U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4683

NAT'L INST. OF STAND & TECH R.I.C.
A11103 710400

National PDES Testbed Report Series

REFERENCE

NIST PUBLICATIONS

The Validation Testing Laboratory User's Guide

NATIONAL PDES TESTBED

QC
100
.U56
#4683
1991

NIST

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4683

National PDES Testbed

NATIONAL PDES TESTBED

The Validation Testing Laboratory User's Guide

J. Newton Breese
Michael McLay
Gerard Silvernale

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher,
Secretary of Commerce

National Institute of Standards and Technology
John W. Lyons, Director

May 2, 1991

Abstract

Product Data Exchange Using STEP (PDES) is the United States activity supporting the Standard for the Exchange of Product Model Data (STEP), a proposed international standard for product data representation and exchange.

This document is a quick reference for those using the Validation Testing Laboratory of the National PDES Testbed located at the National Institute of Standards and Technology (NIST) in Gaithersburg, Maryland. It is not intended for those wishing to learn about the tools in the Testbed, but is rather for those who already have a working knowledge of the programs and terminology.

The Testbed was initiated in 1988 under the sponsorship of the U.S. Department of Defense Computer-aided Acquisition and Logistic Support (CALS) program. A major goal of the Testbed is to provide technical leadership in a national effort to implement a complete and useful specification for the exchange of product data. This specification must be designed to meet the needs of U.S. industry and the CALS program.

The National PDES Testbed supports and actively participates in the international effort to develop the Standard for the Exchange of Product Model Data (STEP). The STEP development effort is led by the International Organization for Standardization (ISO) TC184/SC4.

Disclaimer: Certain commercial products are identified in this document in order to specify adequately the configuration management procedures. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products identified are necessarily the best available for the purpose.

The work described was funded by the United States Government and is not subject to copyright.

Acknowledgements

The authors would like to recognize their colleagues who were also involved in collecting and organizing the information presented in this document. In particular, Kathleen Strouse and Mary Mitchell carried out the initial effort and the production of the preliminary document that formed the basis for this document. In addition, we would like to thank the members of the PDES Testbed Hotline team, Scott Bodarky, Sharon Gray, Clarence Johnson, Deborah Nickerson, and Gaylen Rinaudot, for their critical and technical input.

Introduction 1-1

1 General information 1-1
1.1 User Guide conventions 1-2
1.2 Testbed activities 1-3
1.3 Accounts 1-3
1.3.1 Network Information System accounts 1-4
1.3.2 ComputerVision accounts 1-4
1.3.3 VAX accounts 1-4
1.3.4 Database accounts 1-4
1.4 Documentation 1-5
1.5 Using a modem 1-6
1.6 Validation Testing Laboratory environment 1-6

2 Logging in and logging out 2-1
2.1 Logging in and starting X Windows 2-1
2.2 Logging out 2-2

3 UNIX 3-1
3.1 Assigning or changing your password 3-1
3.2 Directories 3-2
3.2.1 Changing directories 3-2
3.2.2 Creating directories 3-3
3.2.3 Removing directories 3-3
3.3 Files 3-4
3.3.1 File permissions 3-4
3.3.2 Listing file names 3-4
3.3.3 Copying files 3-5
3.3.4 Moving files 3-5
3.3.5 Displaying files with "more" 3-6
3.3.6 Searching files 3-6
3.3.6.1 Using "more" 3-6
3.3.6.2 Using "grep" 3-6
3.3.7 Removing files 3-7
3.4 Creating command aliases 3-7
3.5 Printing and plotting 3-7
3.6 Standard input and output 3-8
3.7 Shell scripts 3-8

3.8 Getting help - the "man pages" 3-8

4 General utilities 4-1
4.1 Editors 4-1
4.1.1 The default editor, vi 4-1
4.1.2 Emacs 4-1
4.2 Mail 4-2
4.2.1 Mailing lists 4-2
4.2.2 Addressing mail using aliases 4-3
4.2.3 UNIX mail 4-3
4.2.3.1 Sending UNIX mail 4-3
4.2.3.2 Inserting files 4-4
4.2.3.3 Sending copies 4-5
4.2.3.4 Confirming a letter 4-6
4.2.4 Emacs mail 4-6
4.2.4.1 Sending Emacs mail 4-6
4.2.4.2 Reading Emacs mail 4-8
4.2.4.3 Sending replies 4-10
4.3 News 4-10
4.4 Configuration management 4-12

5 Remote login and file transfers 5-1
5.1 Remote login to other systems 5-1
5.2 File transfers 5-1
5.2.1 FTP file transfers 5-1
5.2.2 Kermit file transfers 5-2

6 The ComputerVision/CADDS geometric modeler 6-1
6.1 Logging into the ComputerVision system 6-1
6.2 Starting the CADDS software 6-1
6.3 Displaying a geometric model 6-2
6.3.1 Retrieving an existing model 6-2
6.3.2 Creating or editing a geometric model 6-3
6.4 Creating the AMRF part model file 6-3

7 The CV-to-STEP file translator 7-1
7.1 Converting to STEP file format 7-1
7.2 Format conformance 7-1

8 The IBM RT/CATIA geometric modeler 8-1
8.1 Logging into the IBM RT system 8-1
8.2 Starting the CATIA software 8-1
8.3 Loading an existing model 8-1
8.4 Plotting 8-2
8.4.1 Define a "sheet" 8-2
8.4.2 Define a "window" 8-2
8.4.3 Set the "scale" 8-2
8.4.4 Finish the plot(s) 8-2
8.5 Converting files from one format to another 8-3
8.5.1 Converting an IGES file to a CATIA model 8-3
8.5.2 Converting a CATIA model to an IGES file 8-4
8.6 Starting CATIA without re-logging on 8-4

9 The IGES to PDES translator 9-1
9.1 Introduction 9-1
9.2 IGES to PDES translator mapping 9-1
9.3 IGES to PDES translator execution 9-2

10 Compiling Express using Fed-X 10-1
10.1 Eliminating nested schemas: Fed-X-flatten 10-1
10.2 Creating a database schema from Express: Fed-X-SQL 10-1
10.3 Creating a QDES editor image from Express: Fed-X-QDES 10-3

11 The STEP physical file parser: STEPparse 11-1
11.1 The STEP physical file syntax checker: stepparse_step 11-2
11.2 The STEP to QDES translator: stepparse_qdes 11-2
11.3 Loading a STEP file into a database: stepwf_sql 11-2

12 Using the QDES model editor 12-1
12.1 Getting started with QDES 12-1
12.1.1 QDES on a Sun with the generic environment 12-1
12.1.2 QDES on a DECstation 3100 12-2
12.2 Loading a new Express File into QDES 12-2

13 The geometry visualizer 13-1
13.1 Introduction 13-1
13.2 Implementation environment 13-1
13.3 Visualizer execution 13-1
13.4 Main menu options 13-2

13.5 Graphics Control Utilities (GCU) 13-3
13.6 Mouse functionality 13-5
13.7 Restrictions 13-6

14 The validation databases 14-1
14.1 The database management system (DBMS) 14-1
14.2 Logging in and out 14-1
14.3 Creating a STEP database 14-2
14.3.1 Account inter-relationships 14-2
14.3.2 Loading the schema into one of the master accounts 14-3
14.3.3 Setting up the tester accounts 14-3
14.4 Database utilities 14-4
14.5 Utility command syntax and use 14-5
14.6 Database backups 14-9

15 Creating a STEP physical file from a relational database 15-1

16 STEP file utilities 16-1
16.1 Converting from old to new format 16-1
16.2 Searching a STEP file for entities 16-1
16.3 Reformatting STEP physical files 16-2
16.3.1 The print exchange file (pef) filter 16-2
16.3.2 The remove non-printing (rmvnonpr) characters filter 16-3
16.3.3 Adding linefeeds to STEP physical files 16-3
16.3.4 Renumbering STEP physical file entities 16-4

17 Bugs and updates 17-1
17.1 Reporting bugs 17-1
17.2 The PDES Testbed Hotline 17-1
17.3 Getting updates 17-1

Figures

Figure 1 FASD/PDES Computers Fig-1
Figure 2 FASD Global Systems Fig-2
Figure 3 Testbed STEP Software (BPR2.1/VTS 2.0) Fig-3

Appendices

Appendix A Dial-in to the National PDES Testbed at NIST A-1
Appendix B Contacts for functional titles B-1
Appendix C Current host assignments C-1
Appendix D Current hardware and corresponding software versions D-1
Appendix E Locations of selected hardware at NIST E-1
Appendix F Current paths to and directories of selected files F-1
Appendix G Current formats for input and output from PDES Testbed tools G-1

References Ref-1

Glossary Glossary-1

Introduction

This guide provides instructions for using the various hardware, software, and service components at the Validation Testing Laboratory (VTL).

The VTL is the physical site of the National PDES Testbed, which was established at the National Institute of Standards and Technology (NIST) to provide testing and validation facilities for the Product Data Exchange using STEP (PDES) activity in support of the Standard for the Exchange of Product Model Data (STEP), a proposed international standard for product data representation and exchange.

The Testbed is funded by the Computer-aided Acquisition and Logistic Support (CALS) program of the Department of Defense.

This guide contains the following information about the VTL:

- logging in and out of the on-site computers
- running some basic UNIX commands
- using some general utilities (editors, e-mail, news)
- doing remote logins and file transfers
- running the basic PDES Toolkit
- using existing utilities for the validation databases
- reporting bugs and getting updates.

It also includes reference and glossary sections.

1 General information

The software components that exist in the laboratory were selected or developed to support the development and testing of the STEP standard [CLARKd] [MITCHELL] [PTI018.01.00]. Using these instructions, you will be able to extract a part model from the geometric modeling systems found in the laboratory, translate the part model into a STEP file, edit the STEP file using a model editor, create a database representation of the STEP file, and query the database using SQL. Additional documentation on specific components is available, as well as on-line instructions.

This guide will tell you how to use the VTL even if you are not an expert in the operation of any of the component systems. It does not, however, provide sufficient instructions to enable you to become an expert.

The VTL software exists in whole or in part on the following computer platforms:

Sun 3/60, 3/80, and SPARCstation (SunOS); DECstation 3100 and 5100 (ULTRIX); IBM RT (IBM AIX); VAXcluster, VAXstations (VMS). The VAX and IBM systems are also used to run selected commercial software (see Appendix D).

The PDES Testbed Hotline is the principal mechanism for reporting problems and getting assistance. The PDES Testbed Hotline is discussed more fully in Section 17.2 and Appendix B.

This section includes information on how to use this document, Testbed activities, computer accounts needed, getting other documentation, using a modem, and the VTL environment.

1.1 User Guide conventions

The following are conventions used in this document to describe the information that must be input by users.

First, although the UNIX system prompt is user definable, the default prompt looks like this:

```
(host)~:
```

where host is the name of the machine on which you are working, and ~ is for the current working directory. This will be used throughout this document to represent the system prompt. Note also that each program that you run can also have a different prompt. For example, if you run Oracle Corporation's SQL*Plus program, the prompt is

```
SQL>
```

Other conventions used in this document follow:

- system and program prompts are shown in normal weight type
- input that is to be typed by the user is shown in **boldface**
- a description of variable input that is to be provided by the user is shown in ***boldface italics***
- optional input is shown within square brackets [ ]
- commands found in text will be shown in **boldface**

In addition, other notations and conventions used in this document for specific keystrokes are:

<return>	the return, enter, or carriage return key
<space>	the space bar
<ctrl>	the control key
<esc>	the escape key

Sometimes combination keystrokes are required; that is, two or more keys must be pressed at the same time. These combinations are represented as individual keys separated by a hyphen (-), such as "<ctrl>-c", which means press the control key and the letter c at the same time as opposed to "<esc> k", which means press the escape key, release it, then press the letter k.

