4. Recording File Use	14
	C. DESIGN ACTIVITY LOGGING AND REPORTING	15
	1. Logging and Recording of Events	15
	2. Reporting	17
	3. Log Maintenance	19
	D. DESIGN SYSTEM CONTROL AND SUPPORT	20
	1. Maintenance of the Valid Users File	20
	2. Initialization and Termination of Design Sessions	21
	3. Initialization and Termination of Design Projects	22
III.	USE OF THE DESIGN ADMINISTRATION SYSTEM	25
	A. PERMANENT FILE MANAGEMENT CAPABILITIES	25
	1. File Naming Conventions	26
	2. File Access Control	27
	3. Attached CPFMS Files	30
	4. File Management Subroutines	31
	a. CIR Subroutines	32
	COMCAT	32
	COMATC	34
	COMPRG	35
	COMUNL	36
	b. PF Subroutines (CATLOG, ATTACH, PURGE, RENAME)	36
	5. The Non-Resident Utility Program FILEOP	40
	6. Command Procedure COMRFILES	45
	B. LOGGING AND REPORTING CAPABILITIES	49
	Command Procedure REPORT	49
	Command Procedure LOGMAINT	54
	Command Procedure LOGSET	56
	Subroutine COMLOG	57

Page

C. DESIGN CONTROL AND SUPPORT PROCEDURES..... 58
Command Procedure MODUSERS..... 58
Command Procedure INITDES..... 62
Command Procedure DESDONE..... 63

ACKNOWLEDGMENTS..... 67

APPENDIX A - AUXILIARY DESIGN ADMINISTRATION SUBROUTINES.. 69

APPENDIX B - OPERATING PROCEDURES FOR DESIGN ADMINISTRATION SUBROUTINES..... 75

APPENDIX C - USE OF THE COMRADE ABSOLUTE SUBROUTINE LIBRARY..... 77

APPENDIX D - ESTABLISHING DESIGN ADMINISTRATION CAPABILITIES WITHIN A DESIGN SYSTEM..... 83

REFERENCES..... 89

LIST OF FIGURES

1 - DTNSRDC Control Data 6700 Computing Facility..... 2

2 - COMRADE Permanent File Management System..... 9

3 - Sample CPFMS File Directory Structure..... 11

4 - Sample Output From Procedure REPORT..... 18

5 - Use of Procedure COMRFILES, Sample Terminal Session.. 47

6 - Use of Procedure REPORT, Sample Terminal Session..... 50

7 - Use of Procedure LOGMAINT, Sample Terminal Session... 56

8 - Use of Procedure LOGSET, Sample Terminal Session..... 57

9 - Use of Procedure MODUSERS, Sample Terminal Session... 61

10 - Use of Procedure INITDES, Sample Terminal Session.... 63

11 - Use of Procedure DESDONE, Sample Terminal Session.... 65

LIST OF TABLES

1 - Contents of FILEOP Subsystem Common Locations..... 42

2 - Meaning and Format of FILEOP Parameters..... 43

I. INTRODUCTION

The Computer-Aided Design Environment (COMRADE) System[1] provides software designed to facilitate the integration and coordination of programs and data into interactive design systems that can be conveniently used by non-computer specialists such as designers, engineers, and managers.[2] At present, COMRADE comprises three major components--an Executive System,[3] a Data Management System,[4,5] and a Design Administration System.[6] All three components are operational on the DTNSRDC Control Data 6700 SCOPE 3.4 computing system graphically portrayed in Figure 1. This report describes the Design Administration (DA) portion of the COMRADE system.

COMRADE Design Administration capabilities enable managers of large, computer-aided design efforts to control and monitor

---

[1]Rhodes, T. and W. Gorham, "COMRADE - The Computer-Aided Ship Design Environment Project - An Introduction," DTNSRDC Report 76-0001 (Jan 1976).

[2]Brainin, J., "Use of COMRADE in Engineering Design," Paper presented at American Federation of Information Processing Societies National Computer Conference, New York (Jun 1973).

[3]Tinker, R. and I.L. Avrunin, "COMRADE Executive System Users Manual," DTNSRDC Report 76-0002 (Jan 1976).

[4]Gorham, W. et al., "COMRADE Data Management System Host Language Interface Users Manual," DTNSRDC Report 76-0006 (Jan 1976).

[5]Gorham, W. et al., "COMRADE Data Management System Conversational Interface Users Manual," DTNSRDC Report 76-0007 (Jan 1976).

[6]Chernick, C.M., "COMRADE Design Administration System," Paper presented at American Federation of Information Processing Societies National Computer Conference, New York (Jun 1973).

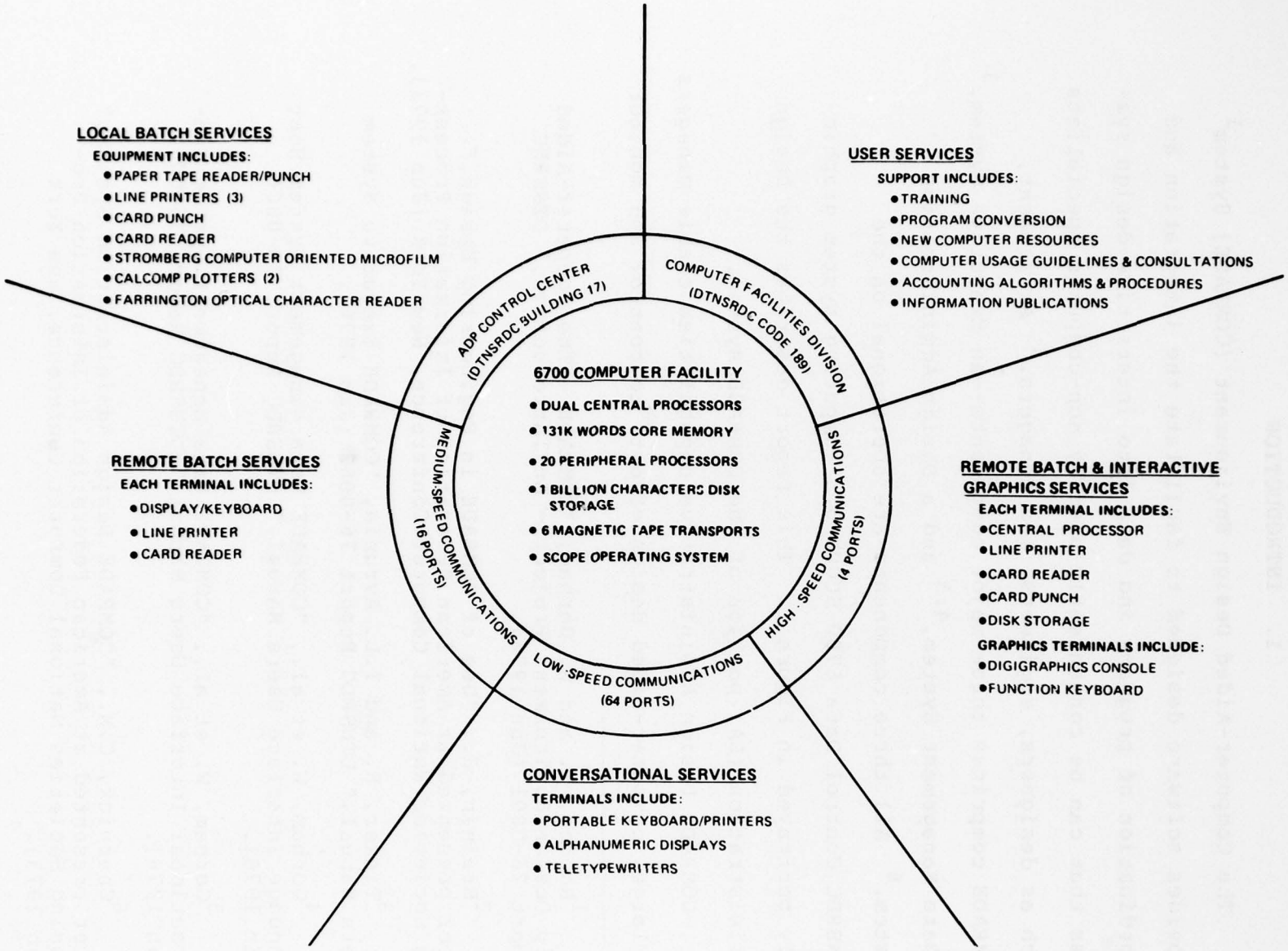


Figure 1 - DTNSRDC Control Data 6700 Computing Facility

their design activity, and hence to regulate access to their COMRADE design systems. Through use of the DA software, major computer-aided design tasks can be organized as sets of projects. DA software capabilities include file access control, design-process monitoring, and design-project initialization and termination.

Fundamental to the functioning of the Design Administration capabilities is a file directory which allows a specific file to be associated with one or more COMRADE design system projects. A "lock and key" mechanism allows the DA System to deny a user access to files not pertinent to his designated area of design responsibility. Access to a file can be further screened through the use of a password, assigned when the file is cataloged, so that only those persons providing the correct password and the correct file access "key" will be granted file access.

In addition to providing file access control, the COMRADE Design Administration System provides procedures for logging and reporting [of the different] design system activities. Reports can be based on an individual designer's activity, the activity within a design project, or the activity within an entire design system. The report can be either comprehensive or confined to only those particular events and dates specified, at the user's option. Printing of these reports can be designated to be done at any one of the predetermined output sites.

Another benefit of the Design Administration System is the support it provides for design system operations in the form of capabilities for initialization and termination of design projects.

The pages that follow describe all of the functions envisioned for the COMRADE Design Administration System, even though current constraints on funding and manpower, and emphasis on other aspects of the COMRADE software, have delayed a complete implementation. The philosophy, concepts, and terminology of the Design Administration System are described in Section II which serves as an overview of eventual DA capabilities and methodology. Section III discusses some of the DA conventions and identifies specific procedures currently being used for the development and operation of an integrated ship design system.[7] Use of these procedures, as well as the FORTRAN-callable file management subroutines presently available, is described.

The COMRADE System in general and the Design Administration System in particular are in a continual state of evolution and refinement. The DA capabilities described in Section III (with the exception of the DESDONE procedure) were operational as of January 1976; any significant changes will be documented as future revisions to this report.

II. PHILOSOPHY OF THE DESIGN ADMINISTRATION SYSTEM

A. THE ROLE OF THE DESIGN ADMINISTRATION SYSTEM

The COMRADE Design Administration System supports those organizational and administrative functions required for development and operation of an integrated computer-based design system. Its capabilities enable the manager of a large computer-aided design system to control design activity and to monitor the progress of, and hence, to control the use of, his COMRADE design system. Moreover, the DA software tools allow a manager to subdivide his design workload into sets of individual projects such as would be required by a large engineering firm. An example of an important DA function is maintenance of a file of valid workers for each design project. This file associates a project worker with a set of manager-assigned "keys" that determine the design procedure and file access privileges of that worker.

Another important part of the DA software is the logging and reporting mechanism it provides for keeping track of progress through the design procedures. Reports may be generated based on an individual worker's activity, activity within a project, or activity within an entire system, and they may be either comprehensive or limited to only those events and dates the user specifies. Many of the reports provide data about terminal session costs, thus enabling the design manager to profile design status not only from a milestone point of view but also from a budgetary point of view.

The Design Administration System also supports procedures for initialization and termination of a design project, although these procedures--and many of the other DA capabilities as well--are generally restricted to such authorized persons as project managers or lead designers. Such restrictions are needed since the initialization process, for example, typically involves the creation of several files and sets the stage for the commencement of design work. One of the files created at this time is the file of valid workers for a given design.

To permit the COMRADE DA to serve as a focus for all types of administrative functions involved in the development and operation of a computer-based design system, it has been structured, in part, as an extension of the operating system's file management architecture. File management conventions, logging and reporting methodology, and such other design support functions as initialization and termination of projects within a COMRADE design system are discussed. The procedures provided for the initialization and termination of design projects interact with the DA file management structure.

Although conceived in support of COMRADE computer-aided design systems, the DA software has been developed as a general project management tool available to any COMRADE application system.

B. ACCESS AND MANAGEMENT OF PERMANENT FILES

1. The File Directory

The file management system is critical to any computer-based system, and especially to an integrated design system which can be effective as a productive design tool only if the files can be quickly and easily accessed and if use of these files can be restricted to qualified users. The COMRADE Design Administration Permanent File Management System (CPFMS) maintains a file catalog or directory of the various data files. As currently implemented, both the data files and the files constituting the CPFMS directory are cataloged within the DTNSRDC SCOPE 3.4 operating system and are protected by sets of passwords. The directory files, which are organized as a tree structure, expedite the accessing of the data files.

The link between the file management requests of a COMRADE design system - e.g., a design procedure - and the CPFMS file directory is achieved by means of two sets of user-callable subroutines known as the CPFMS Interface Routines (the CIR subroutines) and the SCOPE permanent file utility routines (the PF subroutines). The CIR subroutines are built on the PF subroutines. Typically, a design procedure will call one of the CIR subroutines which will interrogate the CPFMS file directory and the design system's Subsystem Common communication region[3] and then call the appropriate PF subroutine. Each PF subroutine corresponds functionally to one of the file management operations initiated by way of a SCOPE job

control statement, such as ATTACH, CATALOG, or PURGE. Occasionally a design procedure may need to access a file from the SCOPE permanent file library directly in which case it may simply issue a call to the particular PF subroutine needed. However, note that files cataloged into the CPFMS directory may be accessed only by a CIR subroutine; they cannot be directly accessed by any of the PF subroutines.

