NASA-TM-100576 19880014801

NASA Technical Memorandum 100576

THE ENVIRONMENT FOR APPLICATION SOFTWARE INTEGRATION AND EXECUTION (EASIE) VERSION 1.0

VOLUME IV

SYSTEM INSTALLATION AND MAINTENANCE GUIDE

DONALD P. RANDALL
KENNIE H. JONES
LAWRENCE F. ROWELL

April 1988

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665-5225

PREFACE

The Environment for Application Software Integration and Execution, EASIE, provides both a methodology and a set of software utility programs to ease the task of coordinating engineering design and analysis codes. The need for such techniques and tools has stemmed from the computer-aided design and engineering activities within the Space Systems Division (SSD) at the Langley Research Center. In SSD, the Vehicle Analysis Branch (VAB), with emphasis on advanced transportation systems, and the Spacecraft Analysis Branch (SAB), with emphasis on advanced spacecraft, share a common need to integrate many stand-alone engineering analysis programs into coordinated, quick-turnaround, user-friendly design systems. In particular, the most needed capabilities include easy selection of application programs, quick review and modification of program input/output data, and logging of the actual steps that were executed during the study. Although the application programs used by VAB and SAB differ, the design methods used by their engineers are quite similar, and great efficiencies can be gained by providing a computer "environment" that provides the capabilities mentioned above.

EASIE is both a user interface and a set of utility programs which support rapid integration and execution of programs about a central relational database. In general, the EASIE system addresses the needs of four different classes of people who will

be involved in the buildup of an engineering design system. Certain individuals may serve in more than one of these roles, but the following terms will help to clarify several distinct activities associated with the EASIE system.

The first classification represents the engineer/designer/analyst. This group conducts the design study through the execution of modeling and analysis programs and the generation of data required to evaluate the design against its objectives. EASIE documentation will refer to this group as "EASIE system users" or, more often, as "users." In general, these users are only interested in executing programs already installed into an EASIE design system.

A second group aided by EASIE will be referred to as "application programmers." These programmers/engineers are responsible for the development and improvement of modeling and analysis programs used in the engineering design process. They are the experts with respect to particular application programs and can define the input and output variables. This definition must be done before inclusion of a program with others in the integrated system.

The third group can be referred to as "program implementers," since their function is to provide an environment where all the software tools work together with a minimum of effort. These people will use information provided by the application programmers and will install or modify the programs in an EASIE system by creating appropriate data constructs in the database and locating files where needed by the EASIE executive.

The fourth classification is that of "design team leader" or "design manager." This is the individual or group responsible for identifying parameters important to the design study and for configuration management of the data as it is produced by the design team. This design manager must have an overview of the total data requirements for the analysis process and must be concerned foremost with the integrity of the data.

With these terms defined, the four volumes of EASIE documentation can be associated with the groups most likely to use them. Each of the volumes addresses different aspects of the support tools, and each is intended to be usable independent of the others.

Volume I, Executive Overview, provides information about the functions, concepts, and historical development of EASIE and should be read by anyone trying to determine if EASIE would be beneficial to his work.

Volume II, Program Integration Guide, describes the portion of the EASIE tools supporting both the integration of application programs into a central database and the definition of the data dictionary used during data review and modification. This volume will be used primarily by the "program implementer" and the "design manager" in their responsibilities for the actual installation of appropriate programs into a fully integrated design system. However, the "application programmer" may also use tools described in this volume to assist in the documentation of input/output variables for the application program.

Volume III, Program Execution Guide, describes the portion of the EASIE tools supporting the selection and execution of application programs, building of menus, and editing of program data. This volume will be of foremost importance to the "users" who will perform design studies. In addition, the "program implementers" will find the sections concerning the construction of application-dependent procedures helpful. Finally, this document will also be used by the "design manager" for reviewing data and design activities.

Volume IV, System Installation and Maintenance Guide, describes the procedure of loading the EASIE system onto a computer. It also gives some insight into the hardware and software dependencies of the EASIE code. This, most likely, will be needed by the "program implementer" to familiarize himself with the directory structure and location of the various EASIE components. Although the design of EASIE is intended to reduce the system dependencies, this version nevertheless reflects in several ways the current implementation using the Relational Information Management (RIM*) database management system and the VAX/VMS+ operating system.

*Trademark of Boeing Computer Services

+Trademark of the Digital Equipment Corporation

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 INTRODUCTION

The Environment for Application Software Integration and Execution (EASIE) tools include a program executive and a collection of data management utilities designed to provide a consistent environment for the integration and execution of application software. The purpose of this document is to supply the programmer charged with implementing or maintaining the EASIE software with the required information for performing this task. This document may also provide additional insight to an engineer/designer/analyst wishing more detailed understanding of the organization of the EASIE software tools. When additional information on the utilization of the EASIE software tools is required, the other three volumes of EASIE documentation [1, 2, and 3] in this series should be consulted.

The computing environment for developing the EASIE software tools was a DEC VAX 11/785 running VMS 4.4. When this software is being installed, it is very helpful to be familiar with VAX/VMS utilities and other VAX/VMS features such as the hierarchial file structure, file descriptions and extensions, user-defined symbols and logicals, and command procedures. The EASIE software is written entirely in DEC FORTRAN 77, using standard VAX/VMS compile, load, and execute processes including the use of the DEC FORTRAN "INCLUDE" files and the creation and use of object libraries.

In addition to functioning as an EASIE software installation guide, this document endeavors to address programming issues not

addressed in the other EASIE documentation [1, 2, and 3] and is organized into several sections relating to different aspects of the EASIE software utilities. Section 2 details the hierarchical file organization employed to house the EASIE tools. Section 3 identifies alternatives for setting up an application programming environment utilizing the sample problems contained in Volumes II and III. Section 4 provides more detail on the modules that comprise the individual EASIE software components. Section 5 is an explanation of the task of installing EASIE from the supplied installation package. Section 6 provides the information for instructing the engineer/designer/analyst for making additions to the EASIE system.

This document can stand alone in its description of the EASIE software installation (section 5); however, some prior knowledge of the contents of Volumes II and III will be helpful throughout the other sections, since terms and concepts from the other EASIE volumes are used here with little or no explanation.

2.0 EASIE HIERARCHY

The organization of the EASIE software takes advantage of the VAX/VMS hierarchial file structure. Although each installment has flexibility in tailoring the placement of both the EASIE tools and any application programs to be associated with the system, the organization shown in Figure 1 is recommended. The examples used to illustrate the integration and execution environments described in Volumes II and III are presented in this figure and referenced throughout this document. In describing the various levels of the EASIE hierarchy, familiarity with standard VAX/VMS file specifications is assumed.

The top level of the hierarchy is the AIDE (a former alias for EASIE) directory. Beneath this directory are the major subdirectories that comprise EASIE. The BUILD_DICT subdirectory contains the utilities that are used in constructing the data dictionary and the input/output templates within an associated data dictionary database. The REVIEWER utility for editing program input and output resides in the REVIEW subdirectory. The EASIE executive software resides in the PROG subdirectory. The PROC, HELP, and DB subdirectories are repositories for files needed while running the executive. The PROC subdirectory contains DCL command files referenced from within the EASIE executive. The text files within the HELP subdirectory are accessed while exercising the on-line HELP facility available in the executive. Finally, the DB subdirectory includes text files containing the major executive command menus. This subdirectory

also contains the list of master workspaces [3] accessible in this Version 1.0 of EASIE. The major files in each of these subdirectories are briefly described in Section 4 of this document.

The remaining three subdirectories, EXAMPLE, EXMENU, and EXSIMPLE are included to illustrate the alternative methods for integrating application programs into this hierarchy. The purpose and contents of these three subdirectories are discussed in Section 3.