An example:

(host)~:**rlogin brew -l *username*<return>**

means that from the system prompt, the user types in "rlogin brew -l " followed by the user's system user identification and a carriage return.

1.2 Testbed activities

Most of the development and testing activities are being done in the UNIX environment.

Each testing or development project will be assigned specific systems to accomplish its tasks. Your project leader should provide you with the names of the host systems on which you will be working. The PDES Testbed Hotline or the National PDES Testbed Administrator can also assist you. See Appendix B for the Hotline and Testbed Administrator contacts and Appendix C for current host assignments for testing activities. Testers with new activities should contact the Testbed Administrator 30 days prior to use in order to ensure the allocation of appropriate resources and proper computer environment for your needs.

1.3 Accounts

You will need to obtain user accounts for those systems which you will be using. If you need an account or have a problem with your account, contact the PDES Testbed Hotline. (See Section 17.2 and Appendix B for additional information on

the Hotline.)

Throughout this guide, *username* will refer to the name of your account. When asked to type a command that includes *username*, substitute the name of your account. When asked to type *password*, use your password. See Section 1.1 for other notational conventions used in this guide.

1.3.1 Network Information System accounts

Each user of the National PDES Testbed needs a Network Information System (NIS) account (specifically, a cme.nist.gov NIS account). If you do not have an account, please contact the PDES Testbed Hotline (see Section 17.2 and Appendix B for additional information).

The Network File System (NFS), allows your userid, password, and files to be shared by virtually all the computers in the network: Suns, DECstations, IBM RTs and PCs (see Section 1.3.2).

1.3.2 ComputerVision accounts

Selected users will need to use the ComputerVision (CV) systems to develop product design data with the geometric modeler. All users of the geometric modeler on the ComputerVision systems will use the same account. To get the account name and password, see Appendix C or call the PDES Testbed Hotline (see Appendix B).

1.3.3 VAX accounts

You need a VAX account to use the geometric evaluator or the Leverage data modeling software. To get a VAX account, call the PDES Testbed Hotline or the Vax Account Manager (see Appendix B).

1.3.4 Database accounts

If you need to use a validation database in the testbed, you will need a database account. To request an account, call the PDES Testbed Hotline or the Database Administrator (see Appendix B). When you invoke the database software, you will need an account name and password. Your database account name and password will not necessarily be the same as your UNIX or VMS account name and password. Instructions for changing your database password are given in Section 14.2.

1.4 Documentation

Documentation for the National PDES Testbed and the software developed by NIST is available on-line at NIST. Since reference documents are updated frequently as new versions of software are developed, you are encouraged to look in the central document repository for the list of available documentation and the most recently issued versions of the documents. On-line help, known as "man pages", is available for any modules in the PDES, Inc. BPR Toolkit. (See Section 3.8 for help using the "man" command).

Many Product Data Exchange using STEP (PDES) documents written at NIST are written using publishing software that generate PostScript files (generally they have a .ps extension). The PostScript files can be printed on any PostScript printer. The ASCII format for these documents can be supplied on request. They are also available in hardcopy.

Some PDES documents are written using WordPerfect software; these documents will be distributed as files having either a .wp5 or .w51 extension, depending on and referring to the version (either 5.0 or 5.1) of WordPerfect that was used to create the file.

Documents issued in any of the above formats can be retrieved using either File Transfer Protocol (FTP) or Kermit from your local machine. FTP is the faster and more reliable service but it requires an Internet connection. Retrievals using Kermit can be performed over dial-up lines. (See Section 5.2 for more about remote file transfers.) Note that filename length limitations on PCs under DOS will automatically truncate filenames if the name of the file being retrieved is too long and no new name is given. [RESSLER]

If you have a NIS account, PDES documentation can be found in one of the ~pdes subdirectories (see Appendix F).

If you don't have a NIS account, you can remotely login to durer.cme.nist.gov (using telnet or FTP) and entering **anonymous** as your login username. (See Section 5 for more about remote logins.) A set of current documents is kept in one of the directories (see Appendix F).

PDES documentation can also be obtained from the CALS/PDES Project Office at NIST. Contact the Configuration Control Administrator (see Appendix B) for a list of documents available from the Library.

To obtain copies, see the Order and Information Form at the end of this document.

Documentation on the software developed or enhanced by PDES, Inc. is posted on the PDES, Inc. bulletin board. These documents are available to all PDES, Inc. member companies. *The PDES, Inc. Toolkit Systems User Manual* can be found in a BBS File Area in the file "userman.com".

1.5 Using a modem

You do not have to be at NIST to use the National PDES Testbed. All of the Testbed Software and PDES, Inc. Toolkit Software can be used from a remote site with the exception of QDES, an editor for STEP file generation, and the Visualizer. Instructions for dialing in to the National PDES Testbed from a remote location are in Appendix A.

1.6 Validation Testing Laboratory environment

The STEP-based software developed at NIST includes:

- the Fed-X Express Compiler (written in ANSI C, bison, and flex) for parsing an Express file and generating a data dictionary for use by other modules;
- the STEP Parser (written in ANSI C, bison, and flex) for parsing a STEP file into working form;
- the SQL database schema generator (written in ANSI C) for producing SQL-compliant schema definitions used to create a relational database from an Express file;
- the STEP SQL Data Loader (written in ANSI C) for loading a STEP file into database objects;

(All of the ANSI C code described above made use of the GNU gcc compiler. However, the software was designed to be portable and has been compiled and executed with other ANSI C compilers. See Appendix D for current version.)

- the QDES STEP editor, implemented in Smalltalk-80;
- CV Extractor, the CADDS-model extraction and conversion software, written in C;

- the STEP-Plus library, the basis for other STEP application code, used by the IGES to PDES translator, written in C++;

The PDES, Inc. developed software:

- PDES out, to convert data stored in a relational database into a STEP physical file;
- IGES to PDES, to convert an IGES format file into a STEP physical file;
- Old to New, geometry schema update utility, to convert Express Geometry 1B format to Sweden format;
- Visualizer, a 3-D graphic tool for displaying parts

Hardware systems also at NIST include:

- Sun 3 and Sun 4 SPARCstations
- DECstation 3100 workstations
- IBM RT workstations
- DEC VAX Cluster
- DEC Vaxstation 3100 workstations
- Macintosh
- Compaq 386 and IBM PC/AT computers
- ComputerVision CAD workstation

2 Logging in and logging out

2.1 Logging in and starting X Windows

X Windows is a network-based windowing system which allows you to run one or more processes simultaneously. For more information about X Windows, refer to [OREILLY]. For example, you can run database software in one window, edit a document in a second window, and compile a program in a third; each software program could be running on a different computer in the network.

Unfortunately, at this time, you cannot operate the X Window user interface software from a remote site. Attempting to do this will lock-up the host system and cause a disruption of service. Remote X Window service may be available at a later date.

To use X Windows at NIST, log in to one of the Suns located in the Validation Testing Laboratory.

hostname login: ***username*<return>**
password: ***password*<return>**

The system will respond with something like:

Last login: Fri Sep 15 23:10:49 on console

Sun OS Release 4.1 (GENERIC) #1: Tue Mar 6 17:27:17 PST 1990

At this point, if you are using the generic environment and a Sun console at the VTL, X Windows will automatically start up and you will see at least two windows open, one console window and one Emacs editing session. The three mouse buttons will provide access to other windows and functions if pressed while the cursor is located in the background, usually referred to as the root window (the cursor will look like "X" instead of an arrow).

If you are at a Sun console but are not using the generic .login file, you will have to start X windows yourself. Login as above. At this point, instead of X Windows automatically starting, you will get the system prompt:

(host)~:

You can then start X Windows:

(host)~:**startx<return>**

You should become familiar with the following window operations:

- moving windows around your screen
- changing the size of a window
- raising and lowering windows
- converting windows to icons and vice versa
- refreshing the screen
- creating additional xterm windows
- exiting X Windows

Several different X Windows window managers exist and may be used on other systems. In the VTL, the window manager used by X Windows is called terminal window manager (twm). For information about twm, use the Unix **man** command (described in Section 3.8). To access the on-line "man page" for twm window manager, type:

(host)~:**man twm<return>**

2.2 Logging out

To log out of the Sun, first return to the system console by "killing" the window manager. The easiest way to do this is to type **exit** or to simply press **<ctrl>-d** while the mouse pointer is in the console window. Once you are out of X Windows type:

(host)~:**logout<return>**

You will then see the login prompt

hostname login:

3 UNIX

This section illustrates the use of some basic UNIX commands. If you learn these commands, you will be able to do most everything necessary to do your work; however, to become a more proficient and efficient user of the National PDES Testbed, you should also work through an introductory UNIX book [MCGILTON].

UNIX is case-sensitive; the convention is to use lower-case letters for commands and usernames. Lower-case letters are also used for filenames, although this is not required.

3.1 Assigning or changing your password

One of the first things you will want to do when you login to the system the first time is change your password. Your original password will be assigned by the system manager. To protect yourself and the system, your password should be changed to something else as soon as possible. It is strongly recommended that a combination of letters and numbers that do not compose a word or have other significant meaning be used.

To set your password, use the **yppasswd** command:

(host)~:**yppasswd<return>**
Changing yp passwd for username.

The system will then prompt you for your old password:

Old yp password: ***oldpassword*<return>**

Now you can assign or change your password:

New password: ***newpassword*<return>**
Retype new password: ***newpassword*<return>**
Yellow pages passwd changed on host.

If you forget your password, please call the PDES Testbed Hotline (see Appendix B).

3.2 Directories

The UNIX file system is hierarchical; files that contain other files are called directories. To access a particular directory or file, you must know where it is within the file system. Its location can be specified in absolute terms (that is, with respect to the root, or highest level, of the file system) or in relative terms (that is, with respect to the location of the directory in which you are currently working). The "map" for getting from one location within the file system to another is called a path.

The root of the file system is referred to by the slash character (/). Therefore, an absolute path to a file would always begin with /. The slash (not back slash (\) as with DOS) is also used to separate the names of directories within the path. For example, the path /usr/local/bin/dial refers to a file named dial in the directory named bin; bin is a subdirectory of local and local is a subdirectory of usr.

The current directory is indicated by a single period (.); its parent directory is indicated by two periods (..). These characters can be used to specify a relative path name to a file. In the example above, local is the parent directory of bin. Local is also the parent of a directory named etc. Suppose you are working in the bin directory. To get to a file named bday within the etc directory, you can use the absolute path /usr/local/etc/bday or the relative path ../etc/bday.

3.2.1 Changing directories

To change directories, use the **cd** command:

(host)~:**cd** ***pathname/directoryname*<return>**

where ***directoryname*** is the name of the directory to which you want to change and ***pathname*** is either the absolute path or the path relative to the current directory. Thereafter, the system prompt will include the name of the new directory in place of the ~ symbol. For example,

(host)~:**cd /usr/oracle<return>**

(host)/usr/oracle:

The symbol ~ (tilde), by itself, is used to refer to your home directory. To return to your home directory from any other directory, type:

(host)/usr/oracle:**cd<return>**

(host)~:

To get to ***username***'s home directory, type:

(host)~:**cd ~*username*<return>**

(host)*pathname/username*:

To get to a directory relative to a user's home directory type

(host)~:**cd ~*username*/[*pathname*/]*directoryname*<return>**

3.2.2 Creating directories

To create a new directory, use the **mkdir** (make directory) command:

(host)~:**mkdir *directoryname*<return>**

where ***directoryname*** is the name of the new directory you are creating. The directory will be created as a subdirectory of the one in which you are currently working.