The process by which a typical COMRADE design procedure interacts with the Design Administration Permanent File Management System is shown in Figure 2. As the figure illustrates, the CPFMS system is, in effect, an extension of the file management architecture of the SCOPE operating system. Thus naming conventions, access controls, and permissions accorded permanent files managed by way of SCOPE job control statements are supported by CPFMS.

2. File Naming

As mentioned previously, the file directory for the CPFMS is a set of password-protected files hierarchically organized into four conceptual levels. The first level--identified as L1--consists of a single permanent file, named COMR, which lists the names of all design systems currently active within the COMRADE environment. The second, or L2, level of the directory consists of a number of files, one for each active design system, which lists the names of all design projects currently active for the design systems. The next level of the CPFMS directory--the L3 level--also consists of a number of

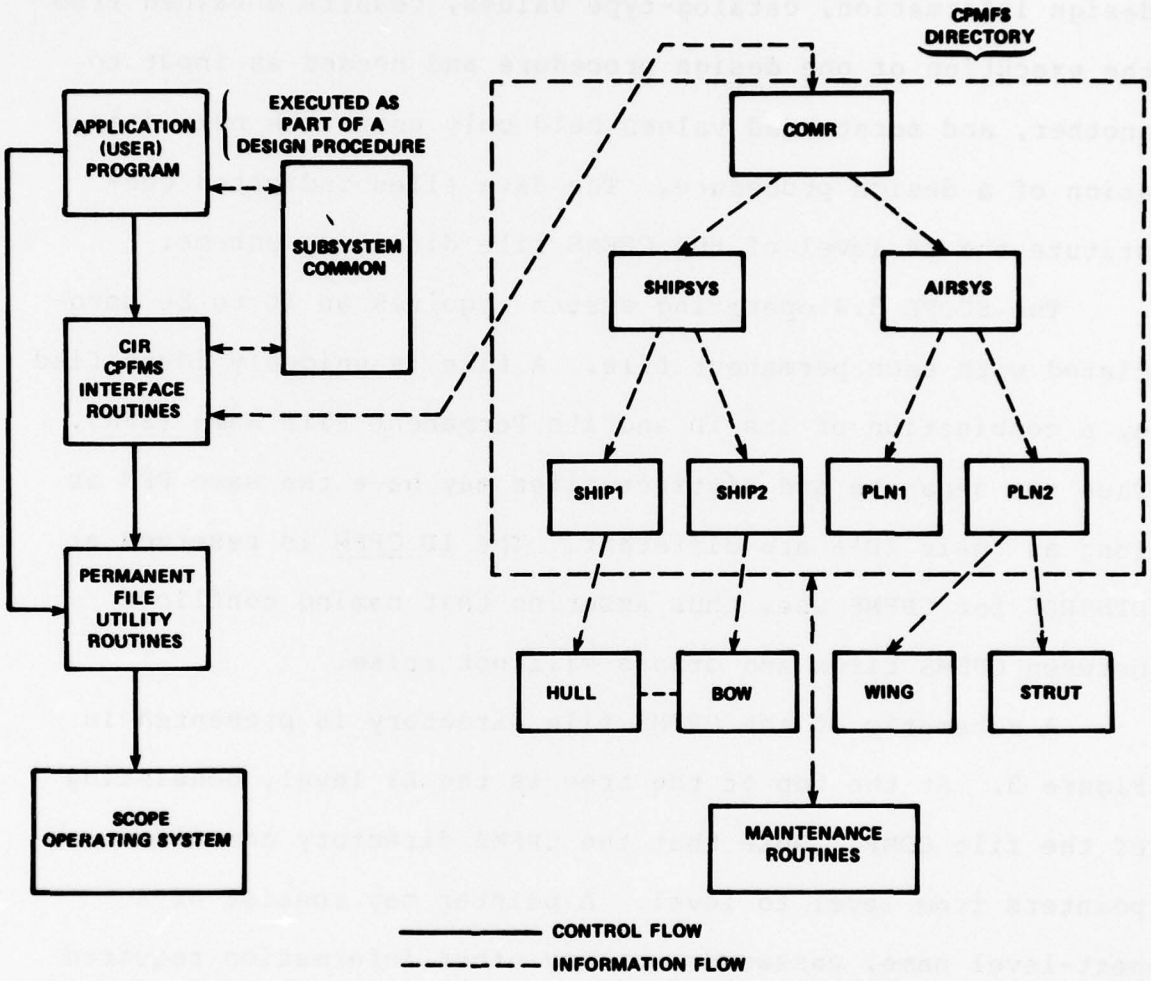


Figure 2 - COMRADE Permanent File Management System

files, one for every active project within the design systems. Each L3 file lists the data files associated with a particular project. Typically, these data files will contain current design information, catalog-type values, results obtained from the execution of one design procedure and needed as input to another, and scratchpad values held only until the next iteration of a design procedure. The data files indicated constitute the L4 level of the CPFMS file directory scheme.

The SCOPE 3.4 operating system requires an ID to be associated with each permanent file. A file is uniquely identified by a combination of its ID and its Permanent File Name (PFN). Thus two separate and distinct files may have the same PFN as long as their ID's are different. The ID CPFM is reserved at DTNSRDC for CPFMS use, thus assuring that naming conflicts between CPFMS files and others will not arise.

A schematic of the CPFMS file directory is presented in Figure 3. At the top of the tree is the L1 level, consisting of the file COMR. Note that the CPFMS directory contains pointers from level to level. A pointer may consist of a next-level name, passwords, or any other information required to access the particular file indicated. The name of the file being pointed to is, by convention, an orderly concatenation of the names provided at higher levels, beginning with the L2 name. Thus the name of an L2 permanent file is simply the name of an active COMRADE design system (these names are recorded on the L1 file). The name of an L3 permanent file is obtained

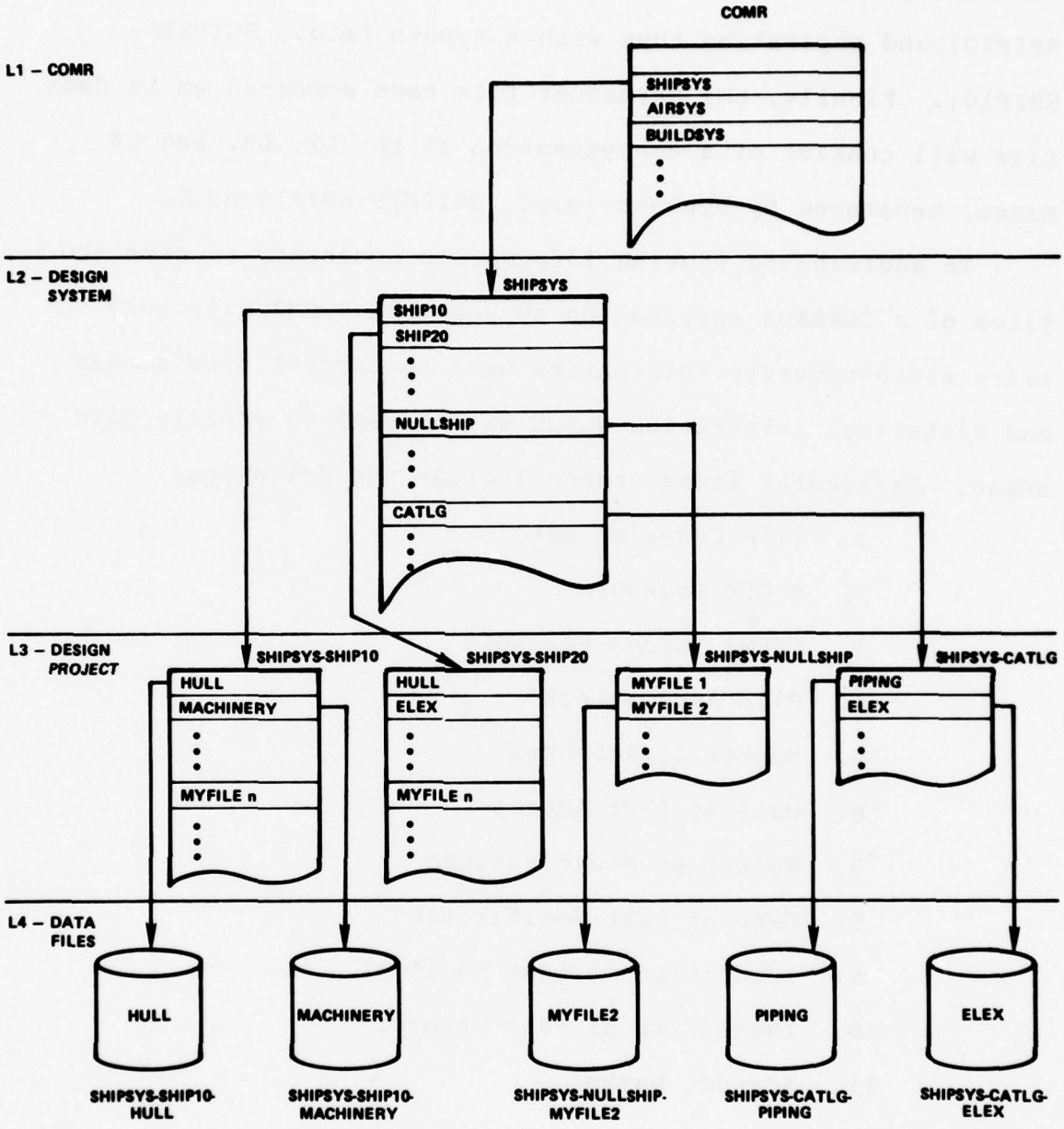


Figure 3 - Sample CPFMS File Directory Structure

by concatenating the name of the project's parent system (e.g., SHIPSYS) with the name of an active project design (e.g., SHIP10) and separating them with a hyphen (e.g., SHIPSYS-SHIP10). Finally, the permanent file name accorded an L4 data file will consist of a concatenation of the L2, L3, and L4 names, separated by hyphens (e.g., SHIPSYS-SHIP10-HULL).

In addition to storing information necessary to name the files of a COMRADE application system, the CPFMS directory entries store security information used to control file access and historical information which may be used to profile file usage. Particular items stored include the following:

1. File Creation Date
2. SCOPE Password
3. CPFMS Pseudo Password
4. File Access Lock
5. Number of Accesses
6. Date of Last Access
7. Number of Modifications
8. Date of Last Modification
9. Identifier of Last Modifier
10. Identifier of File Creator
11. Account Number

3. File Access Control Mechanisms

Control over access to a CPFMS file is imposed using two different privacy mechanisms: the pseudo-password and the file access lock. A pseudo password (not the actual SCOPE system permanent file password) may be assigned to a CPFMS file at the time the file is cataloged. When a request for this file is issued, the pseudo-password for that file is checked by CPFMS. As indicated by Figure 4, both the pseudo-password and the SCOPE password are stored in a CPFMS directory entry.

The second mechanism for controlling file access is the file-access-lock (FAL)/file-access-key (FAK) check. The FAL associated with each file is usually assigned to the file by its creator at the time the file is cataloged. A portion of a FAL may be associated with a different subset of a project effort, perhaps the hull portion of a ship design, for example. Each COMRADE user is assigned his own FAK which is subdivided into three fields, called subFAK's, that pertain to three types of permission: read only, read and write, and purge. When a user requests use of a file, a CPFMS routine compares one of his subFAK's with the FAL for that file. This scheme allows access to files to be restricted to qualified subsets of users through the proper assignment of FAL's and FAK's.

The file privacy mechanisms described allow certain portions of a design task to be reserved for particular members of a design project, even though other members of that design project

are authorized to perform other portions of the design task. Similarly, a user with cognizance over several private files may effectively prevent access to those private files by others by judicious use of the pseudo password.

4. Recording File Use*

Among the items indicated in Figure 4 are several pieces of information pertaining to a file's history: for example, the name of the file creator and the file creation date; the account number assigned for file storage billing; the dates when the file was last accessed and last modified; and the name of the person who last modified or altered the file's contents.

Recorded file use information can be especially useful in determining the degree of control needed for a computer-based design system that comprises many program files and data files. The information contained in the entries of the CPFMS directory forms the basis for such control. For example, on the basis of the recorded file-use information, the project manager can compile a file history which allows him to evaluate one aspect of a design system's activity. It may be that such monitoring will reveal that preferred sequences of design procedures simply obviate the need for certain files to be stored on-line. In such a case it would probably be economical to archive the file to an off-line medium and restore it only when it is needed.

---

*The mechanism for reporting file use information has not yet been designed.

C. DESIGN ACTIVITY LOGGING AND REPORTING

The logging and reporting mechanism of the COMRADE Design Administration System allows managers of large computer-aided design efforts to monitor design activity. Fundamental to this capability is a SCOPE permanent file that contains information about events performed in a given design system. When events have been logged, the logged information remains in the log file until someone in authority requests that it be deleted. At present, no capability for automatic deletion of this information exists.