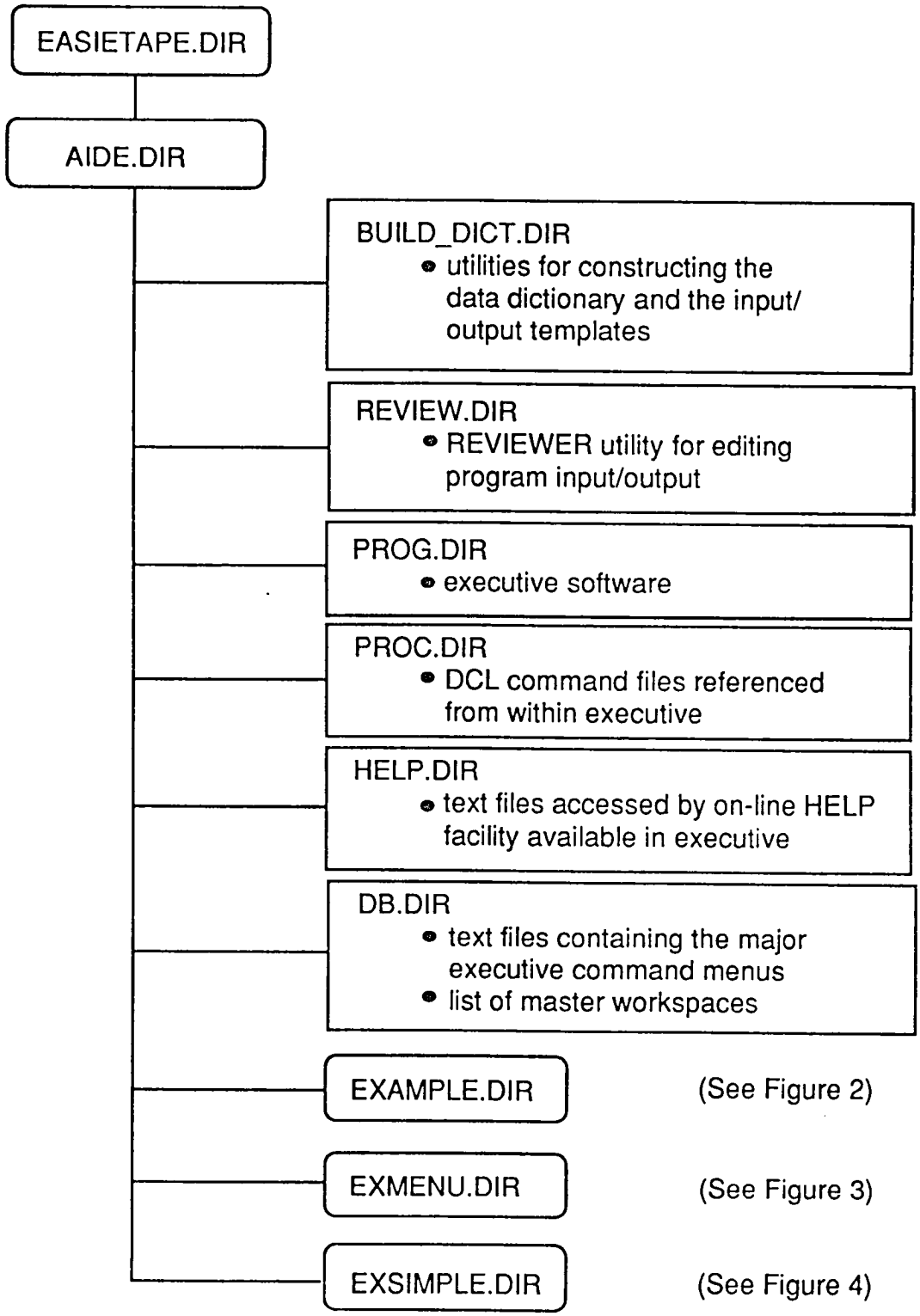

Figure 1. - EASIE Software File Hierarchy

3.0 EASIE PROGRAM INTEGRATION

The techniques for integrating systems of application programs into EASIE are detailed in the Program Integration Guide [2]. The intent of this section is to present several approaches for organizing the application-program-dependent files into a structure compatible with EASIE. The subdirectories EXAMPLE, EXMENU, and EXSIMPLE, shown in Figure 1, will serve as illustrations for various ways to organize files and set up the communication link between the user, EASIE, and the application program areas. The mechanism for specifying the necessary data path is the EASIE workspace [3]. The utilization of the workspace as related to the above three illustrative examples is discussed in this section. Finally, an automated tool, in the form of a VAX/VMS command file, for constructing an EASIE subdirectory structure is detailed.

The EASIE tools key on the VAX/VMS file extensions when performing specified operations. These files come in pairs, an object file and an associated description file. For example, a workspace file, a configuration database file, and an application executable file must all be accompanied by a corresponding description file. A complete list of the relevant EASIE file extensions is contained in APPENDIX A. For the purpose of identifying categories of EASIE files, this document employs the file extension and the VAX/VMS "wildcard" character asterisk ("*"). The "*" is used to designate one or more application program names, configuration names, or other user-supplied names.

The recommended approach for integrating a suite of application programs into EASIE is illustrated in the EXAMPLE subdirectory shown in Figure 2. An examination of the files in this directory and its subordinate subdirectories shows that the EASIE data management utilities [2] have been run and that the necessary preparations for interfacing with the executive [3] have been performed. The required files have been organized into four subdirectories: DICTNRY, SOURCE, PROG, and CFG.

The DICTNRY subdirectory contains the data dictionary database (in the form of the three RIM files, DATADB*.DAT, where "*" is 1, 2, or 3).

The source code, object code, command files, and other utilities for compiling and linking the application programs reside in the subdirectory SOURCE. It should be noted that the database interface routines generated by the EASIE MAKDICT utility [2] are also stored in the subdirectory SOURCE. The EASIE code generator named these routines *IN.FOR for input and *OUT.FOR for output together with the DEC FORTRAN "INCLUDE" files *IN.COMMON and *OUT.COMMON where "*" represents the application program name.

The application program executables *.EXE and the EASIE REVIEWER [2] files *IN.REV or *OUT.REV reside in the PROG subdirectory. The required description files [3] for the applications program *.APPLD and the REVIEWER files *.TPLD also exist in the PROG subdirectory.

The CFG subdirectory contains the "master" configurations *.DIR that may be copied to a user's area for execution. Each master configuration is described by an *.CFGD description

file. The remaining file in the CFG subdirectory is the workspace [3] which points to the master configurations and application programs.

The file PROGRAM.PGD in the PROG subdirectory is required by the EASIE executive when executing user application programs. The file is created by the SYSTEM LIBRARY PROCESSOR in producing the PROGRAM LIBRARY [2]. The EASIE executive needs this file to identify the method and/or steps necessary in exercising an application program.

The final step in preparing an EASIE execution environment is to update the master workspace file, MASTER_WS.DAT, which resides in the [.AIDE.DB] directory (figure 1). In this instance, the entry of EXAMPLE for USER___ID is added to the MASTER_WS.DAT file, in order to identify EXAMPLE as a legal EASIE ID [3] and to locate the corresponding EXAMPLE.WS workspace. With the organization and specification of the execution environment now complete, an EASIE session using EXAMPLE as the USER_ID: will initiate EASIE execution from a working area, copy a master configuration (from the [.AIDE.EXAMPLE.CFG] subdirectory), and begin issuing EASIE commands.

An alternate method for structuring the integration of a suite of application programs into an EASIE execution environment is shown in the EXMENU directory listed in figure 3.
In this organization, only a single subdirectory named CFG is required. The EXMENU approach utilizes the EASIE executive facilities for procedures and menus. The subdirectory [.EXMENU.CFG] contains the required procedures *.PROC, procedure descriptions *.PROCD, and the menu files *.PROC__*. It is

unnecessary to locate application programs in source or executable format and configurations in this area because the actual location of these files is specified in the workspace, EXMENU.WS. In essence, the EXMENU ID appears to be used to log into a separate design system. The EXMENU ID only provides menus and pointers to other programs contained in other ID areas. As in the previous paragraph on the ID EXAMPLE, the final step is to update the master workspace file, MASTER_WS.DAT, to reflect the new execution environment. An EASIE session using EXMENU as the USER_ID: will run EASIE from a working area choosing EXMENU as the desired design system.

A partial implementation of another method for structuring the integration of a suite of application programs into an EASIE execution environment is described in the EXSIMPLE subdirectory shown in Figure 4. The files in this area represent the results from exercising a simple integration example of EASIE Volume II Program Integration Guide [2]. There are no subdirectories under EXSIMPLE, implying that all programs, configurations, and REVIEWER files reside in this single subdirectory. The responsibility for creating the required description files for the programs, configurations, and REVIEWER files is assigned by the design manager at each installation. The design manager must also create the DATADB*.DAT database files. When these functions are completed, the EXSIMLE.DIR files may be copied to another work area. In this manner, the engineer/designer/analyst may exercise EASIE in a self-contained environment. This alternative method for structuring the EASIE execution environment is not

suited to multiple user environments or multiple configurations. When exercising EASIE in this environment, entering a carriage return [CR] in response to the "ID" prompt produces a proper workspace in the user's own area. (WARNING - The EASIE engineer/designer/analyst should be warned that using this approach, configuration directories function as both master and user configurations because the master and user areas are the same.)

The preceding paragraphs and figures may appear to depict a complex program integration and execution environment. However, the integration process is performed only once for each suite of application programs. Also, this effort is seen only when the EASIE software is installed and the details of the organization are transparent to the engineer/designer/analyst.

In an effort to ease the EASIE software installment, a VAX/VMS command file, AIDEINIT.COM, is provided to automate the creation of the EASIE setup as depicted in the EXAMPLE subdirectory. Invoking the command file AIDEINIT.COM before beginning the integration process will facilitate proper file placement. The functions of this command file include prompting for EASIE ID [3], creating directory and subdirectories as prescribed above, generating an empty data dictionary database, updating the master workspace file, and producing a workspace file to be used with this environment.

The command file is listed in APPENDIX B. VAX/VMS symbols are utilized wherever possible to eliminate the drawback of hard-wired file specifications. The designated symbols are identified and defined in Section 5 of this document.