3.2.3 Removing directories

To remove a directory, use the **rmdir** (remove directory) command:

(host)~:**rmdir *pathname/directoryname*<return>**

To remove a directory and its contents, use the **-r** option with **rmdir**:

(host)~:**rmdir -r *pathname/directoryname*<return>**

WARNING! In general, be careful when you use rmdir; you could accidentally wipe out your entire directory structure depending on the set of options used. In particular, the rmdir command with the -r option is very powerful, as ALL subdirectories and files under *directoryname* will also be

removed.

3.3 Files

3.3.1 File permissions

When a file is created it is assigned an owner and a group. Initially, the creator of the file is the owner and the creator's default group (a user can belong to more than one group) determines the group ownership. The owner of a file can change the ownership using the **chown** command and change the group the file belongs to by using the **chgrp** command.

(host)~:**chown** ***newownername filename*<return>**

(host)~:**chgrp** ***newgroupname filename*<return>**

The system administrator can set up groups and add or remove users from groups.

Associated with each file is a set of permissions indicating who is allowed to read, write, and execute it. Permissions are assigned for three sets of users: owner, group, and world. Every user of the system is included in the world. By assigning the proper permissions for each file, you can determine who gets to use your work. Use the UNIX command **chmod** to change the permissions associated with your files. Consult the "man pages" (described in Section 3.8) for instructions on using **chmod.**

3.3.2 Listing file names

To see which directory you are in, type **pwd** (which stands for print working directory):

(host)~:**pwd<return>**

To list the files in the current directory, use the **ls** command:

(host)~:**ls<return>**

If you want to list all files, including hidden (dot) files, type:

(host)~:**ls -a<return>**

You can get more information about files and the permissions associated with them by adding the **-l** option:

(host)~:**ls -al<return>**

Group ownership of a file can be displayed using the **-g** option:

(host)~:**ls -lg<return>**

3.3.3 Copying files

To copy a file, use the **cp** command:

(host)~:**cp *source-file destination-file*<return>**

where the ***source-file*** is the name of the file you want to copy and ***destination-file*** is the name of the file to which you are copying.

To copy to and from different directories, include the path to the target directory in the command:

(host)~:**cp *source-path/source-file dest-path/dest-file*<return>**

To copy files from one machine to another, use the **rcp** command. For example, to copy a file from your current directory to a directory on another machine, type:

(host)~:**rcp *source-file remote-machine:dest-path/dest-file* <return>**

where the ***remote-machine*** is the name of the machine to which you are copying. Section 5 describes some alternative methods of accessing data on other machines.

3.3.4 Moving files

The **mv** command is used to move files within or among directories; the method of specifying source and destination directories is identical to that used with the **cp** command:

(host)~:**mv *source-path/source-file dest-path/dest-file*<return>**

3.3.5 Displaying files with "more"

You can use the **more** command to display a file one screen-full at a time:

(host)~:**more** ***filename*****<return>**

A message at the bottom of the screen will indicate what percentage of the file has been displayed. To move through the file one line at a time, press **<return>**; to display the next page of the file, press **<space>**. To see options, press **h**. To exit the file, press **q**.

3.3.6 Searching files

3.3.6.1 Using "more"

After you have displayed part of a file with the **more** command, you can search the remainder of the file for a specific string by typing / followed by the string you want to find.

--More--(19%)/

When you press the / key, the prompt will change, and you can then enter the string you want to find. For example, to find the word "format", type:

/format<return>

When an occurrence of the word "format" is found, the page containing it will be displayed. (The word will not be highlighted, so you have to look for it. However, it can usually be found on the fourth line from the top of the page.)

To continue searching for subsequent occurrences of the same string, type / again.

3.3.6.2 Using "grep"

"Global regular expression print", **grep**, returns a single line containing the specified string. To use grep:

(host)~:**grep "*searchstring*" *filename*<return>**

where ***searchstring*** is the string you want to find and ***filename*** is the file in which you want to look. Note that the quotation marks are optional if the search string

does not contain spaces.

3.3.7 Removing files

To remove a file, use the **rm** command:

(host)~:**rm** ***filename*<return>**

3.4 Creating command aliases

If you find yourself using the same series of commands over and over again, or typing long strings of parameters for particular commands, you may want to create some command aliases; these "nicknames" for command strings are stored in a file and can be typed in lieu of the associated strings. Aliases are stored in a file called ".aliases" in your home directory.

An alias file might contain the following string:

alias l 'ls -al'

which specifies that the command string 'ls -al' be executed when the letter l is typed.

3.5 Printing and plotting

To print out a file, use the **lpr** command. Instructions for using **lpr** can be found in the "man pages" (see Section 3.8 for a discussion of the "man pages"). A typical print command using **lpr** would look like this:

(host)~:**lpr** ***filename*<return>**

You can also print out files in fancier formats using the **enscript** command. Instructions for using **enscript** can also be found in the "man pages" (see Section 3.8).

Here is a convenient alias to include in your .aliases file:

alias prcode 'enscript -2rGfCourier6'

Using **prcode**, created above, files are printed in two columns per page using small (6-point) Courier font. The print is rotated ninety degrees on the page; a

heading showing the date, time, page, and filename is included on each page. To use the **prcode** command, type **prcode** followed by the filename:

(host)~:**prcode** ***filename*****<return>**

There are Macintosh and IBM RT computers available for plotting at NIST. For help in locating and using the Macintosh, please contact the PDES Testbed Hotline (see Appendix B).

3.6 Standard input and output

There are three standard UNIX I/O devices: stdin ("standard input" - the default is the keyboard), stdout ("standard output" - the default is the screen), and stderr ("standard error" - the default is also the screen). These standard I/O devices can be redefined using command line arguments.

For a more complete discussion of command line arguments, consult the "man page" for the **csh** command. (See Section 3.8 for a discussion of "man pages").

3.7 Shell scripts

The UNIX shell is the program that interprets shell commands (such as those you type at the keyboard). The shell does not necessarily have to take its input from the keyboard; it can also interpret commands stored in a file. A sequence of shell commands stored in a file is referred to as a shell script.

Numerous examples of shell scripts can be found in almost any UNIX book.

3.8 Getting help - the "man pages"

If you need help, you can use the **man** command to display pages from the UNIX Programmer's Manual. "Man pages" are available for all modules in the PDES Toolkit. For example, to get information about the use of the **cp** (copy) command, type:

(host)~:**man cp<return>**

If you know what kind of operation you want to perform, but you are not sure what command you need, you can use the **-k** option to get a list of appropriate commands:

(host)~:**man -k copy<return>**

You can even get help on **man** by typing:

(host)~:**man man<return>**

4 General utilities

4.1 Editors

4.1.1 The default editor, vi

The longstanding UNIX editor is vi. It will not be discussed here except to tell you how to end a vi session without changing anything. If you are interested in learning vi, it is discussed in virtually every UNIX reference available, one of which is referenced at the end of this document [MCGILTON].

If you inadvertently find yourself in a vi session type

<esc>:

The cursor will appear at the bottom of the window with a colon, ":" preceding it. Then type

q!

The editing session will end with no changes being made.

4.1.2 Emacs

The Emacs editor is an extensible full screen text editor. Like UNIX, it is a tool with which you must be familiar before you can accomplish anything useful in the Validation Testing Laboratory. An on-line Emacs tutorial is available on the Suns. To avoid becoming frustrated, you should learn to perform the following basic Emacs operations with confidence. The convention used in this discussion is that the C stands for the control key and M stands for the meta key (or Shift-ESC). The meta key is sometimes labeled with a diamond.

- moving forward or backward one character using C-f or C-b (or arrow keys)
- moving forward or backward one word using M-f or M-b (or arrow keys)
- moving forward or backward one line using C-n or C-p (or arrow keys)
- deleting a character forward or backward using C-d or Delete

- loading a file into Emacs using C-x C-v or C-x 4 f
- saving a file using C-x s
- using the Emacs mail facility

If you are using the generic environment, an Emacs editing session will automatically start up as one of the X Windows when you log in. If you don't use the generic login or your Emacs session does not already exist for some reason, you can start an Emacs session by typing

(host)~:**emacs &**

or by choosing "emacs" from the middle mouse button menu (see Section 2.1). (Note that the & means "start the process in background").

To run the Emacs tutorial, position the cursor in the Emacs window, and type:

C-h t

In addition to the basic editing capabilities that are demonstrated in the tutorial, Emacs can be used for tasks such as managing mail or reading news. The Lisp-like language in which Emacs is written can be used to create new editor applications, such as debuggers or context sensitive editors.

For a more exhaustive discussion of Emacs see [STALLMAN].

4.2 Mail

Section 4.2.3 describes how to use Unix mail. More detailed instructions can be found in the Unix manuals. Alternatively, you may send mail using the Emacs tool; instructions for using Emacs mail are given in Section 4.2.4. Emacs mail provides an interface to UNIX mail, so you may prefer to simply use Emacs and ignore the UNIX mail instructions.

4.2.1 Mailing lists

The network already has many mailing lists available for use. To get a list of the groups type

(host)~:**maillist<return>**

A list of groups will be displayed.

To see the individual members of a group type

(host)~:**maillist** ***mailgroup*<return>**

where ***mailgroup*** is one of the groups returned by the earlier **maillist** command (given without an argument).

All PDES, Inc. users have mail aliases to forward mail to their MCI accounts or Internet address. Many PDES, Inc. users have their mail sent to one of the VTL computers. Then, using a file named .forward, located in their home directory, the mail is forwarded to a specified address. The .forward file contains a single line that is the address of the final destination of the mail.

The prefix pd- is added to the user names in order to minimize name space conflicts with any NIST personnel. For example, to send mail to PDES, Inc. member J. Doe, the mailing address would be pd-doe.

4.2.2 Addressing mail using aliases

You can also put aliases in a customized ".mailrc" file so that you can send copies to several people without having to type their names every time. For example, if you frequently send the same mail to, for example, two different addresses, "addr01" and "addr02", include an alias in .mailrc:

alias both_addr addr01@cme.nist.gov addr02@cme.nist.gov

You can then use the alias "both_addr" when specifying the recipients of your letters and both "addr01" and "addr02" will receive the mail. (Contact the PDES Testbed Administrator (see Appendix B) for a copy of mailing lists that have been set up for PDES, Inc..)

4.2.3 UNIX mail

4.2.3.1 Sending UNIX mail

To send mail to someone, use the mail command:

(host)~:**mail -v** ***recipient*<return>**

where ***recipient*** is the name and address of the person, or persons, to whom you are sending your letter. If the recipient is on the same machine or local network as you, you don't need to specify an address. For example, to send mail to yourself, you would type your username in place of recipient. To send mail to the PDES Testbed Hotline at NIST, you would specify the name and address:

(host)~:**mail -v hotline@cme.nist.gov<return>**

Mail will prompt you for the subject of your letter:

Subject:**Comments about Testbed<return>**

After typing the subject, you can enter the text of your letter. After typing the message, press **C-d** to send the letter. For example,

(host)~:**mail -v hotline@cme.nist.gov<return>**

Subject: Comments about Testbed<return>

I just wanted to let you know that the VTL User's Guide is the most lucid and concise document I have seen in my twenty years in the business.<return>

C-d

The parameter **-v** means verbose; when you use it, mail will be sent with the verbose option, which gives you some feedback about the status of your letter. For example, if you are sending mail to someone at a remote location, and the verbose option is set, the mail facility will tell you whether you were successfully connected to the remote machine and whether your message was delivered. If you don't want this information, don't use **-v**.

If, after typing your letter, you decide you don't want to send it, type **C-c C-c** to abort the mail facility. Your letter will be stored in a file called dead.letter; you can edit it later if you wish.