1. Logging and Recording of Events

At present, three types of events are logged in: a sign-on to a design system; the start of work on a particular design; and the invoking of a design procedure command. Provision has been made in COMRADE for other events to be defined and, optionally, logged. At certain points in a program, a programmer can declare events by inserting calls to the subroutine COMLOG which will place entries into the log file. During program execution, each call to COMLOG will signify the occurrence of an event. The programmer identifies the type of event that has occurred by recording a unique pair of identifiers for that event. Each identifier consists of two integers, one known as the primary type and the other as the secondary type, the latter used strictly to distinguish between entries of the same primary type. The primary-type integer may also be used during selective logging of event sequences

to indicate the relative importance of the various events (for example, update of a design file may be a milestone event, whereas creation of a file of scratchpad values may not be as significant). The more important an event, the lower the primary type integer assigned to it. Selective logging results in the logging of only those events whose primary type is lower than or equal to a predefined primary type. Normally, this predefined primary-type, also known as the "high type to log" or HTL value, is manager assigned, specified when the log for the system is generated. Once specified, the HTL may be increased or decreased by authorized personnel. An HTL value of zero indicates that no events are to be logged.

After the importance of the different classes has been determined and the primary types for the events have been assigned, selective logging will proceed automatically whenever a design procedure is executed. When called, COMLOG performs an HTL test. If the HTL is equal to zero, no logging of design procedure event entries will take place. If HTL is not equal to zero, a comparison between the primary type for the procedure and the HTL will be made and logging performed as appropriate.

In summary, the capability for selective logging enables the manager to achieve selective recording of his design system's activities; it also serves as a convenient tool for debugging a design system. To illustrate, while a system is undergoing development, the HTL value for its log can be set

quite high, thereby assuring that events ordinarily considered insignificant will be logged. When the system is functioning to specifications, the HTL may be lowered so that only milestone events will be logged.

2. Reporting

A procedure entitled REPORT has been developed to generate reports on the information in the log. Selective reporting is enabled through the use of a parameter identified as the "highest level to report" or HLVN value. REPORT can be made to query a system's event log and to report on only those design procedure events having a primary type less than or equal to HLVN. Thus reports may be either comprehensive, reflecting all system activity, or selective, reporting on only those events predetermined to be important, such as design file updates, for example.

Certain other parameters may be specified during the REPORT procedure: to determine at which site the report shall be printed; which project shall be reported on; which activities shall be included; what period of time shall be covered; and what the maximum number of lines that may be printed shall be.

A report typical of the type requested during preliminary aircraft design as performed by the system identified as AERO is provided in Figure 4. In this example, the period reported on extends from 12 November 1975 through 13 November 1975, the

value of HLVN is 30, and three terminal sessions are covered. On 12 November 1975 a user known as CASJJONES entered the AERO system to work on a project known as VSTOL6. He executed two design procedures, FUEL and DRAG, at a total terminal session cost of $1.30. On the same date a user known as CARSSMITH entered the AERO system for a terminal session which cost a total of $8.61. The CP and dollar values given represent central processor unit times and charges for the DTNSRDC CDC-6700 computer.

```
REPORT    11/13/75 - FOR PERIOD FROM 11/12/75 THRU 11/13/75
          HIGHEST LEVEL REPORTED = 30


11/12/75  CASJJONES    VSTOL6
 09:22  AERO ENTERED
 09:23    FUEL     - CP     .37  $    .25
 09:42    DRAG     - CP    4.64  $    .97     ECP     4.27  EL  $    .72
 09:52    SIGNOFF  - CP   11.43  $   1.30     ECP     6.59  EL  $    .33

      ESTIMATED COST = $   1.30

          CARSSMITH    DESIGN4
 10:35  AERO ENTERED
 10:37    STRUTS   - CP     .44  $    .34
 10:56    AIRFL    - CP   22.76  $   4.39     ECP    20.32  EL  $   3.75
 11:07    SIGNOFF  - CP   46.55  $   8.61     ECP    25.79  EL  $   4.52

      ESTIMATED COST = $   8.61

11/13/75  CARSSMITH    DESIGN4
 08:33  AERO ENTERED
 08:33    STRUT1   - CP     .54  $    .34
 08:42    STRUTS   - CP   12.46  $   1.45     ECP    11.92  EL  $   1.11
 08:57    STRUT1   - CP   31.62  $   5.83     ECP    19.16  EL  $   4.38
 09:12    SIGNOFF  - CP   43.92  $   8.94     ECP    12.30  EL  $   3.11

      ESTIMATED COST = $   8.94


REPORT COMPLETED
```

Figure 4 - Sample Output from Procedure REPORT

3. Log Maintenance

When an event has been logged, it remains in the log file until deleted by authorized personnel. Old entries need to be deleted so that the log can be kept as compact as possible. If the user chooses, an entry being deleted may be archived on an off-line storage medium such as a magnetic tape reel. This data base of archived log information can serve both as an historical account of the activities within a particular system and as backup information for previously generated hard copy reports.

The log maintenance procedure requires a user to specify a "save date." All event entries made in the log file before the "save date" are deleted. If the deleted entries are to be archived, the user must supply the identity of the tape reel which is eventually to receive the archived information. Meanwhile, the deleted entries are dumped onto a temporary mass storage file which is then cataloged in the SCOPE permanent file system. The log maintenance procedure will then automatically submit a job to the batch input queue which will result in the copying of the temporary file of deleted entries onto the designated tape reel, thus permanently preserving them for future reference. If hardcopy is to be produced from archived entries, the entries must first be restored to the log file before the reporting procedure is executed.*

---

*The procedure to restore the archived entries has not yet been designed.

D. DESIGN SYSTEM CONTROL AND SUPPORT

The design system control and support subsets of the COMRADE Design Administration System differ widely; however, both are integral to the successful performance of any project. These subsets are responsible for maintaining the file of valid users (i.e., workers) for a given design project, for initializing and terminating procedures within the design system, and for initializing and terminating the project itself.

1. Maintenance of the Valid Users File

Each COMRADE application system design has its own file of valid users. This file is a password-protected SCOPE permanent file which contains the names, file access keys, and design-procedure access keys for all workers on a given design project. (The design procedure lock and key mechanism described by Tinker[3] is nearly identical in concept to the CPFMS lock and key mechanism, described earlier in the section "File Access Control Mechanisms.") The procedure developed for maintaining the valid users file can perform the following: display all user names; display all user names with file and procedure access keys; add and delete user names; and search for, display the name of, and modify a particular user's keys. Typically, use of the procedure is reserved for the project manager or lead designer, since the file of valid users provides the framework for access control. Thus the integrity of a design is dependent upon the judicious maintenance of the access control information in the valid users file.

2. Initialization and Termination of Design Sessions

Design Administration requires a user to sign on to a design system before he can perform any work on a particular design. Many of the functions performed by the sign-on procedure are normally system dependent and are concerned with initialization of such system and design operations as (1) the creation of a temporary log file which will be merged with the actual log at system sign-off time; (2) the retrieval of the Level-Two (L2) and Level-Three (L3) files required for CPFMS operations; and (3) the retrieval of the file of valid users, the design procedure control block file (i.e., the PCB file), and the system file environment file (FEF). (The PCB and the FEF files are used by the COMRADE Executive and are described in detail in Tinker.[3])

After all pertinent files have been readied, the user's right to be working on the requested design is checked. If he is an authorized user (as indicated in the file of valid users), appropriate log entries are made (in the temporary log) and he is informed that he may begin to issue design procedure commands. If he is not an authorized user, the system will deny him access and will print an informative error message.

When the user has completed work on the indicated design, he may wish to change design projects. In such instances, the valid users file which applied to the design project just worked on is released and a new users file and an L3 file appropriate for the next design project to be worked on are

retrieved and readied, and the right of access is again checked. If the user is authorized to work on the new design project, he can begin executing design procedures which may or may not be the same as those executed for the previous design. The user's privileges may vary from one project to another.

After the user has finished his design session, he must terminate system operations by executing the sign-off procedure. The sign-off procedure, executed merely by the user issuing the command SIGNOFF, is quite straightforward and involves release of the Level-Two and Level-Three CPFMS files, release of the PCB, FEF, and valid-users files, and placement of an entry denoting the sign-off into the event log. The final entry records the sign-off time and any other pertinent information.

3. Initialization and Termination of Design Projects

Before any design procedures for a particular project can be executed, someone--probably the project manager or design coordinator--must have already established the project as a work unit by performing the design initialization procedure. The process of closing down a design after all productive work has halted is performed by the design-termination procedure. Again, the right to use both of these procedures is normally restricted to authorized personnel.

The function of the design initialization procedure is threefold: to create a file of valid users of the design; to establish the appropriate pointer in the CPFMS directory (at Levels two and three) so that all files associated with the

design may be managed properly by the CPFMS; and, usually, to initialize the design file, which is both the repository of all data associated with the design and the composite of data produced by the various design disciplines.

When the file of valid users for a design is first created, only the name of the user currently executing the design initialization procedure is stored on the file. At some future time, this user may call upon the procedure which maintains the valid users file and store additional names, thereby fully defining the technical staff who will eventually work on the design project.

The design project initialization procedures just described have been operational for some time. The following design termination procedure has not yet been implemented.

Termination of a design project can be characterized as the orderly archival or deletion of files no longer needed. The termination procedure is performed when the work on a particular portion of the design has been completed or has reached an interim stopping point. Since files that have been deleted usually cannot be recovered and restored to an on-line status, generally only temporary files--such as a scratchpad of values--are deleted. Important files such as the design file and any catalog files are archived.

The archiving of a file is accomplished by copying the file onto a secondary storage medium such as a magnetic tape reel. All the files for a given design may be archived at

once, upon command; the user need merely identify his off-line storage medium and the Level-Three name in the CPFMS directory structure (i.e., the design name) of the project containing the files to be archived. The deletion of all files at once may be accomplished in a similar manner. Files may be individually archived and deleted if the user prefers.

Archived files may be restored at the option of an authorized user (again, probably the manager or lead designer). If work on an archived design is to resume, for example, pertinent files will have to be removed from the magnetic tape and be placed onto on-line mass storage, and the CPFMS directory will then have to be updated to reflect this restoration. Only then can work on the design be executed.

III. USE OF THE DESIGN ADMINISTRATION SYSTEM

Detailed descriptions of the capabilities (i.e., procedures and FORTRAN-callable subroutines) of the COMRADE Design Administration System follow. Several conventions have been observed in preparing these descriptions. In subroutine calling formats, user-supplied parameters have been underlined and parameters returned by the subroutines have been overlined. In the description of each command procedure (Section III, B), a figure is included to illustrate dialogue employed at the terminal during procedure execution; underlining indicates information entered by the terminal user. Numbers at the left of the lines in each figure are used to facilitate discussion of the procedures.

Appendix A describes a set of utility subroutines which have been found useful to Design Administration System programmers. Appendixes B and C describe procedures for accessing DA subroutines for the purpose of constructing and executing application programs. Appendix D discusses those steps which must be performed in order to set up DA capabilities (i.e., CPFMS, REPORT, INITDES, etc.) for use within a particular design system. Generally, use of the DA capabilities will be restricted to authorized system personnel.

A. PERMANENT FILE MANAGEMENT CAPABILITIES

Recall that the file management system is critical to any computer-based system and especially to an integrated design system which can be effective as a production design tool only

if the files can be quickly and easily accessed by qualified users. The COMRADE Permanent File Management System (CPFMS) affords the COMRADE design system user and programmer/developer a file management capability not available through the SCOPE permanent file system alone. CPFMS limits file access to qualified design project personnel. The access control mechanisms require that CPFMS files be accessed only through the CPFMS system.

1. File Naming Conventions

CPFMS is an extension of the SCOPE 3.4 permanent file management architecture wherein all files are uniquely identified by the permanent file name (PFN) and ID. To distinguish them from other files, CPFMS files have the ID CPFM. With one exception, discussed below, each CPFMS PFN will consist of from one to three level names (corresponding conceptually to Levels Two through Four of the CPFMS directory) known as the L2, L3, and L4 names; a level name may consist of from one to eight characters. Thus, each PFN will be constructed in one of the following forms:

L2

L2 - L3

L2 - L3 - L4

where

L2 is the name of design system with which the file is associated (e.g., SHIPSYS)

L3 is the name of a design project (e.g., SHIP10)

L4 is a user-assigned name denoting file contents (e.g., HULL)

The hyphen (-) is the mandatory separator of level names.

The one exception to the above convention is the file having the name COMR (i.e., the first level CPFMS directory file). This file records the names of all systems using CPFMS. As shown in Figure 3, it is the root of the tree-structured CPFMS directory.

2. File Access Control

In general, CPFMS files may be used only from within the design system for which they were created. Control over access to a CPFMS file is maintained using two different privacy mechanisms: the pseudo-password and the file-access lock (FAL)--file access key (FAK). Both the pseudo-password and the FAL may be assigned when a file is cataloged in the CPFMS system.

If a pseudo-password has been assigned, the would-be file user may be required to submit the pseudo-password for a check. If the submitted and the assigned pseudo-passwords are identical, the user passes the check.

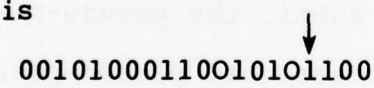
The FAL-FAK check is more complex. The FAL assigned to a file is contained in one 20-bit field. Each bit in a FAL may be associated with a different area of a project effort (e.g., the piping portion of a ship design). Each project user is associated with a 60-bit FAK that is subdivided into three

20-bit fields, called sub-FAK's, that pertain to three types of permission:

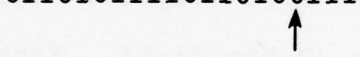
BITS 59-40	BITS 39-20	BITS 19-0	FAK = 1 CDC Computer Word (60 bits)
PURGE Permission	READ-WRITE Permission	READ-ONLY Permission	

Each of the 20-bit positions of a sub-FAK is itself associated with a different project area. Those sub-FAK bit positions corresponding to a project user's area(s) of responsibility will be set to 1 by way of the MODUSERS procedure (described in the section entitled "Design Control and Support Procedures"). when a worker requests use of a file, the 20-bit positions within a sub-FAK are compared with those of the FAL for that file. For the user to pass the FAL-FAK check, each bit that is a 1 in the file's FAL must be matched by a corresponding 1-bit in the user's sub-FAK.