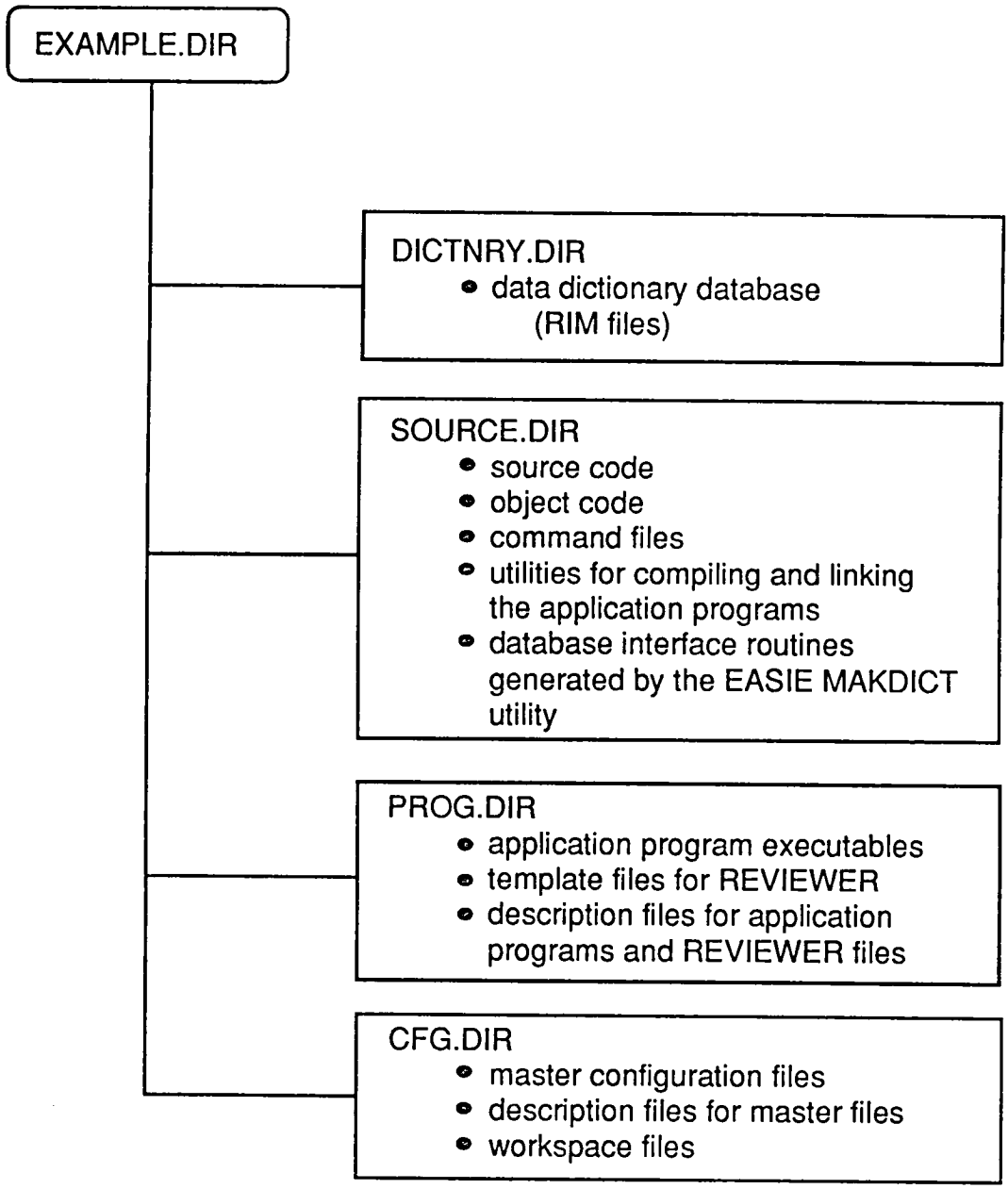

Figure 2. - Recommended Application Program File Structure

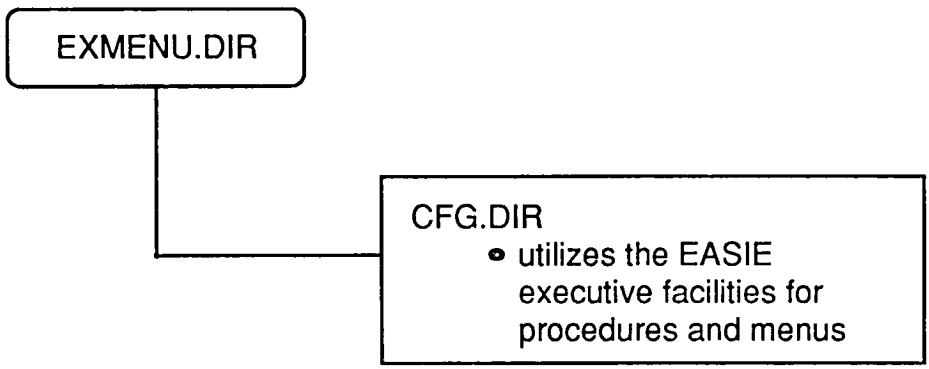

Figure 3. - Menu-Driven Application Program File Structure

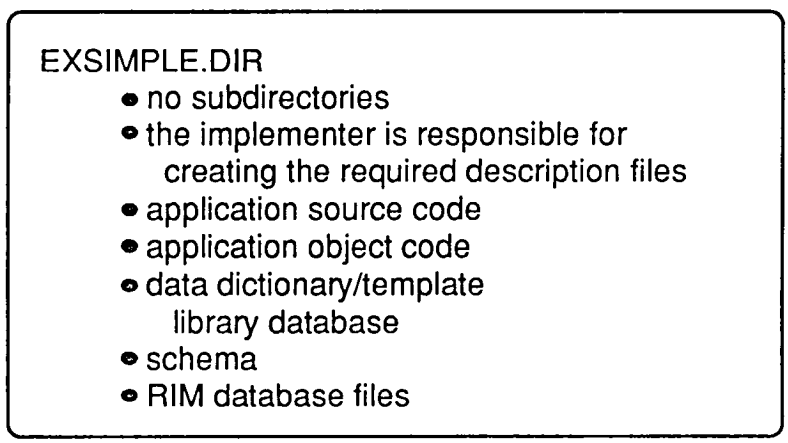

Figure 4. - Self-Contained Application Program File Structure

4.0 EASIE UTILITIES

This section presents the individual EASIE executive and data management utilities from a programmer's viewpoint. The component files are identified, and the compile/load sequences are specified. When appropriate, terminal-, DBMS-, and VAX/VMS-specific-dependencies are discussed.

4.1 EASIE Executive Initialization

The files comprising the EASIE executive reside in the [.AIDE.PROG] directory. The files in this area include the DEC FORTRAN source modules *.FOR, the FORTRAN "INCLUDE" files *.INC, the compiled object code files *.OBJ, the generated object libraries *.OLB, the compile and link command files COMPILE.COM and LINK.COM, and the executable task image MASTER.EXE.

The required FORTRAN source files are listed and briefly described in Table 1. The compilation of these modules requires no special compiler options. A sample command file for linking the EASIE executive is listed in APPENDIX C. Notice that this command file employs VAX/VMS symbols in order to eliminate hard-wired pathnames.

TABLE 1. - EASIE EXECUTIVE FORTRAN MODULES

NAME	DESCRIPTION
AIDEUTIL	Collection of utilities for performing type conversions and manipulating VAX/VMS file specifications.
BATCH	Routines for controlling the executive's background processing capabilities.
DISPLAY	Routines for listing executive menus and user text files to terminal.
EDTPL	Routines designed to permit user template editing. At the present time, these routines are incompatible with other EASIE data management utilities.
ERRORSUBS	Routines for categorizing executive errors and outputting messages to user's log and/or terminal.
FASTAIDE	Routines for initializing executive and controlling its operation.
GET	Routines for controlling executive functions normally associated with the EASIE procedure facility.
GETMENU	Routines for processing user-supplied menus associated with user-supplied procedures.
GLOBS	Routines for reading, writing, and updating EASIE workspace files.
INTERACT	Routines used for communicating with users such as issuing prompts and soliciting responses.
INTERPRET	Routines for parsing user input and validating user-supplied information.

TABLE 1. - (continued)

NAME	DESCRIPTION
MASTER	EASIE executive driver program.
PERMENU	Module of routines to control selection of executive menus and other functions such as listing configurations or applications.
PUTTRM	Program that prompts the user for the terminal type.
RESPPROCS	Module containing the major executive command routines.
RIMSUB	Routines communicating with the user's execution environment. These utilities do not communicate with resident DBMS.
SLOG	Utilities for modifying user's login environment via workspace editing.
SYSUTIL	Utility routines for interacting with VAX/VMS during executive execution.
TEMPSUB	Routines controlling such executive function as "execution" of programs or procedures or online "help".
TERMINAL	Routines for directing limited terminal independence capabilities.

4.1.1 Executive System Dependence

The selection and manipulation of EASIE executive files depend on the VAX/VMS file specification. The executive is interfaced to VAX/VMS through standard VAX/VMS system interface routines. In some cases, a VAX/VMS utility is used directly such as invoking the LIB$FIND_FILE to identify configurations, applications, workspaces, and procedures. In other instances, LIB$SPAWN is employed to execute a VAX/VMS command string for such functions as invoking the ED editor or executing an application.

In an effort to aid in the migration of the EASIE executive to another host, the VAX/VMS dependencies have been isolated to the extent possible. The vast majority of VAX/VMS dependencies are consolidated in the two modules, AIDEUTIL.FOR and SYSUTIL.FOR. There are other VAX/VMS dependencies in some of the remaining executive source modules, but these should be relatively easy to identify and correct. These dependencies usually take the form of creating a VAX/VMS string such as ED to invoke the editor or RUN/NODEBUG to execute a program. There may also be brackets from a VAX/VMS file specification or references to a VAX/VMS symbol.