4.2.3.2 Inserting files

Instead of typing your letter directly, you can insert the contents of a file that you have previously created and edited. Using Emacs, vi, or any editor you prefer, compose a message and save it in a file. Then, after using the **mail** command to

initiate a letter and enter the subject, type

~r *filename*

to insert the contents of a message file named "memo1.pdes":

(host)~:**mail hotline@cme.nist.gov<return>**

Subject:**Comments about Testbed<return>**
~r memo1.pdes<return>

The contents of the file memo1.pdes will appear in your letter. You can type in additional text before or after inserting files. For example, you can type in some introductory comments, insert a file, type some more sentences, insert another file, and type closing comments before pressing **C-d** to send the message.

You can also use the **~r** command to insert your signature file. A signature file contains information that you want to include in all your outgoing messages. By storing this information in a signature file, you can avoid having to retype it every time you compose another message. (When you use UNIX mail, you can name your signature file anything you want, and then insert it at the end of your message, just as you would insert any other file. However, when you use the Emacs mail facility described in Section 4.2.4, your signature file is expected to be named ".signature", and so for consistency, it may be a good idea to use that name for UNIX mail also.)

4.2.3.3 Sending copies

Before you send your letter, you can specify that a copy be sent to another user by typing **~c** followed by the recipient's name and address. If you want to send copies to several recipients, type **~c** once, then type each recipient's name followed by a space. For example, to send copies to two recipients, "address_1" and "address_2",

(host)~:**mail hotline@cme.nist.gov<return>**

Subject:Comments about Testbed<return>

I just wanted to let you know that the VTL User's Guide is the most lucid and concise document I have seen in my twenty years in the business.<return>

~c address_1 address_2 <return>

You can modify your .mailrc file so that you are automatically prompted for the list of users to whom you want to send copies. Either using the Defaults editor (see your Sun manuals) or another editor, include the command **set askcc** in your .mailrc file.

4.2.3.4 Confirming a letter

To look over your letter before you send it, type **~p** to display the contents of the letter. First the letter, and then (continue) will be displayed, indicating that you can continue using the mail facility.

4.2.4 Emacs mail

To run Emacs mail, you have to be in Emacs. Make sure you have a directory named "Mail" in your account. More detailed information about the Emacs mail facility can be found in the Emacs manual. Again, the convention used in this discussion is that the C stands for the control key and M stands for the meta key (or Shift-ESC). The meta key is sometimes labeled with a diamond.

4.2.4.1 Sending Emacs mail

To send mail in Emacs, first select and initialize the *mail* buffer by typing:

C-x m

In your mail buffer, you will see header fields followed by a delimiter message. For example,

To:

CC:

BCC:

FCC:

Subject:

--Text follows this line--

The 'To:' field is used to specify the name(s) and address(es) of the recipient(s) of the letter. The names and addresses of recipients of copies are stored in the 'CC:' field. The 'BCC:' field is used to specify recipients who should receive copies, but whose names should not appear in the header of the delivered message.

The 'FCC:' field is used to store the name of a file to which a copy of the message should be appended. This provides a convenient way to keep track of all the messages you have sent.

The subject of your letter should be typed in the 'Subject:' field.

Your mail buffer might not contain the same header fields as those in the example above; however, you can add any field by simply typing its name and contents above the delimiter line. For example, if your headers include 'To:' but do not include 'CC:' you can type it in yourself:

To:hotline@cme.nist.gov<return>

CC: ***address_1***

(There are also commands that can be used to create the missing fields. These are included in the list of commands shown below.)

Use the arrow keys to position the cursor within each field to enter the header information. After filling in the header fields, use the arrow keys again to position the cursor below the delimiter line to enter your message.

When your message is complete, you can send it by using either C-c C-s (to send the message and keep the *mail* buffer) or C-c C-c (to send the message and select another buffer). Here are some commonly-used Emacs mail commands:

C-c C-s	send the message; keep the *mail* buffer selected
C-c C-c	send the message; select another buffer
C-c C-f C-t	move to the 'To' header field, creating one if there is none
C-c C-f C-s	move to the 'Subject' field, creating one if there is none
C-c C-f C-c	move to the 'CC' header field, creating one if there is none.

C-c C-w insert the field ~/.signature at the end of the message

(When you use Emacs mail, your signature file must be named ".signature")

To exit the mail buffer, type **C-x k**.

4.2.4.2 Reading Emacs mail

To read and dispose of mail within Emacs, type M-x vm<return> (vm stands for view mail).

Mail messages are stored in folders. Use the following commands to peruse the contents of your mail folder(s). Note that as with all Emacs modes, vm will display a listing of mode keybindings if you type **C-h m**.

h summarize contents of current folder

t display hidden headers

g get any new mail that has arrived

v visit another mail folder

Vm displays one message at a time, in order of receipt. The number of the currently-displayed message is shown on the mode line at the bottom of the window. To display another message, type the message number followed by <return>.

If the current message does not fit on the screen, you can scroll through it with the following commands:

<space> scroll forward (up) a page

b scroll backward (down) a page

You can mark for deletion the messages you don't want to keep; you can then un-delete them to get them back. To get rid of a message forever, you must first delete it, then expunge it or quit vm. The following commands are used for deleting, un-deleting, and expunging.

d mark the current message for deletion

u	un-delete the current message
k	mark for deletion all messages with the same subject as current message
#	expunge deleted messages

The following commands are used to scroll amongst your messages, delete messages, or copy messages to files:

n	move to the next non-deleted message
p	move to the previous non-deleted message
N	move to the next message, whether it is deleted or not
P	move to the previous message, deleted or not
M-n	move to next unread message
M-p	move to previous unread message
<tab>	move to last message seen
M-s	incremental search through the folder
j<return>	move to message number j
>	move to end of current message

You can exit vm with or without saving your messages. These commands are used to exit vm:

s	save current message in a new folder, or append to an existing folder
w	write current message to a new folder, or append to an existing folder
S	save entire folder to disk, expunging deleted messages

q	quit vm, expunging deleted messages and saving folder.
x	exit vm without saving folder

4.2.4.3 Sending replies

Within vm, if you want to send a reply to the current message, type the letter r. A buffer in another window will be displayed, with the header fields appropriately filled in. You can then compose and send your reply as described in Section 4.2.4.1.

r	reply to sender of message
R	reply, including text from current message
f	follow-up: reply to all recipients of message
F	follow-up, including text from current message
z	forward the current message

To compose a new message while in vm, use the same commands described in Section 4.2.4.1:

m	compose a message (equivalent to mail command described in Section 4.2.4.1)
C-c C-s	send message to recipient; retain current buffer
C-c C-c	send message to recipient; select new buffer

4.3 News

The news facility in Emacs is called gnus. To use gnus from Emacs, from within an Emacs window type:

M-x gnus<return>

or

M-n<return>

Once you are in gnus, type **C-h m** to get instructions for using gnus. The most often used commands are:

- <space> move to next article, newsgroup, or page
- n move to next unread article
- p move to previous unread article
- N *or* C-n move to next article (including previously read articles)
- P *or* C-p move to previous article (including previously read articles)
- d delete article from reading list, but save for future reference
- u un-delete and article or un-subscribe to a news group
- / search for keyword (you can also use C-s)
- q quit gnus

There are over 700 different news categories available on Internet and at NIST. About two megabytes of data are posted to the system each day. To reduce the number of news categories that are available to you, you can un-subscribe to those you consider inappropriate, uninformative, or boring. Un-subscribing can be done interactively while gnus is running.

Users of the generic environment have a .newsrc file pre-installed with an edited set of about thirty news categories that may be of interest to PDES users. Alternatively, you can copy the default news category file (see Appendix F) to ~/.newsrc (your own home directory). You can customize your .newsrc file by subscribing or un-subscribing from the various news categories that interest you. The gnus software will make the appropriate entries in your .newsrc file.

4.4 Configuration management

Revision and configuration control can be accomplished via an interim Configuration Management System (CMS) consisting of a set of procedures for users to follow and the Revision Control System (RCS).

The PDES toolkit and related software have already been placed under the interim CMS. [BODARKY] [KATZ]

You can also control your own files using this interim CMS.

Start by creating a subdirectory, called RCS, under the directory you want to control.

(host)~:**mkdir RCS<return>**

You may then "check in" files using the "ci" command:

(host)~:**ci *filename*<return>**

After a file is "checked in", copies may be "checked out" using the "co" command:

(host)~:**co *filename*<return>**

Using "co" without any options will not allow you to make any changes in the file. If you want to make changes, the file archive must be "locked":

(host)~:**co -l *filename*<return>**

When a file is checked back in to a "locked" archive, a new revision is added to the archive.

A list of revisions to a file under RCS can be obtained by typing:

(host)~:**rlog *filename*<return>**

5 Remote login and file transfers

5.1 Remote login to other systems

To login to another UNIX system on the network, type:

(host)~:**rlogin** ***hostname*****<return>**

where ***hostname*** is the name of the remote computer.

To login to a non-UNIX system (such as the VAX), use the telnet command:

(host)~:**telnet** ***hostname*****<return>**

After you login using either **rlogin** or **telnet**, the remote system will then request your username and password, as described in Section 1.3.

5.2 File transfers

5.2.1 FTP file transfers

The FTP file transfer utilities allow you to transfer files to and from other systems on the network. You must know the username and password of a valid account on the remote system. To start FTP type

(host)~:**ftp** ***hostname*****<return>**

You will be prompted for the username and password. After entering them, you will see the FTP prompt, generally

ftp>

The basic FTP operations (all entered at the FTP prompt) with which you should become familiar are:

- help display FTP commands
- get transfer a file from another system to your system
- put transfer a file from your system to another system

- mget transfer multiple files from another system to yours
- mput transfer multiple files from your system to another
- prompt toggle prompt on/off for each file transfer
- cd change directories on the remote system
- lcd change directories on your system
- binary sets transfer mode to binary

You can also transfer files to and from NIST over Internet by specifying the address of the host machine:

(host)~:**ftp *hostname*.cme.nist.gov<return>**

To change directories on the remote machine, use that machine's syntax. For example, a command to change directories on the VAX would look like this:

set default [*DIRECTORY.SUBDIR-1.SUBDIR-2*]

FTP can also be used to transfer files to and from PCs. (In most cases, to transfer files between a PC and another machine, the PC must be the local machine).

5.2.2 Kermit file transfers

Kermit file transfer utilities are available on the Sun and on PCs in the VTL.

Kermit on the Suns is invoked by typing:

(host)~:**kermit<return>**

You will then see the Kermit prompt

C-Kermit>

You then will need to set up the connection protocol before actually making a connection:

C-Kermit>**set line /dev/ttya<return>**
C-Kermit>**set baud 9600<return>**
C-Kermit>**connect<return>**

Invocation and operation on a PC is only slightly different:

PC-PROMPT>**kermit<return>**

The Kermit prompt will be displayed. Make the following entries:

Kermit-MS>**set port com1<return>**
Kermit-MS>**set baud 9600<return>**
Kermit-MS>**connect<return>**

At this point, more information on uploading and downloading using Kermit can be obtained by typing **help**. The "man pages" can also be used to get more information on the use of Kermit (see Section 3.8).

To exit Kermit cleanly, make sure you are in local Kermit, then exit by:

Kermit-MS>**hangup**
Kermit-MS>**exit**

6 The ComputerVision/CADDS geometric modeler

Geometric models can be created and edited using the CADDS package on a ComputerVision (CV) system. If you already know how to use the CV system, you can create a new geometric model or edit a previously created model. If you don't know how to use the system, you probably shouldn't attempt these operations. However, you can retrieve an existing model, as described below.

The locations of the CV systems are listed in Appendix E.