To illustrate the FAL-FAK check, suppose that a user issues a request to access a file with read-write permission, that the FAL for that file is



and that the user's read-write sub-FAK is



In this instance, the FAL-FAK check will not be satisfied since two of the bit positions (indicated by the arrows) are not equal to 1.

A CPFMS file cannot be attached (with read-write or read-only permission) or purged unless the user has passed the pseudo-password check or the FAL-FAK check, whichever one has been designated by the file creator through a 7-bit entity known as the file access control field (FACF). A FACF is formatted in the following manner:

BIT 6	BITS 5-4	BITS 3-2	BITS 1-0	
F	PC	WC	RC	FAC F= 7 BITS

where

F is the force bit

PC is the purge permission code

WC is the read-write permission code

RC is the read permission code

The PC, WC, and RC codes pertain to purge file requests, read-write attach requests, and read-only attach requests, respectively. Each of these codes is contained in a 2-bit field, allowing for values of 0, 1, 2, or 3. Control over a file's access is specified via the code value according to the following convention:

Value	Meaning
0	File has no pseudo-password; the FAL-FAK check is mandatory

1 Pseudo-password check is optional; perform the FAL-FAK check only if the pseudo-password is not submitted

2 Pseudo-password is mandatory; grant access only to the file creator

3 Pseudo-password is mandatory

As indicated, the pseudo-password check is optional in the second case (Code 1) and will be performed only if the user submits a pseudo-password on his attach or purge request. If a pseudo-password is submitted and the check is not satisfied, access to the file will not be granted.

Note that the pseudo-password check is mandatory in the third and fourth cases (Codes 2 and 3). In these instances, if the user neglects to submit a pseudo-password on his request, access to the file will not be granted.

Finally, note the use of the FACF force bit (i.e., the F bit). The PC, WC, and RC codes are designed such that the would-be file user must satisfy either the pseudo-password check or the FAL-FAK check. Through the use of the force bit (set equal to 1), provision exists for requiring the user to satisfy both the pseudo-password check and the FAL-FAK check.

3. Attached CPFMS Files

After the file access control check(s) have been satisfied, CPFMS will grant a user access to a file. A list of all files to which an active COMRADE design system user has been granted read-write or read-only permission is maintained within a communication region known as Subsystem Common. This list may

be interrogated by design system programmers to determine which CPFMS files have been attached and what permissions are associated with the files. A FORTRAN-callable subroutine with entry point GETCOM is used to obtain information from Subsystem Common. A detailed description of GETCOM is available in Tinker.[3]

The list of attached CPFMS files is contained in Subsystem Common locations 11 through 70. For each attached file there exists a three-word entry formatted as follows:

Word				
1	LFN	Zeroes	C	P
2	L3	Zeroes		
3	L4	Zeroes		

where

LFN is the 7-character logical file name (zero-filled)

C is a 1-bit code signifying that the file was cataloged in the current design procedure

P is a 1-bit code signifying that the file was attached with read-write permission

L3 is the 8-character Level Three name (zero-filled)

L4 is the 8-character Level Four name (zero-filled)

Note that up to 20 CPFMS files may be attached simultaneously. Because the list of attached files is variable, an entry having Word 1 equal to zeroes is used to signal the end of the list.

4. File Management Subroutines

As stated earlier, the link between the file management requests of a COMRADE design system and the CPFMS file directory

is achieved by means of two sets of subroutines: the CPFMS Interface Routines (the CIR subroutines), and the SCOPE permanent file utility routines (the PF subroutines). Typically, a design procedure will call one of the CIR subroutines which, after examining the CPFMS directory and Subsystem Common, will call the proper PF subroutine. Each PF subroutine corresponds to one of the SCOPE file management operations, that is, attach, catalog, purge, etc. Occasionally, a procedure may need to access a file from the SCOPE library directly, in which case it will simply call the appropriate PF subroutine. Note that a file that has been cataloged into CPFMS may be accessed only by a CIR subroutine; it cannot be directly accessed by a PF subroutine.

a. CIR Subroutines

. COMCAT

COMCAT (IERR, LFN, L3, L4, IFACF, IPSPW, IFAL)

Subroutine COMCAT catalogs a file into the CPFMS System. The user must make certain that the file has been placed on a permanent file device prior to the call to COMCAT. (See Subroutine REQUEST in Appendix A.)

Parameters:

IERR Error Code:

= 0 Catalog successful

= 5 L3 does not exist

= 7 LFN already in list of attached files

= 8 List of attached files is full

= 9 Catalog requested on an L4 not under the current L3

= 10 File busy

= 12 L4 already cataloged

= 14 Illegal LFN

= 15 SCOPE directory is full

= 16 Latest index not written on random file

= 17 File not on permanent file device

= 18 Duplicate name on SCOPE directory

= 19 LFN is already a permanent file

= 22 Subsystem Common does not exist

= 23 L3/L4 is already in list of attached files

= 24 CPFMS directory is full

= 25 Timing error

LFN Logical file name, seven characters or less, which is left-justified, zero- or blank-filled

L3 Level-Three name, eight characters or less, left-justified, blank-filled

L4 Level-Four name, eight characters or less, left-justified, blank-filled

IFACF Contains the file access control field (FACF) in the rightmost seven bits; the FACF is described in detail in the section "File Access Control"

IPSPW Pseudo-password, five characters or less, which is left-justified, blank-filled, set equal to zero if no pseudo-password is defined

IFAL Contains the file access lock (FAL) in the rightmost 20 bits; the FAL is also described in the section "File Access Control"

. COMATC

COMATC (IERR, LFN, L3, L4, IPRM, IPSPW)

Subroutine COMATC attaches a CPFMS file

Parameters:

IERR Error code:

= 0 Attach successful

= 1 Illegal pseudo-password

= 2 LFN already in use

= 3 Creator not issuing attach request

= 4 FAK not compatible with FAL

= 5 L3 does not exist

= 6 L4 does not exist

= 7 LFN already in list of attached files

= 8 List of attached files is full

= 9 Write permission requested on an L4 not under the current L3

= 10 File busy

= 14 Illegal LFN

= 22 Subsystem Common does not exist

= 23 L3/L4 already in list of attached files

LFN Logical file name

L3 Level-Three name

L4 Level-Four name

IPRM Permission indicator:

= 0 File is to be attached with read-only permission

= 1 File is to be attached with read-write permission

IPSPW Pseudo-password

. COMPRG

COMPRG (IERR, LFN, L3, L4, IPSPW)

COMPRG purges a CPFMS file, effectively removing it from the directory system.

Parameters:

IERR Error Code:

= 0 Purge successful

= 1 Illegal pseudo-password

= 2 LFN already in use

= 3 Creator not issuing purge request

= 4 FAK not compatible with FAL

= 6 L4 does not exist

= 7 LFN already in list of attached files

= 9 Purge permission requested on an L4 not under the current L3

= 10 File busy

= 14 Illegal LFN

= 22 Subsystem Common does not exist

= 23 L3/L4 already in list of attached files

= 25 Timing error

= 26 LFN in use but not in list of attached files

LFN Logical file name

L3 Level-Three name

L4 Level-Four name

IPSPW Pseudo-password

. COMUNL

COMUNL ($\overline{\text{IERR}}$, $\underline{\text{LFN}}$)

COMUNL unloads a CPFMS a file, effectively making it inaccessible for subsequent processing until it is re-attached.

Parameters:

IERR Error Code:

= 0 Unload successful

= 1 Unknown LFN

= 8 LFN not found in list of attached files

b. PF Subroutines (CATLOG, ATTACH, PURGE, RENAME)

With the exception of the subroutine name the four SCOPE permanent file (PF) utility routines CATLOG, ATTACH, PURGE, and RENAME have the same calling format.

CALL subnam (IERR, LFN, IPFN, ID, ITK, IRD, IEX, IMD, ICN, IMR,
IAC, ICY, IRP, IXR, ILC, IRW, ISN, IVS, IFO, IST)

where

subnam is either CATLOG, ATTACH, PURGE, or RENAME

Except for the error code (i.e., IERR), each of the arguments in the calling format corresponds to a parameter that could be provided by way of the SCOPE job control statements CATALOG, ATTACH, PURGE, and RENAME. Use of many of the arguments is optional; hence, calling formats may be truncated. The only arguments normally required for the subroutines ATTACH, PURGE,

and RENAME are IERR, LFN, IPFN, and ID; the subroutine CATLOG normally requires the four just named, and, in addition, ITK, IRD, IEX, IMD, ICN, IMR, and IAC.

Descriptions of the arguments to the PF subroutines follow. Basic permanent file terminology and concepts may be found in the SCOPE Reference Manual.[8]

IERR Error Code:

= 0 Operation successful
= 1 Illegal account number
= 2 LFN already in use
= 3 Unknown LFN
= 4 No room for extra cycle (limit is 5)
= 5 SCOPE directory is full
= 6 No LFN or PFN
= 8 Latest index not written
= 9 File not on permanent file device
= 10 File not in system
= 11 Archive retrieval aborted
= 12 Illegal cycle number
= 13 Cycle number limit reached
= 14 Permanent file directory is full
= 15 Operation attempted on a non-permanent file
= 16 Operation attempted on a non-local file
= 18 File never assigned a device
= 19 Cycle incomplete
= 20 Permanent file already attached

---

[8]Control Data Corporation, "SCOPE Reference Manual," 6000 Version 3.4, Publication No. 60307200.

= 21 File busy

= 23 Illegal LFN

= 24 File dumped

= 27 Alter needs exclusive access

= 29 File already in system

= 32 Illegal setname

= 33 Device set not mounted at a control point

= 56 Operation cannot be performed

= 57 Security violation

= 58 Illegal file definition block address

LFN Logical file name, seven characters or less, left-justified, zero- or blank-filled

IPFN Permanent file name, 40 characters or less, left-justified, blank-filled

ID User identifier, four characters, left-justified

ITK Turnkey password, nine characters or less, left-justified, blank-filled

IRD Read password, nine characters or less, left-justified, blank-filled

IEX Extend password, nine characters or less, left-justified, blank-filled

IMD Modify password, nine characters or less, left-justified, blank-filled

ICN Control password, nine characters or less, left-justified, blank-filled

IMR Multi-read indicator; if non-zero, only read permission is granted

IAC Account number, ten characters

ICY Cycle number, if 0 (and if the parameter ILC is 0), the next highest cycle will be attached or cataloged; if a positive integer in the range 1-999, an attempt will be made to use that cycle; if -1 (and if ILC is 0), the highest cycle will be used and the cycle number will be returned in location ICY

IRP Retention period in days; the maximum is 999 and the default is 30

IXR Except read password, nine characters or less, left-justified, blank-filled; used as passwords for extend, modify, and control unless those passwords have been specified elsewhere.

ILC Lowest cycle indicator; if non-zero, the lowest cycle will be processed

IRW Read/write indicator; if non-zero, multi-read with single rewrite or extend permission is granted (i.e., caller can modify the file, but others can only read). If zero, and if extend, modify, or control permission is to be granted, then exclusive access to the file is granted.

ISN Volume serial number

IFO File organization indicator, left-justified, blank-filled; possible volumes are

DA	Direct access
IS	Indexed sequential
AK	Actual key

IST Station identifier (reserved for future use)

5. The Non-Resident Utility Program FILEOP

The subroutines just described enable application programmers to embed permanent file management operations within design programs. In certain instances, however, programmers may wish to allocate the required file operations to the COMRADE Execution System,[3] thereby segregating file management functions from those of design and analysis. Two means, both of which must be executed by way of the COMRADE Executive's Procedure Definition Language (PDL),[3] have been developed for achieving this objective. A user who is required to manipulate non-CPFMS files may simply execute the appropriate resident utility routines (i.e., ATTX, CATX, PURX, and RENX) in the manner prescribed in Tinker.[3] If, however, he is required to manipulate CPFMS files, he may execute the non-resident file utility program FILEOP.

FILEOP was designed to be a comprehensive permanent file management capability. In addition to its management of CPFMS files, FILEOP can perform a number of operations on SCOPE (i.e., non-CPFMS) files and can also carry out a set of general file functions. Operations within the purview of FILEOP include the ability to catalog, attach, purge, and unload CPFMS files; to catalog, attach, purge, rename, extend, and determine the permissions of SCOPE files; and to request, close-unload, and close-rewind files. A predefined area of Subsystem Common (locations 79-99 inclusive) is used to specify

the particular function that FILEOP is to initiate, and to pass the parametric information required to execute that function.