Because the EASIE executive invokes standard VAX/VMS utilities, the user should not redefine any of these utilities via symbol definitions.

4.1.2 Terminal Independence

The EASIE utilities provide a very limited capability for terminal independence. The default terminal is a generic alphanumeric terminal. In its present form, terminal independence implies "erasing" the screen and text positioning for the Tektronix 401x series terminals and only clearing the screen for the Tektronix 41xx series terminals.

The terminal-independent routines reside in TERMINAL.FOR. These routines are unconditionally loaded by the executive but are linked as an object library by other EASIE utilities. For Tektronix 401x terminals, the proprietary PLOT-10 Terminal Control System (TCS) library is required. A minimal LaRC-developed Tektronix 41xx library is included as part of the installation package to support this class of terminal. To add another class of terminal, edit TERMINAL.FOR and include the device-dependent libraries for the new devices required.

The principal mechanism for communicating the terminal type throughout the software is a file named AIDExx.TRM where 'xx' is the VAX/VMS expansion of the 'TT' symbol associated with the user's job. This file may be written to the user's working area by the program PUTTRM.EXE which is executed from the procedure TOAIDE:[PROC]AIDE.COM. The executive calls a routine TINIT (within TERMINAL.FOR) to read an existing terminal file (AIDExx.TRM). Integrated programs and the REVIEWER employ this technique via the routine PINIT in file TERMINAL.FOR. An installation using the EASIE software may choose to exercise PINIT in a similar manner for any of their application programs.

4.1.3 Use

The EASIE executive may be invoked by entering

RUN TOAIDE:[PROG]MASTER.EXE

The installation process described in section 5 illustrates an alternate method of execution whereby a VAX/VMS symbol, AIDEX, is used to run the command procedure TOAIDE:[PROC]AIDE.COM, which in turn executes the above command line and performs the terminal selection process. In either case, TOAIDE is a symbol used to specify a partial pathname.

The engineer/designer/analyst user should invoke the EASIE executive as described above from a working area that is to contain the relevant configurations (stored as subdirectories and identified by the *.CFGD description files). During execution, EASIE creates several temporary files using a naming convention similar to that discussed in the previous section. The temporary files are prefixed by 'T$' and identified by the appropriate file extension (APPENDIX A). In order to permit multiple users to simultaneously execute EASIE from the same working area, the VAX/VMS symbol 'TT' corresponding to the user's terminal is expanded and added to the file name. For example, the temporary log file may have a name such as 'T$TTA7.LOG'.

The EASIE executive deletes the temporary files upon logout except the *.TRM and *.LOG files remain. If the executive terminates abnormally, other 'T$' files may reside in the working area. If an abort occurs during the execution of an application program, the directory pointer may be positioned inside a configuration. In order to remedy this problem, reposition the

directory pointer to the working area and delete the temporary files.

4.2 Data Management Utilities

The purpose of this section is twofold. The first is to document the location and function of both the SYSTEM LIBRARY PROCESSOR, software used to construct and manipulate the SYSTEM LIBRARY (including DATA DICTIONARY, TEMPLATE LIBRARY, and PROGRAM DICTIONARY), and the REVIEWER, software used to review/modify data [2]. The second is to explain modifications necessary for use of these utilities with a DBMS other than RIM [4].

4.2.1 SYSTEM LIBRARY PROCESSOR

The files comprising the SYSTEM LIBRARY PROCESSOR (executed via the VAX/VMS symbol 'RUNDICT') reside in the [.AIDE.BUILD DICT] directory. Linking the processor is accomplished through execution of the command procedure MAKDICT.COM producing an executable task image MAKDICT.EXE. Examination of this command procedure reveals that the object files (and their associated source files) are required by the processor. The processor currently stores the SYSTEM LIBRARY in a RIM database and can interact with it both to produce a database containing a schema only and to construct a FORTRAN code to retrieve/replace data from/in the database.

An input file to be read by interactive RIM is created by the module MAKSCHEMA.FOR called by selecting the BS (Build Schema) command. The input file created contains commands to interactive RIM for creating the relations defined in the DATA DICTIONARY using the default database name, DATADB. The database name may be changed by editing this file before execution.

The FORTRAN code produced by the SYSTEM LIBRARY PROCESSOR (the BF command) is written by the modules FORMAT.FOR and FORMRIM.FOR. The module FORMAT.FOR does the bulk of the work of transferring data to/from variables from/to a single integer array. The module FORMRIM.FOR writes the calls to the RIM FORTRAN interface library. Resident within the processor is a user-supplied parameter for the number of characters per machine word to be contained in the storage array (input as the response to the processor's first prompt). This parameter is required because DBMS's (RIM, for example) may not be totally machine independent. On a VAX, RIM character attributes are packed into the storage array using four characters per word. This parameter controls how character attributes will be packed by FORMAT.FOR.

The [.AIDE.BUILD_DICT] directory contains two files which create the SYSTEM LIBRARY Schema. The file BUILD.DAT contains commands to interactive RIM for the schema construction. The file BUILDDICT.COM is a command procedure to copy BUILD.DAT into the default directory of the user, execute interactive RIM, and input the file BUILD.DAT thus creating a schema-only database in the default directory.

Program GETDESC.OBJ (source file GETDESC.FOR), when linked with the RIM Library, produces an executable that reads the Data Dictionary and generates an output file containing all parameters/attributes for each relation, including their names, descriptions, and units. This is useful as a quick overview of the database schema contents (table 2).

All FORTRAN modules described above are written in ANSI standard FORTRAN V with the exception of variable names exceeding the 6 character ANSI standard. The processor will successfully compile and link on both the VAX/VMS and PRIME/PRIMOS systems with only one required modification. The FORTRAN input and output unit numbers are assigned through a data statement in the main program (IN=5 for VAX and IN=1 for PRIME; IOUT=6 for VAX and IOUT=1 for PRIME). A modification to these values may be required for conversion to another machine. Dimensions, though large in some cases, are usually assigned as parameter statements, and some may be reduced for use on a non-virtual machine. (However, this is not recommended.)

TABLE 2. - SAMPLE OUTPUT OF PROGRAM GETDESC

THE PARAMETERS FOR THE RELATION DIMEN ARE

NAME	DESCRIPTION	UNIT
LENGTH	BOX LENGTH	M
WIDTH	BOX WIDTH	M
HEIGHT	BOX HEIGHT	M
VOLUME	BOX VOLUME	M

THE PARAMETERS FOR THE RELATION MODEL ARE

NAME	DESCRIPTION	UNIT
NAME	MODEL NAME	
ROTATION	MODEL X,Y,Z ROTATIONS RESPECTIVELY	DEGREES

THE ATTRIBUTES FOR THE RELATION NODES ARE

NAME	DESCRIPTION	UNIT
X	X COORDINATE	M
Y	Y COORDINATE	M
Z	Z COORDINATE	M

THE ATTRBUTES FOR THE RELATION FACES ARE

NAME	DESCRIPTION	UNIT
FACE	FACE CONNECTIVITY (NODE 1 TO 2 TO 3 TO 4 (OR TO 1 IF 4=0) TO 1)	

4.2.2 REVIEWER

The files comprising the REVIEWER reside in the [.AIDE.REVIEW] directory. Upon execution of the command procedure RIMLOD.COM, the source file R4.FOR will be compiled and linked with the RIM library to produce the executable task image R4.EXE.

The value for number of characters per word resident in the SYSTEM LIBRARY PROCESSOR is passed to the REVIEWER through the REVIEWER input file.

Only one VAX dependency is found in the REVIEWER. In the subroutine RDREV, a call to the VAX system function LIB$GET_FOREIGN returns the name of the REVIEWER input file to read and optionally the word "PRINT" (if the output is to be written to a file). For conversion to another machine, this mechanism must be emulated. Otherwise, the above statements concerning ANSI FORTRAN V standards, input/output units, and dimensions apply.

4.2.3 Conversion to Another DBMS

The previous discussion of the SYSTEM LIBRARY PROCESSOR and the REVIEWER indicates a dependence on the RIM [4] DBMS. Another version of the EASIE data management tools which employs Structural Dynamics Research Corporation's PEARL [5] DBMS has been developed. The intent of this section is to contrast the RIM and PEARL implementations in the event that another DBMS is employed.

Several of the modules located in the [.AIDE.BUILD_DICT] are particularly noteworthy for conversion to another DBMS. The

processor currently stores the SYSTEM LIBRARY in a RIM database but may interact with either the RIM or PEARL DBMS.

The SYSTEM LIBRARY PROCESSOR BS command was previously described as a mechanism for building a RIM input file for creating relations within the DATA DICTIONARY. Alternatively, an option of the BS command will call the module MAKPRLSCH.FOR to create a FORTRAN program that, when linked with SDRC's PEARL library and executed, will create the schema for the PEARL database, DATADB. These methods represent two different means of creating the initial database schema. One method is through commands to an interactive database manager, and the second method is by interfacing with a FORTRAN library. One or both of these modules should serve as a model to create the schema for other relational DBMS's.

FORTRAN code produced by the SYSTEM LIBRARY PROCESSOR (the BF command) (written by the modules FORMAT.FOR and FORMRIM.FOR) may be somewhat database independent provided the DBMS of choice retrieves/stores data in a manner analogous to RIM and PEARL.