6.1 Logging into the ComputerVision system

To log in to the system:

login:***username***<**return**>

password:***password***<**return**>

Starting the window system, please wait...

Where the specific ***username*** and ***password*** used to log in can be obtained by calling the PDES Hotline (see Appendix B).

6.2 Starting the CADDS software

The CADDS software runs in the SunView window environment. With the mouse cursor in the background, or "root window" area of the screen, press and hold the right mouse button to pull down the SunView menu and select CADDS from the menu. A window labeled CADDS will open and the CADDS software will load and start.

After the CADDS software has started, four additional windows will be opened: the CADDS 4X status window, text window, graphics window (which partially covers the status and text windows), and drawing window. Also, a CADDS 4X menu labeled SOLIDESIGN will be displayed along the right side of the screen.

You can bring any window to the front by clicking on its border with the left mouse button.

Bring the text window to the front. (The text window prompt is #13#.)

Note that the CV system is NOT case sensitive; you can type commands in either upper, lower, or mixed case. The commands that you enter are parsed as they are entered and the first "n" number of characters needed to produce a unique command are shown in upper case, the rest in lower case. For example typing **list** will produce **LISt** on the screen. In this section, the user input is shown as it will appear to the user on the screen, already parsed, and does not mean that the user must use upper and lower case in that exact manner.

A test part (pipeclamp) has been developed for demonstration purposes and will be used throughout the following examples.

6.3 Displaying a geometric model

6.3.1 Retrieving an existing model

Using the left mouse button, pull down a menu from the file drawer icon (left-most icon in top row on the right of menu) and select:

Activate Part (partname)

Enter the partname in response to the prompt in the text window:

ACTivate PARt->**pipeclamp<return><return>**

(Note that for many CV operations, you must hit the <return> key twice.)

You will see the following message in the text window:

Activating OLD part.
#13#

Using the left mouse button, pull down a menu from the drawings icon (middle icon on top row) and select:

List all Drawings

In the text window, you will see:

#13#LISt DRAwing ALL

PART NAME=PIPECLAMP

Drawing: LAYOUT

#13#

There is only one drawing associated with the PIPECLAMP part. To display this drawing, pull down the menu again from the drawings icon and select:

Activate (drawing)

At the prompt, enter the name of the drawing you want to see (since there is only one, you don't have a choice):

#13#ACTivate DRAwing->**LAYOUT<return><return>**

Four views of the part will be displayed in the graphics window. (You may want to hide the text window to see the entire graphics window.)

6.3.2 Creating or editing a geometric model

Don't bother to attempt this if you are untrained in the use of the CADDS system. For further information, consult a trained user or the CADDS station User Guides.

6.4 Creating the AMRF part model file

First, make sure that the active part has been loaded from the appropriate directory (see Appendix F); otherwise, part model extraction will not be able to write the part model file.

Bring the CADDS text window to the front and type the following command:

#13#USE VNTABLE NEWVNT<return>

The system will respond

Entering Environment: NEWVNT!!

At the prompt, type

#13#**EXTRACT AMRFPM:** (Don't press <return> yet!)

The system will respond

Pick Solid MODEL ent

Using the left mouse button, click on the part view in the upper-right of the graphics window, then press **<return>**.

After a few seconds, you will see several messages describing the part extracted, then

XTRACTPM:Internal data structure completed

XTRACTPM:Writing PM output file

/usr4/pdesinc/parts/pipeclamp/pipeclamp.pm

XTRACTPM:Part Model Extraction Completed.

#13#

The AMRF part model for the pipeclamp part is now stored in the parts directory in the file pipeclamp/pipeclamp.pm.

When you are finished using CADDS, use the left mouse button, pull down the menu from the EXIT icon (left icon in bottom row) and select

Quit & Exit CADDS

You will get two separate warning messages, one for "Quit" and one for "Exit". In response to each, select "confirm" using the left mouse button.

When you are finished using the CV system, use the right mouse button and select "Exit SunView" from the SunView menu, then logout.

7 The CV-to-STEP file translator

The NIST CV-to-STEP file translator will convert files in AMRF part model format, as created by the geometric modeler (see Section 6), into STEP physical file format.

7.1 Converting to STEP file format

You will need to convert the extracted file from AMRF part model format to STEP file format. Currently, this software generates STEP files which conform to a specific version of IPIM, see Appendix G. To execute the program, type:

(host)~:**pm2step** ***partfilename*.pm<return>**

The CV-to-STEP file translator will then convert the AMRF part model file, ***partfilename***, to STEP format and store it in a file named step.out. You should rename step.out using "step" as an extension.

(host)~:**mv step.out** ***partfilename*.step<return>**

7.2 Format conformance

Presently, **pm2step** produces entities which conform to an older version of the STEP geometry and topology models (Part 42). To convert to the most recent edition of Part 42, a utility called **oton** (Old to New) has been created (see Section 16.1 and Appendix G).

(host)~:**oton** ***inputfilename outputfilename*<return>**

Consult the "man pages" (see Section 3.8 on using "man pages") for further information on using **oton**.

8 The IBM RT/CATIA geometric modeler

The CATIA software allows you to create, edit, and display geometric models. The CATIA software is only available on the IBM RT systems (see Appendix E for locations).

8.1 Logging into the IBM RT system

The RT systems use the typical login:

login:***username***<**return**>

password:***password***<**return**>

The specific ***username*** and ***password*** used to log in can be obtained by calling the PDES Hotline (see Appendix B).

8.2 Starting the CATIA software

If you use the username and password given by the PDES Hotline, the CATIA processor will automatically start when you log in. (Using other usernames and passwords will not work in this manner). The CATIA screen on the graphics display will appear.

Holding down the alt-key and pressing the right-shift-key will toggle keyboard control between the left- and right-hand displays. Change keyboard control to the graphics display. The mouse should be located on the digitizer pad. Press mouse button 1 and select "COLD START" from the menu in the upper-right-hand corner. This will open the CATIA session and place you at the FILE menu.

8.3 Loading an existing model

At the FILE menu, you can either enter the filename or press the "Enter/Action" key to see a list of the model files available. If choosing from the list, use mouse button 1 to select the appropriate file.

Once you have selected the model file, CATIA will display the model READ menu. Again, you can either type the name of the model or press the "Enter/Action" key to see a list of the models. If choosing from the list, use mouse button 1 to select the appropriate model.

8.4 Plotting

To plot, you must select the PLOT function. This can be done in one of two ways. You can use the lighted function keys by pressing the second key of the third row of lighted program function keys; or you can use the mouse by moving the mouse cursor to the right side of the screen where the functions are displayed; press and hold mouse button 2; slide the mouse down until the underlying, hidden menu appears; select PLOT and release the button.

8.4.1 Define a "sheet"

After selecting PLOT, you must define a sheet. If the sheet is a new sheet, simply type the name you want to use. YOU MUST REMEMBER THE NAME OF THE SHEET. Once the sheet name has been entered, CATIA will switch to the MODIFY function.

8.4.2 Define a "window"

The next step is to define a window to plot. Position the mouse cursor in the upper-left-hand corner of the area to be plotted and press mouse button 2. Move the mouse cursor to the lower-right-hand corner of the area to be used for the window and press mouse button 2 again. The window is now defined. If the window is not the right size or shape, you can modify it using the FRAME function.

8.4.3 Set the "scale"

Once the window is defined, you should select the SCALE function. A subfunction menu will appear in the upper right section of the screen. Use the left mouse button and select SCALE LINKED. Select an element in the model (for example, a line) to define the units. Then you must type the output plot scale (for example, 1.0 or 0.5).

8.4.4 Finish the plot(s)

Continue as above to define as many plot sheets as you like. When you are done, you must exit CATIA to finish the plotting process.

The plotter should be turned on. Load the paper:

1. raise the plotter cover
2. pull the lever near the right paper guide towards you to open the carriage
3. slide the left paper guide all the way to the left
4. insert a sheet of paper
5. align the paper with the right side paper guide and with the white strip toward the rear of the plotter
6. slide the left paper guide over so that it is aligned with the left side of the paper (**DO NOT MAKE IT TOO TIGHT TO THE EDGE**)
7. push the lever on the right side away from you to lock the paper in place
8. close the plotter cover

To finish plotting, at the system prompt, type the command **plotout**:

(host)~:**plotout<return>**

At the "sheet name" prompt, enter the name of the sheet and press Enter/Action. Continue to press Enter/Action until you see the message "execution in progress". When the program completes, the plotter will draw the plot.

8.5 Converting files from one format to another

There is a CATIA utility that will allow conversion from IGES to CATIA and from CATIA to IGES via an intermediate IIF format.

At the system prompt on the text display type

CATIA>**catutil<return>**

A menu of commands will be displayed. Use the down-arrow key to move the cursor to CATIA Interface Utilities. Press the "Enter/Action" key. A new menu will appear. The new menu will have several options on it, including IGECAT (IGES to CATIA) and CATIGE (CATIA to IGES).

8.5.1 Converting an IGES file to a CATIA model

First, copy your IGES file into the /u/catia directory on the host machine.

The conversion is a two step process: first, the file is converted from IGES to IIF; second, the IIF file is converted to a CATIA model (using **catutil**).

To convert from IGES to IIF:

CATIA>**igesp -i** ***IGES_filename*****<return>**

where ***IGES_filename*** is the name of the IGES file you are converting. **igesp** assumes the extension .iges, and it should not be included when specifying ***IGES_filename***. The output file has the form ***IGES_filename*****.iif**.

Next, run the **catutil** program. Select the first option, IGECAT. Another menu will appear. In the IIF filename section of the menu, enter the name of the IIF file you created above. Enter the name you want for the CATIA model in the CATIA section of the menu. All other options can use their default settings. The REPLACE option in the "to CATIA" section should be set to "NO" unless you want to overwrite an existing model. Press the Enter/Action key until the message "execution in progress" is displayed. When the program completes, the new CATIA model will have been created.

8.5.2 Converting a CATIA model to an IGES file

Simply reverse the above process to convert CATIA to IGES. Using **catutil**, select CATIGE and follow the instructions to create an IGES file.

8.6 Starting CATIA without re-logging on

To re-start CATIA from the system prompt:

CATIA>**catini<return>**

9 The IGES to PDES translator

9.1 Introduction

The IGES to PDES translator is a software application which permits the translation of IGES entities to their equivalent STEP counterparts. The output of the translator is a physical file adhering to the STEP standard for the exchange of product data [SILVERNALE]. The IGES to PDES translator was developed by PDES, Inc..

The IGES to PDES translator was developed using GNU C++ compiler. The only platforms on which the translator can execute are a Sun and a DECstation. The IGES to PDES translator outputs a physical file containing equivalent STEP entities for a subset of IGES entities.

9.2 IGES to PDES translator mapping

The IGES to PDES translator currently supports the following IGES entities which are mapped to their corresponding STEP entities as indicated:

IGES ENTITIES		STEP ENTITIES
100	Circular Arc	Circle, Trimmed Curve
104	Form 1, Conic Arc - Ellipse	Ellipse
104	Form 2, Conic Arc - Hyperbola	Hyperbola
104	Form 3, Conic Arc - Parabola	Parabola
106	Form 11-13, Copious Data	Polyline
108	Form 0, Plane	Plane
110	Line	Polyline
112	Parametric Spline Curve	B-Spline Curve

114	Parametric Spline Surface	B-Spline Surface
116	Point	Point
124	Form 0-1, Transformation Matrix	Transformation Matrix
126	B-Spline Curve	B-Spline Curve
128	B-Spline Surface	B-Spline Surface

All unsupported IGES entities are currently skipped and errors within the IGES input file will usually cause the translator to abort. Extensive error handling has not been implemented into the translator but is planned once error exception handling is implemented in the C++ language.