Each of the FILEOP functions is identified by a name of seven characters or less:

Function	Function Name
CPFMS Catalog	COMCAT
CPFMS Attach	COMATC
CPFMS Purge	COMPRG
CPFMS Unload	COMUNL
SCOPE Catalog	CATLOG
SCOPE Attach	ATTACH
SCOPE Purge	PURGE
SCOPE Rename	RENAME
SCOPE Extend	EXTEND
SCOPE Determine Permissions	PERM
General Request	REQUEST
General Close-Unload	CLUNLD
General Close-Rewind	CLRWWD

FILEOP will initiate the function named in Subsystem Common location 90. Additional Subsystem Common locations will contain the parametric values required to carry out the function. Locations must be set up in accordance with the information presented in Tables 1 and 2. The paragraphs and examples that follow describe the proper method of executing FILEOP via the Procedure Definition Language (PDL).[3]

TABLE 1 - CONTENTS OF FILEOP SUBSYSTEM COMMON LOCATIONS

	Functions												
Location	Catalog	Rename	Attach	Purge	Extend	Perm	Request	CLRWND	CLUNLD	COMUNL	COMCAT	COMATC	COMPRG
C.79	ID	ID	ID	ID									
C.80	PFN	PFN	PFN	PFN									
C.81	PFN	PFN	PFN	PFN									
C.82	PFN	PFN	PFN	PFN									
C.83	PFN	PFN	PFN	PFN									
C.84	AC	AC											
C.85	TK	TK	TK	TK									
C.86	RD	RD	RD	RD									
C.87	EX	EX	EX	EX									
C.88	MD	MD	MD	MD									
C.89	CN	CN	CN	CN									
C.90	CATLOG	RENAME	ATTACH	PURGE	EXTEND	PERM	REQUEST	CLRWND	CLUNLD	COMUNL	COMCAT	COMATC	COMPRG
C.91	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR	IERR
C.92	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN	LFN
C.93			MR								L3	L3	L3
C.94	RP	RP									L4	L4	L4
C.95	CY	CY	CY	CY							IFACF	IPRM	IPSPW
C.96	XR	XR	XR	XR							IPSPW	IPSPW	
C.97			LC	LC							FLK		
C.98			RW										
C.99													

TABLE 2 - MEANING AND FORMAT OF FILEOP PARAMETERS

Parameter	Meaning	Format
ID	Four-character user ID	Four alphabetic characters, left-justified, blank-filled
PFN	Four-word area containing permanent file name	Forty alphanumeric characters, left-justified, blank-filled
AC	Account number/job order number	Ten numeric display-code characters
TK	Turnkey password (first submitted password)	On RENAME, set to 1 to clear that password; otherwise only nine left-justified alphanumeric characters are considered
RD	Read password (second submitted password)	
EX	Extend password (third submitted password)	
MD	Modify password (fourth submitted password)	
CN	Control password (fifth submitted password)	
IERR	Error code	Numeric; 0 if successful, for other error codes see discussion of the PF subroutines
LFN	Local file name	Seven characters, left-justified, blank-filled
MR	Multiread parameter	0 for multiread, non zero for read only
RP	Retention period in days, maximum of 999 (Default is 30)	Integer (0-999)
CY	Cycle number	Integer (0-999)
XR	Except-read password used as extend, modify and control	Nine characters, left-justified; on RENAME, set to 1
LC	Lowest cycle parameter	Used as discussed for the PF Subroutines
RW	Read/write parameter	
L3	Level-3 name	Eight characters, left-justified, blank-filled
L4	Level-4 name	Eight characters, left-justified, blank-filled
IFACF	File access control field	Seven bits, right-justified
IPSPW	Pseudo password	Five characters, left-justified
FLK	File lock	Twenty octal digits, right-justified
IPRM	Read only/read write parameter	Zero for read-only, 1 for read/write

Before executing a program through PDL, the user must set the permanent file ID of the program into System Common Location 22. In the case of FILEOP, this is done by presetting Location 22 to COMR (with the statement PRESET S.22 = "COMR").

Parameters are defined to FILEOP by presetting the proper locations in Subsystem Common. Locations and their contents depend on the function to be executed. Locations 79 through 99 are generally used. For all functions, the function name is placed in Location 90. Location 92 must contain the logical file name and Location 91 the error code. (Error codes for FILEOP functions are the same as those of the file management subroutines discussed in the section "File Management Subroutines" and Appendix A. Note that FILEOP function names are identical to those of the subroutines.) Normally if a function cannot be performed (e.g., a file to be attached is non-existent), an error message is printed on the file OUTPUT. This message can be suppressed by setting Location 91 to NOMESSAGE before executing FILEOP.

Examples

Example 1.

To attach a CPFMS file which has an L3 name of DDG3 and an L4 name of PIPING as the logical file VALUES, a user might code:

```
PRESET  S.22 = "COMR"
PRESET  C.90 = "COMATC," C.91 = "NOMESSAGE"
PRESET  C.92 = "VALUES," C.93 = "DDG3," C.94 = "PIPING."
PRESET  C.95 = 0, C.96 = "PASWD"
EXECUTE FILEOP
GOTO ERROR (C.91.NE.0)
```

Note that this sequence of PDL statements assumes that write permission is not wanted (C.95 = 0), that the pseudo-password is PASWD (in C.96), and that no message will be printed on OUTPUT. Also note that if the attach procedure is unsuccessful, a jump will be made to the PDL statement labeled ERROR.

Example 2.

To catalog a SCOPE file which has an ID of CANV and a logical file name of BEAMS as the permanent file BUILDINGS with an "except read" password of XREAD, a user might code:

```
PRESET  S.22 = "COMR"
PRESET  C.79 = "CANV"
PRESET  C.80 = "BUILDINGS," C.82(2) = 0
PRESET  C.84 = "0123456789," C.85(5) = 0
PRESET  C.90 = "CATLOG," C.91 = 0, C.92 = "BEAMS"
PRESET  C.93(3) = 0, C.96 = "XREAD," C.97(3) = 0
EXECUTE FILEOP
GOTO ERR (C.91.NE.0)
```

If no errors occur, this sequence of PDL statements will cause the file BEAMS to be cataloged with the next highest cycle number, since C.95 is set to 0. Note that the account number is 0123456789 and that a message will be printed on output if an error condition is encountered.

6. Command Procedure COMRFILES

The Design Administration CIR subroutines and the FILEOP program provide programmers with flexibility in determining how best to satisfy their CPFMS file management requirements. For non-computer-oriented personnel there exists a command procedure named COMRFILES which serves as a conversational CPFMS file manipulation capability. COMRFILES may be used to catalog,

attach, purge, unload, and list CPFMS files. Figure 5 presents an example of the use of COMRFILES. An interpretation of the terminal dialogue follows:

(1) The user initiates the COMRFILES procedure in response to the COMRADE signal (i.e., the ????).

(2) The user indicates that he wishes a listing of the COMR-FILES operations.

(3) The user is given a choice of options. He may catalog (C), attach (A), purge (P), unload (U), or list (L) CPFMS files or he may halt the procedure (H).

(4) The user desires to catalog a file.

(5) The user provides the logical file name and the Level 3 and Level 4 names for the file.

(6) The user specifies that password controls will not be maintained for this particular file. Note that the user is also asked for the 60-bit (20 octal digits) file access lock (FAL), but that the actual bit pattern of the lock is concealed by overprinting.

(7) The user lists those CPFMS files which are available for processing.

(8) The user unloads the CPFMS file having the logical file name TAPE11.

(9) The user verifies that no files are active.

(10) and (11) The user attaches the CPFMS file SHIPSYS-NULLSHIP-TAPE11 as the logical file TAPE12. (Assume that COMRFILES is executed from within the design system SHIPSYS.)

```
(1)  ????COMRFILES
(2)  ENTER YES OR NO FOR LIST OF OPERATIONS:  Y
     COMRADE FILE MANAGEMENT OPERATIONS ARE:
     (C)  CATALOG
     (A)  ATTACH
(3)  (P)  PURGE
     (U)  UNLOAD
     (L)  LIST COMRADE FILES
     (H)  HALT COMMAND
(4)  ENTER OPERATION:  C
     ENTER LOCAL FILE NAME:  TAPE11
     ENTER THE 3RD & 4TH LEVEL FILE NAMES (MAX. OF 8 CHAR.)
(5)  L3:NULLSHIP
     L4:TAPE 11
     ●●●●●●●●●●●●●●●●●●●● ENTER FILE ACCESS LOCK (20 OCTAL DIGITS):
(6)  ANY PASSWORD CONTROLS?  N
     FILE CATALOGED
(7)  ENTER OPERATION:  L

       LFN       L3       L4

     TAPE11   NULLSHIP  TAPE11

(8)  ENTER OPERATION:  U
     ENTER LOCAL FILE NAME:  TAPE11
     FILE UNLOADED
(9)  ENTER OPERATION:  L

     NO COMRADE FILES ATTACHED

(10) ENTER OPERATION:  A
     ENTER LOCAL FILE NAME:  TAPE12
     ENTER THE 3RD & 4TH LEVEL FILE NAMES (MAX. OF 8 CHAR.)
     L3:NULLSHIP
(11) L4:TAPE11
(12) ENTER 0 FOR READ PERMISSION; 1 FOR WRITE:1
(13) ●●●●● ENTER PASSWORD:
     FILE ATTACHED
     ENTER OPERATION:  L

       LFN       L3       L4

(14) TAPE12   NULLSHIP  TAPE11

     ENTER OPERATION:  U
(15) ENTER LOCAL FILE NAME:  TAPE12
     FILE UNLOADED
(16) ENTER OPERATION:  P
     ENTER LOCAL FILE NAME:  TAPEX
     ENTER THE 3RD & 4TH LEVEL FILE NAMES (MAX. OF 8 CHAR.)
(17) L3:NULLSHIP
     L4:TAPE11
     ●●●●● ENTER PASSWORD:
     FILE PURGED
     ENTER OPERATION:  L

     NO COMRADE FILES ATTACHED

(18) ENTER OPERATION:  H

     COMRADE COMRFILES ENDED
```

Figure 5 - Use of Procedure COMRFILES, Samples Terminal Session

(12) The user indicates that the file TAPE12 is to be attached with read/write permission.

(13) The user specifies the five-character pseudo-password also concealed by overprinting.

(14) The user makes certain that the file TAPE12 has been attached.

(15) TAPE12 is unloaded.

(16) and (17) The user purges the file SHIPSYS-NULLSHIP-TAPE11 after having supplied the pseudo-password.

(18) The user terminates COMRFILES.

B. LOGGING AND REPORTING CAPABILITIES

. Command Procedure REPORT

Purpose: Produces reports about COMRADE design system activity.

Explanation: REPORT is used to generate a variety of reports which can be used to monitor the progress of a computer-aided design effort. Reports can be generated for an individual designer's activity, the activity within a project, or the activity within a design system. Reports may be comprehensive or limited to only those procedures and dates the user specifies.

Example: Figure 6 provides an example of the use of the REPORT procedure. An interpretation of the terminal dialogue follows:

(1) The user initiates the REPORT procedure in response to the *COMRADE signal*.

(2) The user indicates that he desires to be prompted for parametric values.

(3) The user is asked to enter values for items such as starting date (SD), last date (LD), design project name (DSN), design procedure command name (CMND). Default values for these parameters are as follows:

SD	Today's date less one week
LD	Today's date
USN	ALL
DSN	ALL or the design project name active as a result of this terminal session
HLVN	ALL

Figure 6 - Use of Procedure REPORT, Sample Terminal Session

```
(1)  ????REPORT

(2)  ENTER PROMPT OR NOPROMPT-PROMPT
     WHENEVER INPUT REQUESTED :
     ENTER RUN TO GENERATE REPORT,
           HALT TO HALT COMMAND, OR
           LIST TO LIST PARAMETERS

     ENTER STARTING DATE OF REPORT (MM/DD/YY)  SD=10/01/75
     ENTER LAST DATE OF REPORT (MM/DD/YY)  LD=10/31/75
     ENTER USER NAME OR "ALL"  USN=ALL
     ENTER DESIGN NAME TO SCAN FOR OR "ALL"  DSN=ALL
     ENTER MAXIMUM NUMBER OF LINES TO PRINT  MXLN=20
(3)  ENTER HIGHEST LEVEL NUMBER TO REPORT OR "ALL"  HLVN=ALL
     ENTER SITE WHERE OUTPUT IS TO BE PRINTED
     EITHER HERE
            CDC6700
            CMD1700
            NAVSEC
         OR OTHER
     SITE=HERE
     ENTER FORMAT OF REPORT
          0  FOR NO CP TIMES
          1  FOR TIMES BETWEEN COMMANDS ON SAME LINE
          2  FOR TIMES BETWEEN COMMANDS ON NEXT LINE
(4)        FORMAT=1
     ENTER COMMAND NAME TO SCAN FOR OR "ALL"  CMND=ALL
(5)  ENTER RUN TO GENERATE REPORT,
           HALT TO HALT COMMAND, OR
           LIST TO LIST PARAMETERS

(6)  READY  -LIST
     SD=     10/01/75
     LD=     10/31/75
     USN=    ALL
(7)  DSN=    ALL
     HLVN=   ALL
     MXLN=   20
     SITE=   HERE
     FORMAT= 1
     CMND=   ALL
(8)  READY  -LD=10/15/75
(9)  READY  -MXLN=25
     READY  -LIST
     SD=     10/01/75
     LD=     10/15/75
     USN=    ALL
(10) DSN=    ALL
     HLVN=   ALL
     MXLN=   25
     SITE=   HERE
     FORMAT= 1
     CMND=   ALL
(11) READY  -RUN
```