The user-supplied parameter indicating the number of characters per machine word used in controlling the packing of character attributes requires further explanation. On a VAX, RIM character attributes are packed into the storage array using 4 characters per word, while on the CDC Cyber/NOS computer, RIM character attributes are packed using 10 characters per word. Using the PEARL DBMS on VAX, character attributes are packed using 2 characters per word. This parameter may be useful in conversion to another DBMS.

If the DBMS of choice packs relation rows into a temporary storage array using the same method as RIM and PEARL, it is possible that only FORMRIM.FOR need be modified. If not, modifications to FORMAT.FOR may be required.

The interface with PEARL was accomplished with no modifications to either FORMAT.FOR or FORMRIM.FOR because PEARL is similar to RIM. Instead, an interface library was created (with few subroutines) to convert RIM subroutine calls to PEARL calls. Code produced by the processor is then identical for RIM or PEARL with the exception of number of characters per word. A program may then be linked with both the RIM-to-PEARL interface library and the PEARL library in lieu of the RIM library. This approach is simple and results in no noticeable loss in performance.

The SYSTEM LIBRARY is stored in a RIM database. If RIM is not acceptable, conversion to another DBMS should be possible. All RIM subroutine calls have been isolated in the module RDDICT and can be replaced by calls to another DBMS. Also, the files BUILDDICT.COM and BUILD.DAT (residing in the [.AIDE.BUILD_DICT] directory) used in creating the schema-only database must be edited accordingly to create the SYSTEM LIBRARY schema using another DBMS.

Conversion of the REVIEWER to another DBMS may be accomplished through modification of the following subroutines:

DBOPEN

DBCLOSE

RDPARA

RDATT

WHRVAL

DB2REV

WRTPARA

WRTATTR

REV2DB

The approach used for the PEARL interface requires no changes to the REVIEWER. (Instead, the REVIEWER is linked with the RIM-to-PEARL library and the PEARL library.) An analogous approach may be possible with other DBMS's.

5.0 INSTALLATION

The EASIE installation tape was written on a VAX 11/785 running VMS 4.5. The magnetic tape is 9-track, 1600 BPI and written with the VAX/VMS BACKUP utility. The BACKUP utility was utilized because the so-called "SAVESET" preserves the hierarchical directory structure described in Section 2. APPENDIX D contains the INSTALLATION TAPE BACKUP listing.

Using BACKUP to read the installation tape, the EASIE hierarchy is created starting from the current working area where the directory pointer is positioned. The highest level directory is EASIETAPE and is referenced in this section using the relative file specification [.EASIETAPE].

The command file BUILD.COM (APPENDIX E) located in the .EASIETAPE directory may be used to generate the required EASIE executables and sample environments (section 2) positioned properly within the hierarchy. Before invoking this command file, editing the symbol definitions in BUILD.COM to conform to local disk logical name assignments is required. As alluded to earlier in this document, these symbols are defined to facilitate moving EASIE from one area or system to another.

The symbol TOAIDE defines a file specification from the logical disk name to the AIDE subdirectory. The symbols LOADRIM and LOADP10 identify the locations of the proprietary RIM and PLOT-10 TCS libraries, respectively, used by the EASIE utilities. (The PLOT-10 library used by the executive and the REVIEWER is not mandatory for Tektronix 41xx series or alphanumeric terminals. The RIM FORTRAN interface library is

required for all EASIE data management utilities.) The USERLIB symbol identifies the location of the two user libraries delivered as part of the installation package. By default the libraries are located in the [.EASIETAPE] subdirectory.

The other symbols (RUNRIM, RUNDICT, REVIEW, and AIDEX) defined in BUILD.COM are not used by the command file. The utility of these symbols is addressed in the next section. Of these four symbols, only RUNRIM, which is used to execute interactive RIM, must be redefined by the installer.

Once the symbols are defined, BUILD.COM runs other command files (from the appropriate areas) which link the necessary object modules to build the EASIE components. The implication here is that these command files do not perform recompilation but use the object files from the installation tape. If recompilation is deemed necessary, the individual command files may be scanned to identify source modules. No special compiler options are required.

After the symbols are defined and any necessary recompilations are performed, BUILD.COM may run from the [.EASIETAPE] subdirectory by issuing the VAX/VMS command:

```
@BUILD.COM
```

A listing of this command file is provided in APPENDIX E.

6.0 MAINTENANCE

Assuming the instructions from section 5 were performed successfully, EASIE should now be usable by potential users wishing to exercise the sample program environment detailed in Volume II.

In an effort to aid the engineer/designer/analyst exercising the EASIE utilities, the symbols RUNRIM, RUNDICT, REVIEW, and AIDEX, defined by [.EASIETAPE]BUILD.COM, (APPENDIX E), may be used. The symbols RUNRIM, RUNDICT, REVIEW, and AIDEX are used to execute interactive RIM, the EASIE data dictionary utilities, the REVIEWER, and the executive, respectively. The symbols RUNDICT, REVIEW, and AIDEX use another symbol, TOAIDE, in their definition. TOAIDE designates the location of the EASIE hierarchy based on the established logical disk configuration and should be set when this software is installed. The RUNRIM symbol is also system dependent and must be defined when the software is installed.

A recommended approach for defining these symbols is to imbed their definitions within the system and/or user's "login" file. In this manner, the symbols are always available to the EASIE user. Note that entering AIDEX to invoke the executive actually runs another command file TOAIDE:[PROG]AIDE.COM. This command file is included with the installation package.

The beginning EASIE user executes EASIE from a working area and not from the EASIE area. The directory [.EASIETAPE] and the subordinate subdirectories should be protected from over writing

by the design manager/system administrator responsible for maintaining the EASIE tools. More specifically, the engineer/designer/analyst should have rights to copy master configurations and to execute EASIE or applications programs but should not be permitted write access to this area. This area requires modifications only when correcting an error in an EASIE utility, adding an update, or changing a particular application environment. The latter case might include such activities as adding a new application program, editing an old application program, adding a master configuration, or editing the procedures and menus used to tailor an existing environment.

By contrast, the engineer/designer/analyst should:

- Ensure that the aforementioned symbols are defined.
- Attach to the area where the user configurations procedures, and workspaces reside.
- Initiate execution of the desired EASIE utility.
- Issue commands as discussed in volumes II and III.

One final caution concerns the EASIE executive convention of "spawning" VMS processes to execute certain commands. Depending on system dependent parameters, the error message

"EXPRCLM, exceeded subprocess quota"

may abort the executive. If this error occurs, the VAX system administrator should increase the PRCLM parameter.

APPENDIX A

EASIE FILE EXTENSIONS

NAME	DESCRIPTION
APPL	APPLICATION PROGRAM
APPLD	APPLICATION DESCRIPTION FILE
CFCD	CONFIGURATION DESCRIPTION FILE
COM	BATCH COMMAND FILE
DIR	CONFIGURATION (Also VAX/VMS Subdirectories)
EXE	APPLICATION PROGRAM (Also a VAX/VMS Executable Image)
LOG	EASIE LOG FILE
PROC	EASIE PROCEDURE FILE
PROCD	PROCEDURE DESCRIPTION FILE
REV	EASIE REVIEWER (TEMPLATE) FILE
SAV	BATCH OUTPUT FILE
TPL	EASIE TEMPLATE FILE
TPLD	TEMPLATE DESCRIPTION FILE
UTIL	USER-WRITTEN UTILITY FILE
xxxx_i	"i(th)" MENU FILE ASSOCIATED WITH PROCEDURE xxxx
WS	EASIE WORKSPACE FILE
WSD	EASIE WORKSPACE DESCRIPTION FILE