See [FUHR] for a discussion of the mathematical routines.

9.3 IGES to PDES translator execution

In order to execute the IGES to PDES translator, you will need to redirect input (using the < symbol) from an IGES input file, and redirect output (using the > symbol) to a PDES (STEP) output file. To execute the program at the prompt type:

(host)~:**itop** ***<iges_filename*** ***>pdes_filename*****<return>**

The file named ***pdes_filename*** will contain the STEP physical file translation of the IGES input file, ***iges_filename***.

10 Compiling Express using Fed-X

Fed-X is the Express language information model compiler [CLARKd]. The basic compiler is executed by giving the command **fedex**. There are also three other programs based on the Fed-X compiler, each generating a different type of report based on the compilation: Fed-X-flatten (invoked by **fedex_flatten**), Fed-X-SQL (invoked by **fedex_sql**), Fed-X-QDES (invoked by **fedex_qdes**).

If you want to compile your Express file, but you don't need nor want to generate reports, you can simply use the command **fedex**:

(host)~:**fedex -e** ***express_filename*****<return>**

Only the syntax checking and parsing are done by this command.

10.1 Eliminating nested schemas: Fed-X-flatten

In the latest version of Express, nested schemas will no longer be allowed. In addition, the QDES editor (see Section 12) appears to have problems handling nested schemas. Fed-X-flatten was written to eliminate the nesting from existing Express files. The output of Fed-X-flatten is a single Express schema which contains all of the type and entity declarations in the source file.

Fed-X-flatten should be run on your Express files containing nested schemas before generating database schemas or QDES images.

(host)~:**fedex_flatten -e** ***Express_filename*****<return>**

You will be prompted for an output file name:

output file: ***flattened_Express_filename*****<return>**

10.2 Creating a database schema from Express: Fed-X-SQL

To translate an Express schema into a database schema as defined by the SQL language, use the **fedex_sql** command:

(host)~:**fedex_sql -e** ***express_filename*****<return>**

where ***express_filename*** is the name of the file in which your Express schema is

stored. The system will then prompt you for the name of an output schema file; the SQL statements used to create the database schema will be stored in this file. Your schema file should be named using a ".sql" extension.

output file:***schema_filename*.sql<return>**

Note that several additional files will be created in addition to the schema file specified:

SUPERTYPES.sql	contains the SQL statements needed to generate views for the supertype entities defined in the Express schema. If a file named SUPERTYPES.sql already exists, it will be replaced with the newly created file. In general, you do not have to be concerned with the contents of this file.
DICT_DATA.sql DICT_INDICES.sql	contains information created for the schemas data dictionary
INDICES.sql	can be run later (after loading data) to create the unique indices on the tables.
TABLE_NAMES.txt	the default name of a text file containing the list of abbreviated table names used in the database

The next step is to load the database schema into the database. Invoke your database software and run the SQL file created in the previous step. For example, in Oracle, run **sqlplus** and log into a database. Then at the prompt, type:

SQL>**start *schema_filename*.sql<return>**

where ***schema_filename*.sql** is the name of the output schema file specified above.

The schema file takes a long time to load; it is not unusual for loading to take more than 30 minutes. As the file is loaded, many error messages may be displayed. Most of these messages are generated by attempts to insert duplicate names into the EXPRESSYS$NAMES table and may be disregarded. (A listing of the messages that appear on the screen during the load is stored in the file namederrors.lst).

SUPERTYPES.sql, DATA_DICT.sql, and DATA_INDICES.sql are also automatically "started" when ***schema_filename*.sql** is "started". INDICES.sql must be "started" after the data is loaded (it slows the loading if it is run first).

The database is now ready to be populated.

For a description of the mapping between Express and SQL see [METZ]. This document also describes the data dictionary tables that are built in this process.

10.3 Creating a QDES editor image from Express: Fed-X-QDES

Fed-X-QDES translates Express language information models into Smalltalk-80 code which can be loaded in a QDES image.

(host)~:**fedex_qdes -e *express_filename*<return>**

If no errors are found, you will be asked to provide an output filename where the Smalltalk-80 output will be written:

output file: ***output_filename*<return>**

11 The STEP physical file parser: STEPparse

STEPparse is the generic name for a family of STEP physical file translators built on the NIST STEP Working Form [CLARKa]. There are actually three programs that use this basic translator and are invoked with the commands **stepparse_step**, **stepparse_qdes**, and **stepwf_sql**.

Each of the programs expects an Express schema file and a STEP physical file as input. The Express file should be the same as the one used by the application which will use the output of the translator. In addition, the STEP file to be translated should correspond to the same information model given in the Express schema file, that is, all attributes and references should follow the structure of the schema.

Parsing is done in two parts: first the Express schema is compiled, then, if there are no errors in the first part, the STEP file is parsed.

As the STEP file is parsed, a STEP working form is built. Errors reported MUST be corrected.

The general form of all the commands is

(host)~:***stepparse_translator*** **-e** ***express_file*** **-s** ***step_file***

where ***stepparse_translator*** is the actual STEPparse family program command and ***express_file*** and ***step_file*** are the related Express and STEP files.

Sometimes the translator does not abort after error messages are displayed. If this occurs, type **<ctrl>-c**.

If no error messages are reported, the user will be asked to enter an output filename

Output file: ***output_filename*****<return>**

"Man pages" are available on STEPparse (man stepparse) in general and on each of the translators in particular.

11.1 The STEP physical file syntax checker: stepparse_step

The STEP physical file syntax checker parses a STEP file and produces an equivalent STEP file.

The syntax is the same as the general syntax described above where *stepparse_translator* is **stepparse_step**.

The output file produced will contain the new version of the STEP physical file and is typically named with a .step extension. The entity id's in this file will be the same as the input STEP file, but the order and format may be different.

11.2 The STEP to QDES translator: stepparse_qdes

Before you can edit a STEP file using QDES (see Section 12), it must be converted to a form that can be understood by Smalltalk. To convert the STEP file, use the **stepparse_qdes** command, specifying the Express file, and the STEP file to be used.

The syntax is the same as the general syntax described above where *stepparse_translator* is **stepparse_qdes**.

The output file produced is a QDES intermediate file and can be loaded into a QDES image based on the same Express schema file as used to generate it. It is typically named with a .st (for Smalltalk) extension.

See Appendix F for current directories and files where Express data, CDIM schemas and information about the STEP parser can be found. Your CDIM team leader or the PDES Testbed Hotline should also be able to provide you with current information.

11.3 Loading a STEP file into a database: stepwf_sql

The **stepwf_sql** command is used to load a STEP physical file into the database.

The syntax is the same as the general syntax described above where *stepparse_translator* is **stepwf_sql**.

The output file produced will contain messages to be used for informational and debugging purposes. Name this file whatever you want; in general, you do not have to be concerned with its contents.

You will also be prompted for your database username and password:

Enter Oracle user id: ***O_username*<return>**
Password: ***O_password*<return>**

If you don't have a database account, you should call the PDES Hotline to request one (see Appendix B).

At this point, **stepwf_sql** displays

Loading database.

Errors reported after this are database errors. If an error occurs, the database is rolled back to its original state, before **stepwf_sql** execution.

12 Using the QDES model editor

QDES, the Quick-and-Dirty Editor for STEP, is used to add data such as tolerances, dimensions, and product configuration to a STEP model. Although you could use QDES to edit the geometry and topology of the part model, this is not recommended since there is no way to verify the validity of a geometric model created or modified using QDES.

QDES is implemented in the Smalltalk-80 programming environment, and thus runs only on those machines on which Smalltalk-80 is installed. See Appendix C for hosts on which Smalltalk-80, and thus QDES, is available. To learn how to use QDES, you should read the QDES User's Guide [CLARKb] and work through the tutorial that accompanies it. The instructions that follow describe the procedure for starting QDES, the different operations that you can perform with QDES, and a few hints for using the system.

12.1 Getting started with QDES

12.1.1 QDES on a Sun with the generic environment

First, login to the Sun using your account name and password and start X Windows as described in Section 2.1.

With the cursor positioned in the background window, press and hold the left mouse button; pull down until the QDES choice is highlighted; pull right then down until the QDES model you wish to edit is highlighted; release the mouse button. The editing session will create its own window in which to work.

The model choices look like

qdes-*imagename@hostname*

where the *imagename* is the Smalltalk image of the Express file that you want to use and *hostname* is the DECstation on which QDES will remotely run. Several messages will be displayed while QDES is being loaded. Two windows and three window icons will then be displayed: the QDES Feedback window, the STEP Browser window, and the PDES Class Hierarchy, PDES Topology, and PDES Geometry window icons.

Before running QDES in earnest, you are strongly advised to first run the QDES tutorial, using a simple part. Instructions for running the tutorial are given in the

QDES User's Guide [CLARKb]. The part used throughout the tutorial is a cube with a hole through it; the part file is tutorial.st (see Appendix F for the directory where the file is located).

Be careful when expanding and collapsing the PDES Class Hierarchy, Topology, and Geometry windows! If you select "close" instead of "collapse" you will close the window and eliminate the icon. The pull-down menus in QDES are sticky; that is, if you pull down a menu from one window and select "close", then pull down a menu from another window, "close" will be highlighted in the new menu. If you are quickly closing windows to clean up your QDES display, it is very easy to make a mistake and lose system icons that you really want to keep.

Keep in mind that in QDES, the right mouse button is used to pull down window menus from the bars across the tops of the windows. These menus allow you to perform window operations, such as re-sizing, moving, and hiding. The middle mouse button is used to pull down and select operations from other QDES menus, including the system menu. System operations include clearing the workspace and loading a file. Other menu operations performed with the middle mouse button include selecting a list of elements and creating an entity graph. The left mouse button is used to "click" on a selection within a window or to expand an icon.

12.1.2 QDES on a DECstation 3100

QDES may also be executed from the DECstation 3100 systems (see Appendix E). QDES performance is slightly better on the DECstation than on the Sun.

12.2 Loading a new Express File into QDES

Instructions for loading a new Express file into the QDES model editor can be found in the postscript file qdes-admin.ps [CLARKe] and will not be duplicated here (see Appendix F for the path to the appropriate directory). However, the following notes are included for clarity.

Throughout the instructions in the README file located in the directory discussed above, there are references to the file Qdes<N> where the <N> is the current version of QDES. When following these instructions, type **Qdes-2.2.**

In step 2 of the instructions, the filename <foo>.exp indicates the name of the CDIM Express file you are trying to load. The output file, referred to as <foo>.exp in the instructions, should be given the same name as the Express file, with a .st extension replacing the .exp extension of the Express filename.

If you would like to add an Express file to the QDES menu described in Section 12.1.1, please call the PDES Testbed Hotline for assistance (see Appendix B).

13 The geometry visualizer

13.1 Introduction

The STEP geometry visualizer is a software application that reads in a STEP physical file that includes geometry, and displays this geometry in color in a three dimensional environment. The application allows the user to pan, zoom, and rotate the displayed geometry. Multiple views of the geometry can be displayed. Surfaces can be shaded and output to hardcopy devices such as plotters and printers is supported.

The user also has the ability to select geometry from the screen to determine what the corresponding physical file ID is, or to enter a physical file ID which highlights the entity. In addition, the option of toggling surfaces and labels off/on is provided. The physical file ID is labeled next to each piece of geometry. The size of the label is different for surface, curves and points, with surfaces having the largest text size and points the smallest.

A physical file with about 10 curves, 10 surfaces and 20 points takes about 15 seconds to display on a SUN Sparcstation. This includes reading in the Express schema, parsing the physical file, creating a graphics window, and displaying the actual entities.