Figure 6 (Continued)

```
REPORT    12/ 9/75  -  FOR PERIOD FROM 10/01/75 THRU 10/15/75
10/ 1/74   CATCCORIN   DDG3
 10:12   ISDS ENTERED
 10:13     COMMAND  - CP    3.23     $    .65
 10:15     COMRFILES- CP    5.29     $   1.72  ECP    2.06  $    1.07
 10:19     LOGOUT   - CP    7.18     $   3.31  ECP    1.88  $    1.50
      ESTIMATED COST =  $  3.31

           CATCCORIN    DDG3
(12) 10:23   ISDS ENTERED
 10:25     COMRFILES- CP    3.53     $   1.19
 10:29     SIGNOFF  - CP    7.54     $   3.18  ECP    4.00  $    1.99
      ESTIMATED COST =  $  3.18

           COMRSYSTEM   SYSSHIP
 16:08   ISDS ENTERED
 16:10     HELP     - CP    4.16     $    .65
 16:14     COMMAND  - CP    4.16     $    .95  ECP    0.00  $     .30
 16:17     PDL      - CP    5.48     $   2.10  ECP    1.32  $    1.15
 16:20     COMMAND  - CP   11.99     $   3.12  ECP    6.51  $    1.02
 16:33     SIGNOFF  - CP   19.69     $   6.91  ECP    7.70  $    3.79

  LINE COUNT EXCEEDED

(13) READY   -SITE=CDC6700
     READY   -LIST
     SD=      10/01/75
     LD=      10/15/75
     USN=     ALL
     DSN=     ALL
(14) HLVN=    ALL
     MXLN=    25
     SITE=    CDC6700
     FORMAT=  1
     CMND=    ALL
(15) READY   -MXLN=500
(16) READY   -RUN
      REPORT COMPLETED

(17) READY   -USN=CARSSHORE
     READY   -SITE=HERE
     READY   -MXLN=25
     READY   -LIST
     SD=      10/01/75
     LD=      10/15/75
     USN=     CARSSHORE
     DSN=     ALL
     HLVN=    ALL
     MXLN=    25
     SITE=    HERE
     FORMAT=  1
     CMND=    ALL
(18) READY   -RUN

     NO INFORMATION TO REPORT
(19) READY   -HALT
     REPORT GENERATOR TERMINATED
```

MXLN	Unlimited
SITE	HERE (i.e., the conversational terminal)
FORMAT	1
CMND	ALL

(4) A report format is chosen by the user. Three options with regard to report format are available: No central processor (CP) times or estimated costs will be printed (0); for each design procedure the CP time and costs will be printed on the same line as the procedure name (1); or, the elapsed CP time and costs will be printed on a line immediately following the one containing the procedure name (2).

(5) The user is given a choice of options. He may

- generate a report in accordance with the current parameter values (RUN),
- print the current parameter values (LIST), or
- terminate the procedure (HALT)

(6) The user chooses to print nine parameters.

(7) Note the values printed alongside their abbreviations. A value may be redefined whenever the signal READY- appears. The user need only provide one of the abbreviations and follow it with an equal sign and the new value.

(8) and (9) The user redefines the last date (LD) and the maximum number of lines for the report (MXLN). He could have specified the word PROMPT which would signify that he wishes to redefine all values.

(10) Parameter values are once again printed.

(11) The user now generates a report.

(12) The report is printed at the terminal. Note that during the report period three terminal sessions (two by a user identified as CATCCORIN and one by COMRSYSTEM) were recorded. The report indicates that several procedures (e.g., COMMAND, COMRFILES, PDL) were executed in conjunction with two design projects (DDG3 and SYSSHIP).

(13) Since the line count for the report was exceeded, the user redefines the output site to avoid using the slow-speed terminal.

(14) Having forgotten the MXLN value, the user lists the nine REPORT parameters.

(15) The user redefines the MXLN value.

(16) The previous report is regenerated.

(17) The user redefines the user name so as to limit the next report to the activity of a single individual (CARSSHORE).

(18) A second report is generated. Note that any report is to be printed on the terminal instead of on the CDC 6700.

(19) The user terminates REPORT having completed his work.

. Command Procedure LOGMAINT

Purpose: Removes old information from the various design system logs and, optionally, archives this information onto magnetic tape.

Explanation: LOGMAINT is used to maintain logs which contain information regarding COMRADE design system events. Generally, three types of events are logged: a sign-on to a design system, the start of work on a particular design project, and initiation of a design procedure. Once information concerning an event is logged, it remains in the log file until someone in authority requests its deletion. At that time entries pertaining to the information are deleted (and optionally achieved) if their dates precede to the date provided to LOGMAINT.

Example: Figure 7 provides an example of the use of LOGMAINT. An interpretation of the dialogue follows:

(1) The user initiates the LOGMAINT procedure.

(2) The user is informed that at any time he may terminate LOGMAINT simply by entering HALT or EXIT.

(3) The user provides a save date; [before which time] all information in the active log (i.e., the log identified as a result of having previously signed-on to a particular design system) before that time will be deleted. The format of the date, as indicated, must be MM/DD/YY; to guard against accidental loss of recently logged information. The date must be at least 30 days before the date on which LOGMAINT is executed.

(4) The user chooses to archive the deleted information onto magnetic tape. Whether or not the user archives the information, LOGMAINT will ask him to verify his deletion request.

(5) The user indicates that a labeled tape is to be used for storing the archived information; the label may contain as many as 17 alphabetic characters.

(6) The user provides a slot number which identifies a tape not permanently assigned to the DTNSRDC Computer Center. A slot number, consisting of from one to three digits, is temporarily assigned by the Computer Center staff. If a permanently-assigned tape is to be used, the user simply enters a blank and follows it immediately with a carriage return.

(7) The user provides a visual reel number consisting of from one to ten alphanumeric characters.

(8) The user provides an organization code which may also consist of from one to ten alphanumeric characters. The code identifies a batch job which is executed to copy the deleted entries onto the magnetic tape reel. Batch processing is required, since interactive jobs may not use tapes.

(9) The user verifies his deletion request.

(10) LOGMAINT indicates that it has completed its work. The permanent file space required by the log file is effectively reduced.

```
(1)     ????LOGMAINT
(2)     ON ANY INPUT REQUEST ENTER (HALT) OR (EXIT) TO
        TERMINATE
(3)     DELETE ALL ENTRIES BEFORE (MM/DD/YY): 03/12/74
(4)     ENTER (YES) OR (NO) TO SAVE DELETED ENTRIES ON TAPE =
        YES
(5)     FOR LABELED TAPE ENTER LABEL, ELSE BLANK
        : COMRTESTTAPE
(6)     FOR SLOT TAPE ENTER SLOT NUMBER, ELSE BLANK = 34
(7)     ENTER TAPE VISUAL REEL NUMBER = CA0000
(8)     ENTER YOUR ORGANIZATIONAL CODE : NAVSEC6102
(9)     ARE YOU SURE YOU WANT TO DELETE OLD ENTRIES?
        ENTER (YES) OR (NO) - YES
(10)    LOGMAINT COMPLETED
```

Figure 7 - Use of Procedure LOGMAINT, Sample Terminal Session

. Command Procedure LOGSET

Purpose: Prints the current "high type to log" or HTL value and allows a new value to be specified.

Explanation: LOGSET allows specification of events to be recorded in the log for the active design system. Only events with a primary type equal to or less than the log file's HTL value will be recorded. The value of HTL must be an integer in the range 1-63; if logging is to be turned off altogether, a 0 must be entered as the HTL.

Example: Figure 8 provides an example of the use of LOGSET. An interpretation of the dialogue follows:

(1) The user initiates the LOGSET procedure.

(2) The current value of the log file's HTL is printed.

(3) The user is asked to specify a new value for HTL or to specify EXIT. If EXIT is specified, the current HTL value will be retained and the procedure will terminate. In this example, the user provides a value of 66.

(4) Following notification that 66 is an illegal value, the user is given another opportunity to enter a new HTL value.

(5) The procedure prints the new HTL value and then terminates.

```
(1)     ????LOGSET
(2)     CURRENT HIGH TYPE TO LOG IS 63
(3)     ENTER NEW VALUE OR EXIT - 66
        ILLEGAL VALUE SPECIFIED
(4)     ENTER NEW VALUE OR EXIT - 61
(5)     CURRENT HIGH TYPE TO LOG IS 61
        ????
```

Figure 8 - Use of Procedure LOGSET, Sample Terminal Session

. Subroutine COMLOG

COMLOG(IERR, IPT, IST, IENTRY)

COMLOG places new entries in the log for the active design system. Presently, three classes of system events are logged: a sign-on to a system, the start of work on a particular design, and the invoking of a design procedure.

Additional events may be logged with COMLOG, but only after the importance of all the different event classes has been determined and the primary types for the events have been assigned by the project manager.

Parameters:

IERR	Error Code: = 0 Successful operation = 1 Illegal IPT value = 2 Illegal IST value = 3 Illegal entry length
IPT	Primary type of entry; must be in the range 1-63
IST	Secondary type of entry; must be in the range 1-63
IENTRY	An array of information which will be written directly to the log; the array must terminate with a word of zeroes and must not exceed 80 characters.

C. DESIGN CONTROL AND SUPPORT PROCEDURES

. Command Procedure MODUSERS

Purpose: Maintains the file of valid users of a design project.

Explanation: MODUSERS is used to maintain those files that are used in controlling access to a design system's procedures and files. Each entry in a MODUSERS file will contain three items of information: a 10-character INTERCOM user ID

and two 60-bit "keys" used for controlling access to a project's procedures and CPFMS files. Operations performed on a MODUSERS file are as follows: print all user ID's and keys; modify a particular user's keys; and add/delete a user and his keys.

Example: Figure 9 shows an example of the use of MODUSERS. An interpretation of the dialogue follows:

(1) The user initiates MODUSERS.

(2) The user is informed that the file of valid users of ship design project SHIP10 has been activated.

(3) The user indicates that he wishes to see a list of options available under MODUSERS.

(4) The user is given a choice of options. He may

- Terminate the procedure (0)
- List the INTERCOM ID's for all valid users of project SHIP10 (1)
- List the INTERCOM ID's and procedure and file access keys for all users of SHIP10 (2)
- Display a particular user's keys (3)
- Modify a user's key(s) (4)
- Add a new user (5)
- Delete a user (6)

(5) The user desires a list of all valid users.

(6) INTERCOM ID's for four workers are printed.

(7) The user desires a list of all valid users and keys.

(8) INTERCOM ID's and keys are printed. Note the designations CAK and FAK. The CAL/CAK (command access lock and key) mechanism is used by the COMRADE Executive System to grant access to commands that initiate design procedures. The FAL/FAK (file access lock/key) mechanism is used to grant access to CPFMS files and was described earlier in the section entitled "File Access Control."

(9) The user now wishes to modify a user's key(s).

(10) The user indicates that CAXXGEORGE's keys are to be changed.

(11) CAXXGEORGE's old keys are printed.

(12) The user provides a new set of keys, in octal format, effectively modifying CAXXGEORGE's access privileges for SHIP10 procedures and files. Note the bit patterns of the old and new FAK's, particularly those bit positions 59 through 40 which delineate the purge sub-FAK's. (For a detailed discussion of a FAK and its sub-FAKs, see the section entitled "File Access Control.") Assuming that each SHIP10 file has a non-zero FAL (lock), CAXXGEORGE will now be able to purge files whereas before he had no purge privileges.

(13) The user now decides to delete an entry.

(14) As before, the entry is identified via its INTERCOM ID; CAXXSALLY henceforth is not permitted to work on the SHIP10 design.

(15) and (16) The user verifies that CAXXSALLY has been deleted.

(17) The user terminates MODUSERS.

```
(1)  ????MODUSERS
(2)  MODUSERS - FOR SHIP SHIP10
     WANT A LIST OF OPTIONS?
(3)  TYPE YES OF NO - YES
     OPTION          FUNCTION
        0            END PROCEDURE
        1            LIST ALL USERS
(4)     2            LIST ALL USERS WITH CAKS + FAKS
        3            SEARCH FOR A USER + DISPLAY CAKS + FAKS
        4            SEARCH FOR A USER + MODIFY CAKS + FAKS
        5            ADD USERS
        6            DELETE USERS
(5)  ENTER OPTION - 1
          USER
     CAXXERIC
     CAXXCHARLY
(6)  CAXXSALLY
     CAXXGEORGE
(7)  ENTER OPTION - 2
       USER                CAK                        FAK
     CAXXERIC       0000000000777777777   77777777777700000000
(8)  CAXXCHARLY     7777000000000001777   00000000000011777770
     CAXXSALLY      7777000017003740000O  000002776001777400O0
     CAXXGEORGE     1111111111110007777   00000000077731777000
(9)  ENTER OPTION - 4
(10) ENTER USER NAME - CAXXGEORGE
(11)  CAXXGEORGE    1111111111110007777   00000000077731777000
(12)  NEW CAK-FAK   22277711222777112222  33366600333660055555
(13)  ENTER OPTION - 6
(14)  ENTER USER NAME - CAXXSALLY
(15)  ENTER OPTION - 1
           USER
      CAXXERIC
(16)  CAXXCHARLY
      CAXXGEORGE
(17)  ENTER OPTION - 0
      VALID USERS FILE FOR SHIP10 UPDATED
```

Figure 9 - Use of Procedure MODUSERS, Sample Terminal Session

. Command Procedure INITDES

Purpose: Initializes a design project.

Explanation: INITDES is used to establish a design project as a work unit. Generally, these functions will be performed by the INITDES procedure: 1) Creation of the valid users file required by the MODUSERS procedure, 2) Modification of the CPFMS file directory to support the naming of files for the new project; and 3) Creation of the primary design data base which is required by the new project's design procedures.

Example: Figure 10 provides an example of the use of INITDES. An interpretation of the dialogue follows:

(1) The user initiates INITDES.