APPENDIX B

COMMAND FILE FOR AUTOMATING APPLICATION PROGRAM ENVIRONMENT SETUP

```
$HOME = F$DIRECTORY()
$INQUIRE ID "ENTER EASIE SYSTEM ID "
$STRLEN = F$LENGTH(ID)
$IF (STRLEN .LT. 1) THEN WRITE SYS$OUTPUT "ID MUST BE GT 0
CHARACTERS"
$IF (STRLEN .LT. 1) THEN GOTO EXIT
$IF (STRLEN .GT. 8) THEN WRITE SYS$OUTPUT "ID MUST BE LE 8
CHARACTERS"
$IF (STRLEN .GT. 8) THEN GOTO EXIT
$CREATE/DIR TOAIDE:['ID']
$CREATE/DIR TOAIDE:['ID'.CFG]
$CREATE/DIR TOAIDE:['ID'.CFG.MASTER]
$CREATE TOAIDE:['ID'.CFG]MASTER.CFGD
$CREATE/DIR TOAIDE:['ID'.DICTNRY]
$CREATE/DIR TOAIDE:['ID'.SOURCE]
$CREATE/DIR TOAIDE:['ID'.PROG]
$TA = F$LOGICAL("TOAIDE")
$LEN = F$LENGTH(TA)
$TA = F$EXTRACT(0,LEN-1,TA)
$TA = TA+ID+".DICTNRY]"
$WRITE SYS$OUTPUT TA
$SET DEFAULT 'TA'
$COPY TOAIDE:[BUILD_DICT]BUILD.DAT *
$RUNRIM
INP BUILD
EXIT
$DELETE BUILD.DAT.*
$IF (STRLEN .EQ. 1) THEN B = "        "
$IF (STRLEN .EQ. 2) THEN B = "       "
$IF (STRLEN .EQ. 3) THEN B = "      "
$IF (STRLEN .EQ. 4) THEN B = "     "
$IF (STRLEN .EQ. 5) THEN B = "    "
$IF (STRLEN .EQ. 6) THEN B = "   "
$IF (STRLEN .EQ. 7) THEN B = "  "
$IF (STRLEN .EQ. 8) THEN B = " "
$OPEN/APPEND MWS TOAIDE:[DB]MASTER_WS.DAT
$WRITE MWS ID,B,"TOAIDE:[",ID,".CFG]",ID,".WS"
$CLOSE MWS
$OPEN/WRITE WS TOAIDE:['ID'.CFG]'ID'.WS
$WRITE WS "USRID                ",ID
$WRITE WS "USRLVL                 1"
$WRITE WS "MENUCMD                F"
$WRITE WS "REFCFG"
$WRITE WS "APPLTPL"
$WRITE WS "REFTMPL"
$WRITE WS "APPLTMPLIN"
$WRITE WS "APPLTMPLOUT"
$WRITE WS "REFPCF"
```

```
$WRITE WS "CURPROG"
$WRITE WS "WKSP                 T$"
$WRITE WS "CURMENU              UTILITY SELECTION (MAIN)"
$WRITE WS "CURMENUIDX              1"
$WRITE WS "INUNIT                  5"
$WRITE WS "OUTIUNIT                6"
$WRITE WS "PRINTLEVL               1"
$WRITE WS "HOMEUFD"
$WRITE WS "BASEUFD              TOAIDE:[",ID,".CFG]"
$WRITE WS "PROGUFD              TOAIDE:[",ID,".PROG]"
$WRITE WS "MENULIST             !"
$WRITE WS "SEQEXEC                 F"
$WRITE WS "SEQNUM                  0"
$WRITE WS "PROCSP                  0"
$WRITE WS "PROCPC                  0"
$WRITE WS "PROCEX                  F"
$WRITE WS "EMPTYPROCSTACK          T"
$WRITE WS "AUTODEFAULT             T"
$WRITE WS "EXPROCFILE"
$CLOSE WS
$SET DEFAULT 'HOME'
$EXIT
$EXIT
$WRITE SYS$OUTPUT "EASIE ID NOT ESTABLISHED"
$EXIT
```

APPENDIX C

EXECUTIVE LINK COMMAND FILE

```
$LINK MASTER -
      +FASTAIDE -
      +RESPPROCS -
      +TEMPSUB -
      +INTERACT -
      +INTERPRET -
      +DISPLAY -
      +PERMENU -
      +GET -
      +GLOBS -
      +ERRORSUBS -
      +RIMSUB -
      +SYSUTIL -
      +TERMINAL -
      +EDTPL -
      +SLOG -
      +BATCH -
      +GETMENU -
      +AIDEUTIL -
      +USERLIB:TEK4100/LIBRARY -
      +USERLIB:CSCLIB/LIBRARY -
      +LOADP10/LIBRARY
```

APPENDIX D

INSTALLATION TAPE BACKUP LISTING

```
LISTING OF SAVE SET(S)

SAVE SET:    EASIE.BCK
WRITTEN BY: CSC7
UIC:         [000100,000070]
DATE:        27-MAY-1987 14:25:56.71
COMMAND:     BACKUP/LIST=[CSC7.TEAM7.SCOTT]EASIE.LIS/LOG/VERIFY
             [CSC.EASIETAPE...] TAPE:EASIE.BCK/REWIND/DENSITY=1600
OPERATING SYSTEM: VAX/VMS VERSION V4.5
BACKUP VERSION:   V4.5
CPU ID REGISTER:  018432A5
NODE NAME:         SSD2::
WRITTEN ON:       MSA0:
BLOCK SIZE:       8192
GROUP SIZE:       10
BUFFER COUNT:     3

[CSC.EASIETAPE]AIDE.DIR;1                                    1
[CSC.EASIETAPE.AIDE]BUILD_DICT.DIR;1                         2
[CSC.EASIETAPE.AIDE.BUILD_DICT]BUILD.DAT;1                   6
[CSC.EASIETAPE.AIDE.BUILD_DICT]BUILDDICT.COM;1               1
[CSC.EASIETAPE.AIDE.BUILD_DICT]BUILD_EASIE.COM;2             1
[CSC.EASIETAPE.AIDE.BUILD_DICT]BUILD_EASIE.FOR;4            31
[CSC.EASIETAPE.AIDE.BUILD_DICT]BUILD_EASIE.OBJ;2            29
[CSC.EASIETAPE.AIDE.BUILD_DICT]COMPALL.COM;1                 1
[CSC.EASIETAPE.AIDE.BUILD_DICT]FORMAT.FOR;1                 70
[CSC.EASIETAPE.AIDE.BUILD_DICT]FORMAT.OBJ;1                 45
[CSC.EASIETAPE.AIDE.BUILD_DICT]FORMRIM.FOR;1                71
[CSC.EASIETAPE.AIDE.BUILD_DICT]FORMRIM.OBJ;1                59
[CSC.EASIETAPE.AIDE.BUILD_DICT]GETDESC.FOR;1                 5
[CSC.EASIETAPE.AIDE.BUILD_DICT]GETDESC.OBJ;1                 6
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKDICT.COM;1                 1
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKDICT.FOR;1                83
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKDICT.OBJ;1                60
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKDICTDB.COM;1               1
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKPRLSCH.FOR;1              13
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKPRLSCH.OBJ;1               9
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKREV.FOR;1                 11
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKREV.OBJ;1                 10
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKSCHEMA.FOR;1               8
[CSC.EASIETAPE.AIDE.BUILD_DICT]MAKSCHEMA.OBJ;1               6
[CSC.EASIETAPE.AIDE.BUILD_DICT]MODDICT.FOR;1                 9
[CSC.EASIETAPE.AIDE.BUILD_DICT]OPEN.FOR;1                    6
[CSC.EASIETAPE.AIDE.BUILD_DICT]OPEN.OBJ;1                    6
[CSC.EASIETAPE.AIDE.BUILD_DICT]PROGRAM.FOR;1                 7
[CSC.EASIETAPE.AIDE.BUILD_DICT]PROGRAM.OBJ;1                 7
[CSC.EASIETAPE.AIDE.BUILD_DICT]PUTDESC.FOR;1                 4
[CSC.EASIETAPE.AIDE.BUILD_DICT]RDDICT.FOR;1                 84
[CSC.EASIETAPE.AIDE.BUILD_DICT]RDDICT.OBJ;1                 57
```

APPENDIX D

```
[CSC.EASIETAPE.AIDE]DB.DIR;1                                    1
[CSC.EASIETAPE.AIDE.DB]APEX.DAT;1                               5
[CSC.EASIETAPE.AIDE.DB]CMDS.DAT;1                               2
[CSC.EASIETAPE.AIDE.DB]DATA.DAT;1                               5
[CSC.EASIETAPE.AIDE.DB]DEFAULT.WS;2                             2
[CSC.EASIETAPE.AIDE.DB]HELP.DAT;1                               2
[CSC.EASIETAPE.AIDE.DB]MAIN.DAT;2                               2
[CSC.EASIETAPE.AIDE.DB]MASTER_WS.DAT;3                          1
[CSC.EASIETAPE.AIDE.DB]PBLD.DAT;1                               4
[CSC.EASIETAPE.AIDE.DB]PREX.DAT;1                               7
[CSC.EASIETAPE.AIDE.DB]TBLD.DAT;1                               3
[CSC.EASIETAPE.AIDE.DB]WSC.DAT;1                                7
[CSC.EASIETAPE.AIDE]DBSAV.DIR;1                                 1
[CSC.EASIETAPE.AIDE.DBSAV]AIDMP.DAT;1                          27
[CSC.EASIETAPE.AIDE.DBSAV]IDEASDMP.DAT;1                        8
[CSC.EASIETAPE.AIDE]EXAMPLE.DIR;1                               1
[CSC.EASIETAPE.AIDE.EXAMPLE]CFG.DIR;1                           1
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG]DEFAULT.CFGD;1                  1
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG]DEFAULT.DIR;1                   1
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG.DEFAULT]DATADB1.DAT;1          24
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG.DEFAULT]DATADB2.DAT;1           8
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG.DEFAULT]DATADB3.DAT;1           1
[CSC.EASIETAPE.AIDE.EXAMPLE.CFG]EXAMPLE.WS;1                    2
[CSC.EASIETAPE.AIDE.EXAMPLE]DICTNRY.DIR;1                       1
[CSC.EASIETAPE.AIDE.EXAMPLE.DICTNRY]DICT1.DAT;1                24
[CSC.EASIETAPE.AIDE.EXAMPLE.DICTNRY]DICT2.DAT;1                 8
[CSC.EASIETAPE.AIDE.EXAMPLE.DICTNRY]DICT3.DAT;1                 1
[CSC.EASIETAPE.AIDE.EXAMPLE]PROG.DIR;1                          1
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]BOX.APPLD;1                    1
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]BOXIN.REV;1                    2
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]BOXIN.TPLD;1                   0
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]BOXOUT.REV;1                   1
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]BOXOUT.TPLD;1                  0
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]DRAWIN.REV;1                   3
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]DRAWIN.TPLD;1                  0
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]DRAWIT.APPLD;1                 1
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]MAKGEO.APPLD;1                 1
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]MAKGEOIN.REV;1                 2
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]MAKGEOIN.TPLD;1                0
[CSC.EASIETAPE.AIDE.EXAMPLE.PROG]PROGRAM.PGD;1                  2
[CSC.EASIETAPE.AIDE.EXAMPLE]SOURCE.DIR;1                        2
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOX.COM;1                    1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOX.FOR;1                    1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOX.OBJ;7                    1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXIN.COMMON;1               1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXIN.FOR;1                  6
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXIN.OBJ;7                  3
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXOUT.COMMON;1              1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXOUT.FOR;1                 4
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]BOXOUT.OBJ;7                 3
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAW.COM;1                   1
```