13.2 Implementation environment

The visualizer was developed on an Apollo workstation utilizing the GM/EDS C4 Toolkit [EDS]. This toolkit provides portability to Sun 3 or Sun 4 workstations, and DECstation 3100s running ULTRIX. At this time, no other machines are supported. Porting this application to other platforms would only be possible if GM/EDS agreed to port their toolkit to that platform.

The visualizer also uses NIST developed software to read in the physical file. (see [CLARKa]).

13.3 Visualizer execution

In order to execute the visualizer, you will need a STEP physical file and the Express schema that defines the physical file data. To execute the program at the UNIX prompt, type:

(host)~:**visualizer -e *express_file* -s *STEP_file* window**

The "window" is optional, and will create a separate graphics window on the display. Error messages are written to standard out and a file.

Once the Express and physical files have been parsed, you will be prompted to enter a file name for these messages. This file will also contain a list of the geometry entities displayed and may contain more complete error messages.

The most important restriction is that the visualizer only operates on entities that are defined in the "pre-St.-Louis geometry" schema. The user specifies the Express schema and physical file to use. The visualizer will sift through the entities that are in the physical file and display those entities that it recognizes.

Entities that are not recognized have no effect on the visualizer. The attribute names of the recognized entities must match the "pre-St.-Louis geometry" schema attributes.

13.4 Main menu options

There are five options on the main menu.

1 Verify a PDES Entity - This option will allow you to pick a STEP entity from the screen. The picked entity is highlighted and a message is displayed containing the entity type and physical file ID. Upon selecting this option, a submenu will appear with two more options: Turn Approval On and Clear Reject.

   The Turn Approval On option gives you an opportunity to evaluate a screen pick and then approve it by pressing PROCEED. Pressing REJECT will disable the currently selected entity and allow you to re-select a different entity.

   The Clear Reject option allows you to re-enable all previously rejected entities.

2 Toggle the Labels ON/OFF - The geometry can be displayed with or without the physical file ID used as labels.

3 Toggle the Surfaces ON/OFF - This may be needed to see the underlying curves.

4 Find Entity - If the location of a specific entity is desired, the user can

have that entity highlighted.

5 EXIT - quit the program.

The mouse buttons are left - SELECT, center - PROCEED, and right - REJECT.

See Section 13.6 on use of the mouse for more information.

13.5 Graphics Control Utilities (GCU)

The Graphics Control Utilities (GCU) are a collection of functions that enable you to define and manipulate the graphics environment. There are eleven GCU functions.

FUNCTION	DESCRIPTION
Autoscale	Re-sizes the entire geometry definition so that all the data appears as large as possible on the screen.
Diagonal Zoom	Fills the screen display with an area specified by diagonal points
Hardcopy	Generates a hardcopy of the current screen image, including the dialogue area.
Layout	(1) Manages screen layouts, with six predefined layouts of one to six viewports, each of which has specific orthographic and axonometric orientations. (2) Toggles the viewport outlines and axis display.
Plot	Creates plot files and sends them to the proper plotter queue.
Repaint	Re-displays the current graphics image, including those entities partially erased during some operations.

Reset Display	Resets all the view rotations: (1) If you are using one of the predefined layouts, rotations are reset to the default orientations for these layouts. (2) If you have created additional viewports, rotations are reset to the current global reference systems X, Y, and Z rotation. Scale and translation are unaffected by the reset.
Set Up Graphics	(1) Specifies the menu size, the selector-window, environment size, user interface colors, curve smoothness, and hardcopy specifications. (2) Toggles the display of coordinate feedback on and off.
Surface Display	Controls shading and hidden-line removal
View	(1) Performs analog and discrete rotations, scaling, and translations to move your viewing position up, down, left, right, in, out, and around an object without moving the object itself. (2) Aligns the object with the current reference system. (3) Sets the Z-clipping planes. (4) Provides view alignment functions.
Viewport	Adds, deletes, or moves viewports in a layout.

GCU functions do not disrupt the function of the visualizer. To use a GCU function, you press one of the eleven GCU function keys. The function keys locations are dependent on the platform you are using but have generally the same layout. For the SUN workstations the key mappings are:

KEY	FUNCTION
F1	AUTOSCALE
F2	RESET DISP
F3	DIAG ZOOM
F4	VIEW
F5	VIEWPORT
F6	LAYOUT
F7	SET UP GRAPHICS
F8	SURFACE DISPLAY
F9	REPAINT
SHIFT F8	PLOT
SHIFT F9	HARDCOPY

13.6 Mouse functionality

This application requires that you use a three button mouse. Each button has a specific function inside the application:

left button	SELECT	Use this button to select items from a menu or to select geometry from the screen.
middle button	PROCEED	This button should be pressed to select a highlighted menu option or to continue when a message is displayed that specifies "PRESS PROCEED TO CONTINUE".
right button	REJECT	This button is used to reject a selection.

13.7 Restrictions

The most important restriction is that the visualizer only operates on entities that are defined in the "pre-St.-Louis" schema, the older version of the geometry schema. The user specifies the Express schema and physical file to use. The visualizer will sift through the entities that are in the physical file and display those entities that it recognizes. Entities that are not recognized have no effect on the visualizer. The attribute names of the recognized entities must match the "pre-St.-Louis" schema attributes.

The visualizer recognizes and processes the following entities:

POINTS
- cartesian_point
- point_on_curve

CURVES
- line
- circle
- ellipse
- hyperbola (only as a trimmed curve)
- parabola (only as a trimmed curve)
- poly_line
- b_spline_curve
- trimmed_curve (cannot trim offset curves and surface curves)
- d2_offset_curve
- d3_offset_curve

SURFACES
- cylindrical_surface
- conical_surface
- spherical_surface
- toroidal_surface
- surface_of_revolution
- surface_of_linear_extrusion
- b_spline_surface

Entities that are not supported are:

point_on_surface
plane
rectangular_trimmed_surface
curve_bounded_surface
offset_surface

The following entities are context type entities. Their exclusion should not affect the graphics display.

composite_surface
pcurve
surface_curve
intersection_curve
composite_curve_on_surface
rectangular_composite_surface

The Fed-X compiler and Stepparser are used to retrieve the data. Due to the fact that these use memory-resident data structures, there are limits to the physical file size that can be displayed.

Currently, physical files with greater than about 4000 entities take hours to process. Less than 4000 entities runs in a matter of minutes.

14 The validation databases

14.1 The database management system (DBMS)

The database management system used in the VTL is the Oracle relational DBMS. This DBMS uses a superset of the standard structured query language (SQL) to create and manipulate databases. The software installation in the PDES Testbed includes C and FORTRAN language interfaces as well as a user defined forms interface. Manuals for Oracle are available for use while in the VTL at NIST, or manuals can be purchased from the Oracle Corporation.

14.2 Logging in and out

To use an Oracle database, login or remotely login to the host system assigned to your project.

(host)~:**rlogin** ***project_hostname***<**return**>

Select the database instance to be used if it is different than the default instance for your project. See Appendix C for the current list of database instances available and their computer assignments.

(host)~:**setenv ORACLE_SID** ***database_instance***<**return**>

Obtain a personal account from the Database Administrator (see Appendix B) so that you can practice using Oracle without interfering with anyone else's work. After you have taken the Oracle training course, you will be able to use other databases to perform actual STEP testing.

Start up SQL using your Oracle account name and password:

(host)~:**sqlplus** ***O_username/O_password***<**return**>

You will now get the SQL prompt:

SQL>

Try the following simple SQL command to see a list of all tables available in the current database:

SQL>**select * from accessible_tables;<return>**

A list of owners, table names, and types will be displayed:

OWNER	TABLE NAME	TABLE TYPE
SYSTEM	DUAL	TABLE
REPORT	FR_FIELD	TABLE
REPORT	FR_TEXT	TABLE

To change your password in SQL, type:

SQL>**grant connect to *O_username* identified by *newpassword*;<return>**

where ***O_username*** is your Oracle username and ***newpassword*** is the password you want to use.

To exit SQL, type exit:

SQL>**exit<return>**

14.3 Creating a STEP database

For instructions on creating, loading, examining, and reformatting a STEP testing database, see Sections 10, 10.2, 11.3, 15 and 16.3.

14.3.1 Account inter-relationships

The testing accounts for the validation databases are arranged in a "master/tester" relationship. In this relationship, the "master" is an account which serves as the owner of the objects (tables, views, indexes, sequences, etc.) that make up the schema being tested. The master must "grant privileges" to the testers before they are allowed to use the schema owned by the master. The "tester" accounts are the accounts from which all data loading, testing, and deleting should be done. "Testers" are not allowed to add tables to, alter tables in, nor drop tables from the

"master" accounts. In this way, the schema is at least minimally protected from inadvertent and unwanted changes.

14.3.2 Loading the schema into one of the master accounts

If you are not starting with a new account, that is, if a schema already exists in the master account you are going to use, you should clear it out by "dropping" all the objects in the account. The account you are using (a master account) will actually own the objects. Later on you will grant privileges from this account to other accounts (tester accounts) so that they can use the schema.

Start a database session as described in Section 14.2. You will see the database prompt:

```
SQL>
```

Enter the following command to load the schema created from your Express file using **fedex_sql**:

```
SQL>start schemafile.sql<return>
```

where ***schemafile*.sql** is the name of the output file created during the **fedex_sql** execution. All the objects (tables, views, indexes, sequences, etc.) needed for the schema will be created, but no data will be loaded yet. (Note that an error log is automatically created when ***schemafile*.sql** is started).

Exit the database session.

14.3.3 Setting up the tester accounts

In order for tester accounts to have access to master objects, specific privileges must be granted for each object to every tester that is going to use it. The specific privileges are: delete (data), index, insert, references, select, and update. The testers are specifically NOT allowed to add, alter (change the columns of) or delete tables. A SQL language reference will give you instructions on how to grant the privileges.

Even though the tester has privileges to use the master's objects, the tester is still not quite ready to begin loading data into the schema.

When the tester is first given privileges to use the master's objects, the tester must

reference those objects by declaring the owner's name (the database username) and the object's name

owner.object_name

To make referencing easier, synonyms can be created for ANY object to which an account has access. To make the best use of the master/tester relationship described here and the Testbed tools, it is necessary for the tester to create a synonym for each of the objects to which privileges have been granted by the master (owner). The synonym used is simply the name of the object with the prepended owner name stripped off. The syntax for creating synonyms can be found in a SQL language reference.

Once the synonyms have been created, the data can be loaded using **stepwf_sql**, described in Section 11.3.

To make clearing the accounts, granting privileges, and creating synonyms easier and more consistent, several utilities have been created.

14.4 Database utilities

Once a schema for testing has been created, there are several operations which the user typically wants to perform. Since these operations often need to deal with virtually every object in a schema (potentially there could be hundreds of tables, views, synonyms, etc.), it would be far too tedious and time-consuming for users to enter the appropriate, individual SQL statements required to obtain the results needed.

Therefore, several utility scripts have been developed to perform those "global" operations.

Note that while all of the utilities were designed for the PDES Testbed, some of the utilities are general purpose and can be used with any database. Those marked with an asterisk (*) contain references to PDES Testbed specific objects and may generate some error messages or may not work as expected or at all on other database schemas.

See Appendix F for the directory where these utilities can be found. Your generic environment should already include a path to these utilities.

The utilities are:

userdescribetables userdescribedicttables*	describe the fields of all tables (column names, data types, etc)
usergrantprivs* userrevokeprivs	allow/disallow another user to have access to objects that you own
usercreatesyn	create synonyms for all objects owned by some other user on which you have been granted privileges
userdropobjects userdroptables userdropviews userdropindexes userdropsyn userdropdata* userdropdictdata*	remove or "drop" the identified group of entities from the database
userexport userimport	perform basic exports and imports of a user's schema and data

14.5 Utility command syntax and use

In addition to the command syntax described in Section 1.1, the following conventions are followed:

- ***O_username*** and ***O_password*** are the user's Oracle username and password.
- ***ownerid*** is the Oracle username of the owner of the database object.