(2) The user provides a name by which the new project is to be known. The name will be treated as an L3 name by the CPFMS file management system.

(3) The user indicates that SHIP10 design procedures require initialization of a COMRADE Data Management System[4,5] data base file which will serve as the repository of certain design information. By convention, the L3 and L4 names for this file will be identical. Thus, in this example the name of the design file will be COMR-SHIPSYS-SHIP10-SHIP10.

(4) The user is asked whether personnel working on other projects can save and/or update SHIP10 project files. This option, if exercised via negative responses, enables further control of the number and content of SHIP10 files by restricting file catalog and modify operations to valid SHIP10 personnel only.

(5) INITDES indicates that it successfully initialized the SHIP10 project.

(6) The procedure then encourages the user--for the moment the only valid user of SHIP10--to sign-on to the SHIP10 project and to define any additional project workers.

(7) and (8) The user signs-on as instructed.

(9) The MODUSERS procedure is executed.

```
(1)   ????INITDES
(2)   ENTER NEW DESIGN NAME (8 CHARS. MAX.) - SHIP10
      SHOULD DESIGN FILE BE INITIALIZED?
(3)   YES
(4)   IF SIGNED ON TO ANOTHER DESIGN, MAY USERS:
        * CATALOG FILES UNDER THIS DESIGN?  YES
        * MODIFY FILES UNDER THIS DESIGN?  YES
(5)   SHIP10 INITIALIZED
(6)   -TO DEFINE VALID USERS OF SHIP10
        1.  SIGN-ON TO SHIP10 BY EXECUTING PROCEDURE SHIPSYS
        2.  EXECUTE MODUSERS PROCEDURE
(7)   ????SHIPSYS
(8)   WHICH SHIP?  SHIP10
(9)   ????MODUSERS
                .
                .
                .
```

Figure 10 - Use of Procedure INITDES, Sample Terminal Session

. Command Procedure DESDONE

Purpose: Terminates a design project.

Explanation: DESDONE is used to disestablish a design project as a work unit. Generally, these functions will be executed by DESDONE: 1) Removal of those on-line files associated with the design, 2) Modification of the CPFMS directory so as to discontinue file naming support for the design; and 3) Archival of selected files onto magnetic tape.

Example: Figure 11 depicts an example of the use of DESDONE. An interpretation of the dialogue follows:

(1) The user initiates DESDONE.

(2) The user indicates that he desires a listing of those files associated with the SHIP10 project.

(3) Only the L4 names of the SHIP10 files are listed. Recall that the actual permanent file name accorded an L4 file consists of a concatenation of the L2, L3, and L4 names separated by hyphens (e.g., SHIPSYS-SHIP10-HULL).

(4) The user is notified that he must not continue unless he is certain that the project is to be terminated and that personnel are not presently using SHIP10 procedures and files.

(5) The user continues with DESDONE.

(6) He is presented a choice of options. He may

- purge all files associated with SHIP10 project (1),
- archive and purge all SHIP10 files (2),
- archive selected SHIP10 files while purging all files (3), or
- terminate DESDONE (4)

(7) The user is notified that he must provide a magnetic tape for those files which are to be archived. The slot number for this tape may have to be assigned by the DTNSRDC Computer Center staff.

(8) Having acquired the information needed to identify his tape, the user indicates that he wants to archive certain files while removing all SHIP10 files from on-line storage.

(9) Three L4 files are archived.

(10) The user identifies his magnetic tape. As with the LOGMAINT procedure, a batch job will be executed to copy the files onto the tape reel.

(11) DESDONE has successfully terminated the SHIP10 project. Subsequent work on the project will not be permitted.

```
(1)   ????DESDONE
      ENTER YES OR NO FOR LISTING OF FILE NAMES
(2)   ASSOCIATED WITH SHIP10-YES
(3)   SHIP10 L4 FILE NAMES
             HULL
             MACHY
             ELEX
             SHIP10
             PIPING
(4)   ***NOTICE***
      CONTINUE ONLY IF:
        1.  SHIP10 NO LONGER NEEDED ON-LINE
        2.  OTHERS ARE NOT LOGGED ON TO SHIP10, AND
        3   SHIP10 FILES AREN'T IN USE
(5)   ENTER HALT OR CONTINUE - CONTINUE
(6)   OPTIONS ARE:
        (1)  TERMINATE SHIP10 - PURGE ALL ASSOCIATED FILES
        (2)  TERMINATE SHIP10 - ARCHIVE AND PURGE ALL FILES
        (3)  TERMINATE SHIP10 - ARCHIVE SELECTED FILES - PURGE
               ALL FILES
        (4)  HALT PROCEDURE
(7)   ***NOTICE***
      OPTIONS (2) and (3) REQUIRE THAT YOU PROVIDE A MAGNETIC
      TAPE FOR FILE ARCHIVAL
(8)   ENTER OPTION - 3
(9)   ENTER (A)RCHIVE OR (P)URGE AFTER FILE NAME
      HULL - P
      MACHY - A
      ELEX - A
      SHIP10 - A
      PIPING - P
(10)  FOR LABELED TAPE ENTER LABEL ELSE BLANK - ARCHTAPELABEL
      FOR SLOT TAPE ENTER SLOT NUMBER ELSE BLANK - 24
      ENTER TAPE VISUAL REEL NUMBER - TAPE06
      ENTER YOUR ORGANIZATIONAL CODE - NAVSEC6102
(11)  SHIP10 TERMINATED
```

Figure 11 - Use of Procedure DESDONE, Sample Terminal Session

ACKNOWLEDGMENTS

The author wishes to acknowledge Thomas R. Rhodes and William C. Gorham, Jr., of DTNSRDC, for their technical suggestions and assistance in preparing this report.

APPENDIX A

AUXILIARY DESIGN ADMINISTRATION SUBROUTINES

The COMRADE Design Administration System includes a set of auxiliary subroutines which have been found to be quite useful to DTNSRDC Control Data 6700 programmers. The subroutines allow an application program to

- submit a file to the central site input queue for batch processing (BCHIN);
- declare the properties of a file and request assignment of a physical unit (REQUEST);
- unload a file, effectively disassociating the file from the job (CLUNLD);
- rewind a file (CLRWND);
- make permanent an appendage to a file (EXTEND); and
- determine the permissions (e.g., CONTROL, MODIFY) associated with a logical file which exists at the program's control point (PERM).

Individual descriptions of these subroutines follow.

1. BCHIN

BCHIN (LFN, IDISP, IERR, NEWLFN)

BCHIN submits a card image file to the central site input queue for batch processing. The file must begin with a control card record, must reside on an allocatable device, and must not be a permanent file. If written by FORTRAN I/O, the file should be rewound before the call to BCHIN is made.

Parameters:

LFN	Logical file name, seven characters or less, left-justified, zero- or blank-filled
IDISP	Location to which the output of the submitted job is to be sent: = 1 Central site printer = XX Two-character (right-justified) INTERCOM terminal identifier
IERR	Error code: = 0 Submitted successful = -1 Error in job statement of control card record > 0 Unable to send output
NEWLFN	Location into which the name (seven characters, left-justified) of the submitted job is to be placed.

Example:

To obtain an audit of all files assigned to the user CASM, one might code as follows:

```
      PROGRAM CALBCH (OUTPUT, TAPE3)
C     GENERATE CONTROL CARD RECORD ON TAPE3
      WRITE (3, 20)
   20 FORMAT (*COMR, T10, P4.    1832,SMITH *,/
     1*CHARGE, CASM, 0123456789.*,/
     2*AUDIT (ID=CASM)*)
      REWIND 3
C     SUBMIT TAPE 3 TO INPUT QUEUE - PRINT AT
C     CENTRAL SITE
      CALL BCHIN (5HTAPE3, 1, IERR, NEWLFN)
      PRINT 30, IERR, NEWLFN
   30 FORMAT (I5, 2X, A7)
      STOP
      END
```

2. REQUEST

REQUEST (IERR, LFN, IPF)

REQUEST assigns a particular equipment to a file, performing the function of the SCOPE control card REQUEST.

Parameters:

IERR	Error code:
	= 0 Request successful
	= 22 No equipment available
	= 24 File already assigned
LFN	Logical file name, seven characters or less, left-justified, zero- or blank-filled; or a two-word array formatted as required for the REQUEST macro.[8]
IPF	Indicates the format of the argument LFN:
	≠ 0 LFN contains the name of a file which will be assigned to a permanent file device
	= 0 LFN is a two-word REQUEST macro array

3. CLUNLD

CLUNLD (IERR, ICLT, LFN)

CLUNLD disassociates a file from an executing program, performing the function of the SCOPE control card UNLOAD.

Parameters:

IERR	Error code:
	= 0 Unload successful
	= 1 Unknown logical file

ICLT — Indicates the format of the argument LFN:

= 1 LFN contains the address of the File Environment Table (FET) for the file that is to be closed and unloaded

= 2 LFN contains the logical file name, seven characters or less, left-justified, zero- or blank-filled

= 3 LFN contains the logical file name or file number of the FORTRAN file that is to be closed and unloaded

LFN — Logical file name or number as described for ICLT

4. CLRWND

CLRWND (IERR, ICLT, LFN)

CLRWND rewinds a file, performing the function of the SCOPE control card UNLOAD.

Parameters:

IERR — Error code:

= 0 Rewind successful

= 1 Unknown logical file

ICLT — Indicates the format of LFN:

= 1 LFN contains the address of the File Environment Table (FET) for the file that is to be closed and unloaded

= 2 LFN contains the logical file name, seven characters or less, left-justified, zero- or blank-filled

= 3 LFN contains the logical file name or file number of the FORTRAN file that is to be closed and rewound

LFN — Logical file name or number as described for ICLT

5. EXTEND

EXTEND (IERR, LFN)

EXTEND makes permanent an appendage to a file, performing the function of the SCOPE control card EXTEND.

Parameters:

IERR — Error code:

= 0 Extend successful

= 3 Unknown LFN

= 5 SCOPE directory is full

= 15 Operation attempted on a non-permanent file

LFN — Logical file name, seven characters or less, left-justified, zero- or blank-filled

6. PERM

PERM (IPERM, LFN)

PERM determines the permissions associated with a file.

Parameters:

IPERM — Permission indicator:

= 0 Unknown LFN

> 0 A five-bit code indicating the permission status of LFN. Possible codes are:

(In Bits 0-3)

1000	CONTROL
0100	MODIFY
0010	EXTEND
0001	READ

(In Bit 4)

1	Non-Permanent file
0	Permanent file

LFN — Logical file name, seven characters or less, left-justified, zero- or blank-filled

APPENDIX B

OPERATING PROCEDURES FOR DESIGN ADMINISTRATION SUBROUTINES

A. SUBROUTINE ACCESS

The Design Administration System subroutines reside on a program library created by way of the SCOPE EDITLIB capability.[8] This library, a SCOPE permanent file on DTNSRDC's Control Data 6700 computer, is named SUBLIB, ID=COMR. All users are automatically assigned "read only" permission when they access SUBLIB for use in constructing their application programs.

The control statements needed to access the Design Administration subroutines for the purpose of creating an application program are:

```
ATTACH, SUBLIB, ID=COMR
LIBRARY, SUBLIB
```

Thus, given that a FORTRAN program exists on a local file named HULLDEF, control statements that follow would (1) compile the source code for HULLDEF, (2) load the object code including the DA subroutines, and (3) then execute the program.

```
FTN, I=HULLDEF
ATTACH, SUBLIB, ID=COMR
LIBRARY, SUBLIB
LGO
```

B. CORE REQUIREMENTS

Of the approximately 250 subroutines on SUBLIB, 60-70 are user-callable. Since these subroutines are not mutually exclusive (that is, subroutines A, B, and C might each call subroutine X, but subroutine A might also call subroutine Y; subroutine B might also call subroutine Z; and subroutine C

might call Y and Z), computation of the core requirements of a particular application program is difficult. One way of estimating these core requirements is by constructing a dummy program that calls the needed SUBLIB subroutines, and then allowing the system loader to compute the storage needs. Thus a user desiring to estimate the core required for subroutines REQUEST and CATLOG might code as follows:

```
PROGRAM TEST
CALL REQUEST
CALL CATLOG
END
```

The control statements needed to effect loading of such a program are

```
FTN
MAP, ON
ATTACH, SUBLIB, ID=COMR
LIBRARY, SUBLIB
LOAD, LGO
NOGO
```

Following inspection of the system loader's computations, the user could determine the need for a sophisticated program structuring facility such as overlay,[8] segmentation,[8] or the COMRADE Absolute Subroutine Utility (see Appendix C).

APPENDIX C

USE OF THE COMRADE ABSOLUTE SUBROUTINE LIBRARY

A. SUBROUTINE ACCESS

The COMRADE Absolute Subroutine Utility (ASU)[9] provides a capability for loading and executing the Design Administration subroutines, as well as those auxiliary subroutines identified in Appendix A. Two ASU subroutines--INITLDR and COMRLDR--are used to load and execute DA subroutines from an Absolute Subroutine Library (ASL). Use of an ASL offers the following benefits:

- The amount of core required to run an application program can be reduced, since the DA subroutines (and the auxiliary software) can execute within the same core area.
- Only one copy, the latest, of the DA subroutines is needed, since the subroutines reside on an ASL that is designed to be shared by many application programs.
- Since ASL subroutines are separate from an application program, subroutine errors can be corrected without users being required to recreate their programs.
- Parameters may be passed to and among ASL subroutines, either as arguments in the CALL and COMRLDR or through a shared common block.
- No instruction modification is needed since ASL subroutines are in absolute form.