APPENDIX D

```
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAW.FOR;1              6
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAW.OBJ;7              7
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAWIN.ASSIGN;1         2
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAWIN.COM;1            1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAWIN.FOR;1           11
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]DRAWIN.OBJ;6            8
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEO.COM;1            1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEO.FOR;1            3
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEO.OBJ;6            2
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOIN.COMMON;1       1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOIN.FOR;1          6
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOIN.OBJ;6          3
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOOT.COMMON;1       1
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOOT.FOR;1          9
[CSC.EASIETAPE.AIDE.EXAMPLE.SOURCE]MAKGEOOT.OBJ;6          4
[CSC.EASIETAPE.AIDE]EXMENU.DIR;1                           1
[CSC.EASIETAPE.AIDE.EXMENU]CFG.DIR;1                       1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.PROC;1               1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.PROCD;1              1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.PROC_1;1             1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.PROC_2;1             1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.PROC_3;1             1
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.WS;1                 2
[CSC.EASIETAPE.AIDE.EXMENU.CFG]EXMENU.WSD;1                1
[CSC.EASIETAPE.AIDE]EXSIMPLE.DIR;1                         1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOX.COM;1                     1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOX.FOR;1                     1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOX.OBJ;1                     1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXIN.COMMON;1                1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXIN.FOR;1                   6
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXIN.OBJ;1                   3
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXIN.REV;1                   2
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXOUT.COMMON;1               1
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXOUT.FOR;1                  4
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXOUT.OBJ;1                  3
[CSC.EASIETAPE.AIDE.EXSIMPLE]BOXOUT.REV;1                  1
[CSC.EASIETAPE.AIDE.EXSIMPLE]DATADB1.DAT;1                24
[CSC.EASIETAPE.AIDE.EXSIMPLE]DATADB2.DAT;1                 8
[CSC.EASIETAPE.AIDE.EXSIMPLE]DATADB3.DAT;1                 1
[CSC.EASIETAPE.AIDE.EXSIMPLE]DICT1.DAT;1                  24
[CSC.EASIETAPE.AIDE.EXSIMPLE]DICT2.DAT;1                   8
[CSC.EASIETAPE.AIDE.EXSIMPLE]DICT3.DAT;1                   1
[CSC.EASIETAPE.AIDE.EXSIMPLE]SCHEMA.DAT;1                  1
[CSC.EASIETAPE.AIDE]HELP.DIR;1                             1
[CSC.EASIETAPE.AIDE.HELP]ACT.DAT;1                         7
[CSC.EASIETAPE.AIDE.HELP]CMDS.DAT;1                        2
[CSC.EASIETAPE.AIDE.HELP]EDIT.DAT;1                        4
[CSC.EASIETAPE.AIDE.HELP]ERRS.DAT;1                        8
[CSC.EASIETAPE.AIDE.HELP]HELPFILES.DAT;1                  27
[CSC.EASIETAPE.AIDE.HELP]IND.DAT;1                         1
[CSC.EASIETAPE.AIDE.HELP]INDX.DAT;1                        2
```

APPENDIX D

```
[CSC.EASIETAPE.AIDE.HELP]OBJ.DAT;1                    6
[CSC.EASIETAPE.AIDE.HELP]PERM.DAT;1                   3
[CSC.EASIETAPE.AIDE]LOGIN.COM;1                       1
[CSC.EASIETAPE.AIDE]PROC.DIR;1                        1
[CSC.EASIETAPE.AIDE.PROC]AIDE.COM;2                   1
[CSC.EASIETAPE.AIDE.PROC]AIDEINIT.COM;8               5
[CSC.EASIETAPE.AIDE.PROC]ATTDIR.COM;5                 1
[CSC.EASIETAPE.AIDE.PROC]COPYCFG.COM;5                1
[CSC.EASIETAPE.AIDE.PROC]DELDB.COM;8                  1
[CSC.EASIETAPE.AIDE.PROC]LSTDIR.COM;3                 1
[CSC.EASIETAPE.AIDE.PROC]OPNBATCH.COM;5               1
[CSC.EASIETAPE.AIDE.PROC]UNAIDEINIT.COM;16            1
[CSC.EASIETAPE.AIDE]PROG.DIR;1                        4
[CSC.EASIETAPE.AIDE.PROG]AIDEUTIL.FOR;33             22
[CSC.EASIETAPE.AIDE.PROG]AIDEUTIL.OBJ;6              20
[CSC.EASIETAPE.AIDE.PROG]AIDEUTIL.OLB;6              72
[CSC.EASIETAPE.AIDE.PROG]BATCH.FOR;32                14
[CSC.EASIETAPE.AIDE.PROG]BATCH.OBJ;2                 21
[CSC.EASIETAPE.AIDE.PROG]CRELNM.FOR;42                6
[CSC.EASIETAPE.AIDE.PROG]DCOMPILE.COM;3               1
[CSC.EASIETAPE.AIDE.PROG]DISPLAY.FOR;25              15
[CSC.EASIETAPE.AIDE.PROG]DISPLAY.OBJ;4               20
[CSC.EASIETAPE.AIDE.PROG]DLOAD.COM;36                 1
[CSC.EASIETAPE.AIDE.PROG]EDTPL.FOR;8                 32
[CSC.EASIETAPE.AIDE.PROG]EDTPL.OBJ;3                 33
[CSC.EASIETAPE.AIDE.PROG]ERRORSUBS.FOR;6             11
[CSC.EASIETAPE.AIDE.PROG]ERRORSUBS.OBJ;2             17
[CSC.EASIETAPE.AIDE.PROG]FASTAIDE.FOR;52             30
[CSC.EASIETAPE.AIDE.PROG]FASTAIDE.OBJ;7              69
[CSC.EASIETAPE.AIDE.PROG]GET.FOR;25                  22
[CSC.EASIETAPE.AIDE.PROG]GET.OBJ;7                   26
[CSC.EASIETAPE.AIDE.PROG]GETMENU.FOR;20              12
[CSC.EASIETAPE.AIDE.PROG]GETMENU.OBJ;7               18
[CSC.EASIETAPE.AIDE.PROG]GLOBALS.INC;1                3
[CSC.EASIETAPE.AIDE.PROG]GLOBS.FOR;8                 11
[CSC.EASIETAPE.AIDE.PROG]GLOBS.OBJ;4                 12
[CSC.EASIETAPE.AIDE.PROG]INTERACT.FOR;10             10
[CSC.EASIETAPE.AIDE.PROG]INTERACT.OBJ;3              24
[CSC.EASIETAPE.AIDE.PROG]INTERPRET.FOR;9             36
[CSC.EASIETAPE.AIDE.PROG]INTERPRET.OBJ;3             44
[CSC.EASIETAPE.AIDE.PROG]LOAD.COM;5                   1
[CSC.EASIETAPE.AIDE.PROG]LOCALS.INC;1                 2
[CSC.EASIETAPE.AIDE.PROG]LOCALS2.INC;1                3
[CSC.EASIETAPE.AIDE.PROG]MASTER.FOR;11                3
[CSC.EASIETAPE.AIDE.PROG]MASTER.OBJ;2                 3
[CSC.EASIETAPE.AIDE.PROG]PERMENU.FOR;31              27
[CSC.EASIETAPE.AIDE.PROG]PERMENU.OBJ;6               41
[CSC.EASIETAPE.AIDE.PROG]PTH2FIL.FOR;27               8
[CSC.EASIETAPE.AIDE.PROG]PUTTRM.COM;9                 1
[CSC.EASIETAPE.AIDE.PROG]PUTTRM.FOR;6                 3
[CSC.EASIETAPE.AIDE.PROG]RESPPROCS.FOR;81           134
[CSC.EASIETAPE.AIDE.PROG]RESPPROCS.OBJ;29           157
```

```
[CSC.EASIETAPE.AIDE.PROG]RIMSUB.FOR;45                36
[CSC.EASIETAPE.AIDE.PROG]RIMSUB.OBJ;4                 43
[CSC.EASIETAPE.AIDE.PROG]SLOG.FOR;13                  50
[CSC.EASIETAPE.AIDE.PROG]SLOG.OBJ;4                   50
[CSC.EASIETAPE.AIDE.PROG]SYSUTIL.FOR;55               23
[CSC.EASIETAPE.AIDE.PROG]SYSUTIL.OBJ;6                25
[CSC.EASIETAPE.AIDE.PROG]SYSUTIL.OLB;5                77
[CSC.EASIETAPE.AIDE.PROG]TEMPSUB.FOR;45               53
[CSC.EASIETAPE.AIDE.PROG]TEMPSUB.OBJ;5                84
[CSC.EASIETAPE.AIDE.PROG]TERMINAL.FOR;22              13
[CSC.EASIETAPE.AIDE.PROG]TERMINAL.OBJ;2               17
[CSC.EASIETAPE.AIDE.PROG]TERMINAL.OLB;2               68
[CSC.EASIETAPE.AIDE]REVIEW.DIR;1                       1
[CSC.EASIETAPE.AIDE.REVIEW]R4.FOR;1                  196
[CSC.EASIETAPE.AIDE.REVIEW]R4.OBJ;1                  123
[CSC.EASIETAPE.AIDE.REVIEW]RIMLOD.COM;2                1
[CSC.EASIETAPE]BUILD.COM;3                             9
[CSC.EASIETAPE]CSCLIB.FOR;1                           24
[CSC.EASIETAPE]CSCLIB.OLB;1                           66
[CSC.EASIETAPE]TEK4100.FOR;1                          72
[CSC.EASIETAPE]TEK4100.OLB;1                         144

TOTAL OF 223 FILES, 3366 BLOCKS
END OF SAVE SET.
```