---

[9]Wallace, M., "COMRADE Absolute Subroutine Utility Users Manual," DTNSRDC Report 76-0004 (Jan 1976).

Two permanent files are required when the COMRADE ASL subroutines are to be used. The files are named ASL, ID=COMR and SUBLIB, ID=COMR. The file named ASL contains COMRADE subroutines in absolute form and must be attached whenever an application program is to be executed. The file named SUBLIB (discussed in Appendix B) contains the subroutines INITLDR and COMRLDR. Descriptions of INITLDR and COMRLDR follow.

1. INITLDR

INITLDR (LFN, NAME, IERR)

INITLDR identifies the ASL file that contains the COMRADE subroutines in absolute form. INITLDR should be called only once in an application program, before the first CALL to COMRLDR.

Parameters:

LFN	Logical file name (left-justified, H-format) of the ASL file.
NAME	Reprieve indicator: = 0 No reprieve subroutine is to be executed = REFILE (left-justified, H format) A reprieve subroutine (developed by COMRADE system personnel) named REFILE will be executed following an abnormal error condition; REFILE closes and finalizes the COMRADE Data Management System[4,5] data base file active at the time of the error. If a user-developed subroutine is to be executed, the

name of that subroutine, instead of REFILE, must appear in NAME.

IERR — Error code:
= 0 No error detected
≠ 0 File not in ASL format.

2. COMRLDR

COMRLDR(SUB, PARM1, PARM2,..., PARMN)

COMRLDR loads and executes a specific ASL subroutine. If the subroutine is already in memory, execution will proceed directly. If it is not resident on the current ASL, the message INVALID SUBROUTINE NAME will be written to the dayfile and the program will be aborted. If subroutine INITLDR has not been called, the message INITLDR NOT CALLED will be written to the dayfile and the program aborted.

Parameters:

SUB — The name (left-justified, H format) of the ASL subroutine that is to be executed.

PARM1-PARMN — The parameters to be passed to the subroutine identified by SUB; the parameter list is variable-length.

Example:

```
CALL COMRLDR (6HCOMATC, IERR, LFN, L3, L4, IPRM, IPSPW)
CALL COMRLDR (6HEXTEND, IERR, LFN)
```

B. CORE REQUIREMENTS

Before ASL subroutines can be used, a core area in which the subroutines will execute must have been defined. This area must coincide with the area in which the subroutines have been designed to execute, otherwise an error will occur. The COMRADE ASL subroutines have been designed to execute in an area that begins at location 101 (octal) and will require a maximum of 20K (octal) words. A smaller area may be provided, depending on the specific subroutine to be called. Once the size of the area has been determined, it can be reserved by way of a FORTRAN labelled common statement placed immediately after the PROGRAM statement. The statement may appear as

COMMON/AREA/DUM (20000B)

A list of subroutines resident on the file ASL, ID=COMR follows. The core requirement for each is indicated.

COMRADE ASL SUBROUTINES

NAME	LENGTH (Octal)	NAME	LENGTH (Octal)
ADDO	17101	CATLOG	7315
ADDU	16770	CDA	17204
ATTACH	7315	CHANGE	17725
BLKLEN	7276	CHANGO	17375
BRKUP	12010	CIA	17277
CAD	17131	CLUNLD	11516
CAI	17076	CLRWND	11516
CAO	17070	COA	17277
CAR	17114	COMATC	15271

NAME	LENGTH(Octal)	NAME	LENGTH(Octal)
COMCAT	15540	LOCATE	6322
COMPRG	15320	LSHIFT	6213
COMUNL	13535	MODBN	15722
CRA	17202	MOVE	6276
CULL	13727	NOCC	12266
DEFFIL	7134	PACK	6250
DEFINB	16550	PASOVR	7756
DELETB	17261	PERM	7120
DELETE	17433	PFNIN	6644
DISPOS	15411	PURGE	7315
EMPTY	6312	QUERY	17524
EXTEND	7120	REFILE	12304
FETCH	13341	RENAME	7315
FETCHC	12633	REQUEST	6422
FETCHN	12515	RPK	6330
FETCHR	13503	RWE	6203
FLFN	15155	SBUFF	6747
FTNBIN	12355	SETCH	6241
GET	10221	SETCOM	7107
GETCOM	7107	UNPACK	6213
ILFN	10713	ZADDCOM	6711
INPFN	6254	ZGENCOM	6711

APPENDIX D

ESTABLISHING DESIGN ADMINISTRATION CAPABILITIES WITHIN A DESIGN SYSTEM

The steps for developing a "skeleton" COMRADE application system--a system which has no inherent capabilities other than the ability to grow, that is, to have command procedures established within it--are described in Tinker.[3] Included among the steps is creation of an application system sign-on procedure which must (1) sign off from any system currently in control, (2) validate system users, (3) make available certain system files, and (4) assign values to certain locations of the COMRADE Executive's System Common.[3] Described here are the steps which must be performed in order to establish COMRADE Design Administration capabilities (i.e., the CPFMS system, the logging and reporting procedures, etc.) for use within a particular COMRADE design system. The steps include those discussed in Tinker.[3] Generally, they will be accomplished with the assistance of COMRADE System personnel.

Step 1: Create the Design System Event Log

The logging and reporting mechanism of the COMRADE Design Administration System allows managers of large computer-aided design efforts to monitor design activity. Fundamental to this mechanism is a SCOPE permanent file which contains the logged entries for events performed in a given design system. An event log is created and named through a job which attaches and copies a prototype file named LOGFILSKELETON. The copy may

then be cataloged as a user-named event log, and modified to include such information as the "high type to log" or HTL value, the file creation date.

Step 2: Establish the Design System Within CPFMS

The file directory for the COMRADE Permanent File Management System (CPFMS) is a set of password-protected files hierarchially organized into four conceptual levels. In order to create a new "branch" in the tree-structured CPFMS directory--thereby establishing the design system within CPFMS--four files must be linked together. The files are: the Level One (L1) file named COMR; an L2 file having a permanent file name that is identical to the name of the new design system; an L3 file having a name formed by concatenating the system name with the name of a project used mainly for executing administrative functions and for monitoring design activity; and an L4 file having a name formed by concatenating the names of the design system, the administrative project, and the list of those personnel authorized to execute administrative functions. The L2, L3, and L4 files are created and named through a batch job that generates copies of the prototype files L2SKELETON, L3SKELETON, and L4SKELETON. The copies may then be cataloged and modified to include information which links the three files together. After this linkage is developed, the file COMR must be linked to the L2 file.

Step 3: Create the Design System's FEF and PCB Files

The File Environment File (FEF) and Procedure Control Block (PCB) file contain information about those computer programs and design procedures which constitute a COMRADE design system. A FEF is created by copying a prototype file named COMRFEF-SKELETON. A PCB file is created by initiating procedure PDL and by defining a dummy procedure (consisting, perhaps, of only a STOP control block). Tinker[3] describes the creation of the FEF and PCB files in greater detail.

Step 4: Add File Information for the FEF and PCB Files to the GENERAL FEF

File information identifying the FEF and PCB files must be added to the GENERAL FEF with procedure UPFEF. This step is also described in greater detail in Tinker.[3]

Step 5: Create the Design System's Sign-On Procedure

Access to a COMRADE design system's procedures is usually controlled by way of a system sign-on procedure which (1) signs off from any system currently in control, (2) prepares certain files, (3) validates users, and (4) communicates certain values to the COMRADE Executive. Each of these functions is described in the following paragraphs in greater detail. A sign-on procedure is created and named by the design system developer, but is added to the GENERAL PCB file by COMRADE personnel. A sign-on procedure is invoked whenever the user wishes to begin a design session or switch designs. After the user has finished his design session, he terminates operations by issuing the procedure

SIGNOFF. This action causes the COMRADE Executive to execute the sign-off program.

. Signing Off the Current System

A sign-on procedure signs off any system currently in control by executing the sign-off program whose name appears in System Common. The sign-off program generally will process the system event log, will unload all CPFMS files as well as the Subsystem Common file, and will disconnect the FEF and PCB files.

. Preparing the FEF, PCB, and Subsystem Common Files

A second major function of the sign-on procedure is to make ready the FEF, PCB, and Subsystem Common files. The FEF and PCB files are readied by attaching, copying, and then releasing the original FEF and PCB files; the copies are then used to execute those design procedures accessible to the authorized design system user. The Subsystem Common file is normally generated and initialized by the COMRADE Executive resident utility routine CGENCM.

. Validating Users

A sign-on procedure must validate the potential user of the new system. This function is performed by a program which embodies those techniques and conventions designed by the system developer as necessary to determine that the user may or may not use the system. Generally, the validation process will be accomplished by attaching, reading, and unloading certain CPFMS files (such as the L2 file to make certain that the requested

system project has been initialized, and the L4 file which lists those personnel authorized to work on the requested project), by verifying that the potential user is mentioned in the file of valid project users, and by readying the system event log.

. Communicating Values to the Executive

If the user is not valid for the new system, the sign-on procedure will probably terminate. If the user is valid for the requested system, his procedure access key must be set into the COMRADE Executive's System Common region. Several other values must also be set into predefined locations of System Common, such as the name of the new design system, and the name and ID of the permanent file containing the system's sign-off program. The user's file access key, required by CPFMS, must be set into a predefined location of Subsystem Common.

Step 6: Customize the Existing DA Procedure Library

COMRADE Design Administration (DA) capabilities (i.e., CPFMS, REPORT, INITDES, etc.) have been successfully established and used within the Navy's Integrated Ship Design System (ISDS).[7] In order to expedite development of the ISDS DA procedures, a DA library, from which procedures could be conveniently retrieved and customized to meet a particular design system's needs, was constructed and made available to system developers. In general, a developer will retrieve the desired procedures from the DA library (a SCOPE permanent file

named DAPROCPL), will customize the procedure definition language statements to fit his system's specifications, and will then catalog the resultant statements in preparation for their translation into procedure control blocks.

Step 7: Establish DA Procedures Within the Design System

The procedure PDL is used to create and/or "compile" a procedure definition language program and to add the resultant procedure control blocks to the design system's PCB file. For each customized DA procedure, the system developer will execute the PDL procedure, but only after he has initially signed on to the design system's administative project (thereby setting up the required CPFMS, FEF, and PCB files, etc.). Upon successful completion of a DA procedure's "compilation," the user will be asked to supply a name by which the procedure will be known. The user will respond with a name such as LOGMAINT, INITDES, etc. The user will then be asked to supply an access lock for the procedure. Normally, this will be entered as a 20-digit octal character string that will be used to confine use of the DA procedures to authorized personnel, generally, those valid users of the system's administrative project.

REFERENCES

1. Rhodes, T. and W. Gorham, "COMRADE - The Computer-Aided Design Environment Project - An Introduction," DTNSRDC Report 76-0001 (Jan 1976).

2. Brainin, J., "Use of COMRADE in Engineering Design," paper presented at the American Federation of Information Processing Societies National Computer Conference, New York (Jun 1973).

3. Tinker, R. and I.L. Avrunin, "COMRADE Executive System Users Manual," DTNSRDC Report 76-0002 (Jan 1976).

4. Gorham, W. et al., "COMRADE Data Management System Host Language Interface Users Manual," DTNSRDC Report 76-0006 (Jan 1976).

5. Gorham, W. et al., "COMRADE Data Management System Conversational Interface Users Manual," DTNSRDC Report 76-0007 (Jan 1976).

6. Chernick, C.M., "COMRADE Design Administration System," paper presented at the American Federation of Information Processing Societies National Computer Conference, New York (Jun 1973).

7. Brainin, J., "Functional Description for the Integrated Ship Design System (ISDS)," DTNSRDC Report 4663 (Apr 1975).

8. Control Data Corporation, "SCOPE Reference Manual," 6000 Version 3.4, Publication No. 60307200.

9. Wallace, M., "COMRADE Absolute Subroutine Utility Users Manual," DTNSRDC Report 76-0004 (Jan 1976).

INITIAL DISTRIBUTION

Copies	
1	U.S. Army Electronics Command/AMSEL-GG-CG/ Robert Dunn
1	CHONR 437/Denicoff
1	NAVSEA 6E24/Chaiken
25	NAVSEC
	1 SEC 6105B/ Dietrich
	20 SEC 6105B/Berry
	1 SEC 6114B2/Fuller
	1 SEC 6133E/ Straubinger
	1 SEC 6152D/Lee
	1 SEC 6179A20/Singer
12	DDC
1	NASA-Langley/R. Fulton
1	Gruman Aerospace Corp./ William H. Mueller

Copies	Code	Name
1	185/Corin	
1	1851/Brainin	
1	1853/Thomson	
1	1854/Lynch	
1	1855/Brengs	
1	1856/Skall	
1	189/Gray	
1	1892.1/Strickland	
10	5211.1 Reports Distribution	
1	522.1 Library (C)	
1	522.2 Library (A)	
1	608/Pierce	

CENTER DISTRIBUTION

Copies	Code	Name
1	18/1808/Gleissner	
1	1802.2/Frankiel	
1	1802.3/Theilheimer	
1	1805/Cuthill	
2	1809.3/Harris	
1	182/Camara	
1	1822/Rhodes	
1	1824/Berkowitz	
1	1826/Culpepper	
20	1828/Gorham	
5	1828/Chernick	
1	184/Lugt	