APPENDIX E

BUILD EASIE EXECUTABLES COMMAND FILE

```
$!*************************************************
$!*****      BUILD.COM IS A VMS 4.4 COMMAND     *****
$!*****      FILE USED TO LINK THE REQUIRED     *****
$!*****      EASIE EXECUTABLES.                 *****
$!*****                                         *****
$!*****      IT IS ASSUMED THAT EASIE           *****
$!*****      MODULES WERE TRANSFERRED FROM      *****
$!*****      MAGNETIC TAPE USING THE VMS        *****
$!*****      BACKUP UTILITY (THUS PRESERVING    *****
$!*****      THE NECESSARY DIRECTORY STRUCTURE)*****
$!*****                                         *****
$!*****      THE TOP LEVEL DIRECTORY HAS A      *****
$!*****      RELATIVE PATHNAME OF:              *****
$!*****      [.CSC]                             *****
$!*****      IT IS NECESSARY TO "RUN" THE       *****
$!*****      THE COMMAND FILE, BUILD.COM,       *****
$!*****      WHILE "ATTACHED" TO THIS AREA BY   *****
$!*****      ENTERING:                          *****
$!*****      BUILD                              *****
$!*****                                         *****
$!*****      THE REQUIRED COMMAND, OBJECT, AND *****
$!*****      LIBRARY FILES ARE INCLUDED MAKING *****
$!*****      COMPILATION UNNECESSARY IN MANY    *****
$!*****      INSTANCES.  (IF COMPILATION IS     *****
$!*****      REQUIRED, CHECK THE SPECIFIED      *****
$!*****      SUBDIRECTORY FOR A ".COM" FILE     *****
$!*****      WITH APPROPRIATE COMPILER          *****
$!*****      PARAMETERS.)                       *****
$!*****                                         *****
$!*****      THE INITIAL PORTION OF THE COMMAND*****
$!*****      FILE IS USED FOR DEFINING SYMBOLS.*****
$!*****      THE EASIE IMPLEMENTER MUST RESET   *****
$!*****      THESE SYMBOLS ACCORDING TO HIS/HER*****
$!*****      SYSTEM.   THE SYMBOLS LOADRIM AND  *****
$!*****      LOADP10 ARE USED TO LOAD THE RIM   *****
$!*****      AND PLOT-10 PROPRIETARY LIBRARIES  *****
$!*****      DURING THE LINKING PROCESS.  THE   *****
$!*****      SYMBOL USERLIB IDENTIFIES THE      *****
$!*****      LOCATION OF THE TWO USER OBJECT    *****
$!*****      LIBRARIES, TEK4100 AND CSCLIB.     *****
$!*****      (THESE TWO USER LIBRARIES MAY BE   *****
$!*****      PLACED ANYWHERE BY THE IMPLEMENTER*****
$!*****      AS LONG AS SYMBOL USERLIB REFLECTS*****
$!*****      THE PROPER LOCATION.)              *****
$!*****                                         *****
$!*****      THE OTHER SYMBOLS ARE USED DURING  *****
$!*****      EASIE EXECUTION (IN ADDITION TO    *****
$!*****      THE LINKING PROCESS).  FOR THIS    *****
$!*****      REASON THESE SYMBOLS SHOULD BE     *****
```

```
$!*****       ENTERED INTO A PROSPECTIVES USER'S*****
$!*****       LOGIN.COM FILE.  THE SYMBOLS IN   *****
$!*****       THIS CATEGORY INCLUDE:            *****
$!*****       TOAIDE, RUNRIM, RUNDICT, REVIEW,  *****
$!*****       AND AIDEX.                        *****
$!*****       IN THE CASE OF RUNRIM, THE SYMBOL *****
$!*****       SHOULD INCLUDE THE AREA CONTAINING*****
$!*****       THE RIM EXECUTABLES.              *****
$!*****       THE SYMBOL AIDEX, POINTS TO A     *****
$!*****       COMMAND FILE WHICH MAY BE USED TO *****
$!*****       INITIATE EASIE EXECUTION.         *****
$!**************************************************
$!
$!      DEFINE VMS SYMBOLS
$DEFINE TOAIDE DUB1:[CSC.EASIETAPE.AIDE.]
$DEFINE LOADP10 "DUA2:[LIBS]PLOT10"
$DEFINE LOADRIM "SYS$SYSDEVICE:[RIM.RIM6]RIMLIB"
$DEFINE USERLIB DUB1:[CSC.EASIETAPE]
$RUNRIM :== R SYS$SYSDEVICE:[RIM.RIM6]RIM
$RUNDICT :== R TOAIDE:[BUILD_DICT]MAKDICT
$REVIEW :== $TOAIDE:[REVIEW]R4
$AIDEX :== @TOAIDE:[PROC]AIDE.COM
$!
$!      BUILD THE EASIE EXECUTIVE
$SET DEF [.AIDE.PROG]
$@LOAD
$@PUTTRM
$PURGE
$!      BUILD REVIEWER
$SET DEF [-.REVIEW]
$@RIMLOD
$PURGE
$!      BUILD THE DATA DICTIONARY TEMPLATE UTILITIES
$SET DEF [-.BUILD_DICT]
$@MAKDICT
$@BUILD_EASIE
$PURGE
$!      BUILD THE APPLICATION PROGRAMS CORRESPONDING
$!      TO DOCUMENTED EXAMPLES
$SET DEF [-.EXSIMPLE]
$@BOX
$PURGE
$SET DEF [-.EXAMPLE.SOURCE]
$@BOX
$@DRAW
$@MAKGEO
$@DRAWIN
$SET DEF [-.PROG]
$PURGE
$SET DEF [---]
```

REFERENCES

1. Lawrence F. Rowell; and John S. Davis: The Environment For Application Software Integration and Execution (EASIE) Version 1.0. VOLUME I - EXECUTIVE OVERVIEW. NASA TM-100573, April 1988.

2. Kennie H. Jones; Donald P. Randall; Scott S. Stallcup; and Lawrence F. Rowell: The Environment For Application Software Integration and Execution (EASIE) Version 1.0. VOLUME II - PROGRAM INTEGRATION GUIDE. NASA TM-100574, April 1988.

3. Dr. James L. Schwing; Lawrence F. Rowell; and Russell E. Criste: The Environment For Application Software Integration and Execution (EASIE) Version 1.0. VOLUME III - PROGRAM EXECUTION GUIDE. NASA TM-100575, April 1988.

4. Boeing Computer Services: BCS RIM - Relational Information Management System Version 6.0 User Guide. The Boeing Company, 1983.

5. Structural Dynamics Research Corp.: I-DEAS User Guide Level 3. 5201.004, March, 1986.

NASA

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-100576		

4. Title and Subtitle
The Environment For Application Software Integration and Execution (EASIE) Version 1.0, Volume IV - System Installation And Maintenance Guide

5. Report Date
April 1988

6. Performing Organization Code

7. Author(s)
Donald P. Randall, Kennie H. Jones, and Lawrence F. Rowell

8. Performing Organization Report No.

10. Work Unit No.
506-49-31-01

9. Performing Organization Name and Address
NASA Langley Research Center
Hampton, VA 23665-5225

11. Contract or Grant No.

13. Type of Report and Period Covered
Technical Memorandum

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Washington, DC 20546-0001

14. Sponsoring Agency Code

15. Supplementary Notes
Donald P. Randall and Kennie H. Jones, Computer Sciences Corporation, Hampton, VA.
Lawrence F. Rowell, Langley Research Center, Hampton, VA.

16. Abstract The Environment for Application Software Integration and Execution, EASIE, provides both a methodology and a set of software utility programs to ease the task of coordinating engineering design and analysis codes. This document provides the necessary information for installing the EASIE software on a host computer system. The target host is a DEX VAX running VMS version 4, but host dependencies are noted when appropriate. Relevant directories and individual files are identified, and compile/load/execute sequences are specified. In the case of the data management utilities, database management system (DBMS) specific features are described in an effort to assist the maintenance programmer in converting to a new DBMS. The document also details a sample EASIE program directory structure to guide the program implementer in establishing his/her application dependent environment.

17. Key Words (Suggested by Author(s))
EASIE
Executive Software
Program Integration

18. Distribution Statement
Unclassified - Unlimited
Subject category - 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	49	A03