Each of the commands described here should be given at the **system** prompt; they will automatically begin and end the Oracle session for you.

The commands and their functions are described below.

(host)~:**userdescribetables** ***O_username/O_password*<return>**

userdescribetables produces a file that contains a list of all the tables and views owned by the user along with the names of the columns, the type of data that each column contains, and whether null values may be stored in the column. The file produced can be viewed or printed using any of the system commands available for that purpose.

(host)~:**userdescribedicttables** ***O_username/O_password*<return>**

userdescribedicttables produces the same output for the data dictionary tables (those that begin "EXPRESSYS$") as **userdescribetables** does for the regular tables. In particular, they both use the same output file to store the descriptions. Therefore, if you want to use both commands, the output file from the first command given should be printed or the name changed before the second command is used.

(host)~:**usergrantprivs** ***O_username/O_password grantee*<return>**

where ***grantee*** is another Oracle username.

usergrantprivs is used to give permission to ***grantee*** to use objects created and owned by ***O_username***. The grantee will then be able to save, update, and delete data stored in those objects but will not be able to modify or drop those objects.

(host)~:**userrevokeprivs** ***O_username/O_password grantee*<return>**

Exactly the opposite of **usergrantprivs**, **userrevokeprivs** is used to deny permission to ***grantee*** to use objects created and owned by ***O_username***.

(host)~:**usercreatesyn** ***O_username/O_password ownerid***<return>

where ***ownerid*** is another Oracle username

By using synonyms, a user can create any number of aliases or "nicknames" for tables or other objects. Referencing an object via its synonym is exactly the same as using the real object name. A user may create synonyms for objects he "owns" as well as for objects to which he has been granted privileges.

For example: normally, to reference a column in a table owned by another user (a table to which you have been granted privileges) you must prefix the table name with the owner's Oracle username (e.g., ownerid.tablename.columnname). However, by creating a synonym for ownerid.tablename, the reference becomes synonym.columnname.

Another application is to create queries or other SQL statements using synonyms rather than absolute tablenames. For example, if a user has two tables that have the same columns and data types but which, for whatever reason, he does not want to store as a single table, he can write one set of queries instead of two by using a synonym and then "pointing" that synonym to whichever table he actually wants to use.

usercreatesyn creates synonyms for all the objects owned by ***ownerid*** to which ***O_username*** has been granted permission to use. The synonyms created are simply the tablenames without the ***ownerid*** prefix.

(host)~:**userdropobjects** ***O_username/O_password***<return>
(host)~:**userdroptables** ***O_username/O_password***<return>
(host)~:**userdropviews** ***O_username/O_password***<return>
(host)~:**userdropindexes** ***O_username/O_password***<return>
(host)~:**userdropsyn** ***O_username/O_password ownerid***<return>
(host)~:**userdropdata** ***O_username/O_password***<return>
(host)~:**userdropdictdata** ***O_username/O_password***<return>

The "**userdrop...**" commands are all similar in that they all erase from the database the entire group of things identified by the command name:

userdrop...

objects	drop all of ***O_username***'s objects (tables, views, synonyms, etc)
tables	drop all of ***O_username***'s tables only
views	drop all of ***O_username***'s views only
indexes	drop all of ***O_username***'s indexes only
syn	drop all of ***O_username***'s synonyms only from the owner's object table
data	delete all data from ***O_username***'s tables and drop all sequences
dictdata	delete all data from ***O_username***'s own data dictionary tables (those that start "EXPRESSYS$")

(host)~:**userexport** ***O_username/O_password exportfilename*<return>**

userexport performs an export of all of ***O_username***'s objects and any data stored there along with any grants, indexes, or constraints. Exported data is stored in a file named ***exportfilename***. If ***exportfilename*** does not include a (.), the default extension name ".dmp" will be appended to the filename.

(host)~:**userimport** ***O_username/O_password importfilename*<return>**

userimport imports all the objects and data exported by **userexport** and as stored in ***importfilename***. It does not allow the creation of an object that already exists.

In the future, other commands of interest may be placed in the directory. If you have a system management type task, and aren't sure how to do it, browsing this directory may spark some ideas.

14.6 Database backups

PDES databases will be backed up every morning at NIST. To retrieve previous versions of your data, contact the Database Administrator or the PDES Testbed Hotline (see Appendix B).

15 Creating a STEP physical file from a relational database

The program PDESOUT (po) retrieves the data from a relational database (using SQL) and formats it to the STEP physical file format.

PDESOUT requires two, Oracle specific, command line arguments, the database username and its password respectively (see Section 14). All queries and messages are written to standard error, stderr, and by default appear on the screen. Since all output is normally also directed to stdout (the screen) as well, the user should redirect this data to a file for later use, otherwise it will be lost. (See Section 3.6 for a discussion of standard I/O and redirection).

User input is required to complete the header section of the STEP physical file. This input is queried for and can be supplied from stdin, the keyboard. Optionally, the user may prepare a file containing the required information and direct it to PDESOUT. This information consists of the following five items in the order shown.

1. The user's name and mailing address on as many lines as desired,
2. A blank line,
3. The name of the user's company and organization,
4. A blank line,
5. A one-line description of file contents, and
6. A one-line description of the security classification.

If there is no security classification, the word "NONE" should be supplied.

An example of the command line is:

(host)~:**po** ***db_username db_password <userdesc.dat >pdes.ef***

This command will run the po process using ***db_username/db_password***'s database. The optional input data will be supplied from the file called *userdesc.dat* and the output will be placed in the file called *pdes.ef* which will be in STEP physical file format.

The output file will contain no record separators or unnecessary white space of any kind. Considerations for "pretty printing" to format the output file into something more human readable are addressed by the various filter programs (see Section 16.3).

No provisions are made for blocking output data into records for transfer to tape or other computer media. This must be accomplished by some other system utility.

Definitions of entities to be processed are contained in a database table named OUTFILE_LIST. The user may create and populate such a table to control the output of PDESOUT. If no such table is present or if the table is present, but empty, all the entities in the database will be processed.

NOTE

Entities which are marked NOT SHARABLE in the database should not be included in OUTFILE_LIST. These entities are only output as they are referenced by the entity in whose scope they are defined. If such entities are encountered in OUTFILE_LIST, they will be output incorrectly as SHARABLE entities.

The SQL command for creating OUTFILE_LIST is:

```
CREATE TABLE OUTFILE_LIST
        (ENT_ID CHAR (40) UNIQUE NOT NULL,
        EF_ID INTEGER);
```

In populating this table, the user should enter entity id's into the ENT_ID column. As a matter of good practice, the EF_ID column should be set to NULL. However, should this not be done for some reason, it will not cause an error in the output of PDESOUT since the field will be reset to contain the exchange file sequence numbers which correspond to the entity id's during the execution of PDESOUT. This allows the user to cross reference data in the database to the data in the physical file. After completion, OUTFILE_LIST will contain a mapping between database entity names and the physical file id's which run sequentially from 1 across all the entities of any type. The data in the database, including the contents of OUTFILE_LIST, is assumed to be correct. Likewise, it is assumed that any data consistency checking was done at the time the data in the database was created.

16 STEP file utilities

16.1 Converting from old to new format

The STEP file utility program Old-to-New (**oton**) converts PDES, Inc. Geometry 1B format physical file into a Part 42 format physical file. In particular, the CV-to-STEP translator (**pm2step**) creates Geometry 1B STEP physical files that need to be converted.

The syntax is

(host)~:**oton** ***input_file output_file*<return>**

The ***output_file*** must not be the same as the ***input_file*** and if ***output_file*** already exists, it will be overwritten.

16.2 Searching a STEP file for entities

It is often desirable to get information about specific entities in the STEP physical file. To that end, get-STEP was created. Using get-STEP, the user can query the STEP file for an entity's attributes or for any other entities that use the specified entity.

The syntax is

(host)~:**getstep**

A prompt for the STEP physical filename will then be displayed

Enter STEP Physical file name to search:
***STEP_filename*<return>**

After the filename has been entered, the following menu will be displayed:

Enter a Selection (L,S,E,X)
L (get users - long form)
S (get users - short form)
E (get entity)
X to exit
***option*<return>**

The letter codes are not case-sensitive. If L, S, or E are selected, you will be prompted for a physical file ID:

Enter a Physical File ID
***id*<return>**

If L or S is used and the ID is found, you will get a listing of the entities as they are found

Entity ***id*** is used by the following entities:

(list of entities)

If E is used and the ID is found, you will get a listing of the elements as they are found

The Elements of Entity ***id*** are

(list of elements)

"Man pages" (see Section 3.8) for **getstep** are available. [SILVERNALE]

16.3 Reformatting STEP physical files

According to the STEP specification, non-printable characters, such as carriage returns, are not allowed in STEP physical files. As a result, printouts of these files can be hard to read.

16.3.1 The print exchange file (pef) filter

The print exchange file, pef, filter can be used to print out STEP physical files in a readable format. This filter has been created for the purpose of adding white space to an exchange file in such a way as to make it generally legible. This program also breaks the STEP file into lines of 80 characters or less. This can be used to overcome limitations in some PDES parsers. If the parser cannot read more than 80 characters/line, run the file through **pef** and use the resulting file as input.

As with other UNIX filters, **pef** reads one file on the UNIX input stream and writes the modified file to the UNIX output stream. It may be used with file direction or as a function in a piped command.

Examples:

(host)~:**cat pdes.ef | pef >pdes.ef2**

or

(host)~:**pef <pdes.ef >pdes.ef2**

Either of these commands will take an unformatted exchange file, pdes.ef and format it to a second file called pdes.ef2.

(host)~:**cat pdes.ef | pef | more**

The above command will cause the unformatted exchange file, pdes.ef to be displayed in formatted fashion on the screen, one page at a time using the **more** filter (see Section 3.3.5).

(host)~:**po** ***db_username db_password*** **<userdesc.dat | pef >pdes.ef**

The above command will run the **po** process (Section 15) obtaining the inputs it requests from a file called userdesc.dat. The output PDES exchange file will be piped to **pef**, and the results stored as file pdes.ef.

16.3.2 The remove non-printing (rmvnonpr) characters filter

The remove non-printing (**rmvnonpr**) filter can be used to remove non-printable characters from a readable STEP file.

(host)~:**rmvnonpr <*input_file* >*output_file*<return>**

Additional information about the use of this command can be obtained from the "man pages" (see Section 3.8).

16.3.3 Adding linefeeds to STEP physical files

The filter **fixin** is designed to remove all extraneous white space from STEP physical files and then to add linefeeds immediately after semicolons not in quoted strings. This process has the effect of creating an output STEP physical file where each entity occurs as a single line of text.

The output file is often used with other tools such as QDES (see Section 12) and

stepwf_sql (see Section 11), where program diagnostic statements reference line numbers in their input files.

(host)~:**fixin** < ***input_filename*** > ***output_filename***<**return**>

16.3.4 Renumbering STEP physical file entities

Often the same entity numbers are used in two different STEP physical files that need to be combined. In order to prevent conflicts, one set of numbers must be changed. The renumber program changes all existing entity sequence and reference pointer numbers in a STEP physical file by adding a constant offset value to them.

Renumber requires three input values: the name of the existing STEP physical file, the name of the STEP physical file that will be created, and the integer offset.

The command line syntax is

(host)~:**renumber -i** ***input_file*** **-o** ***output_file*** **-f** ***offset*** <**return**>

where ***offset*** is the integer value that will be added to each entity ID. The offset may be negative, however, if sequence or reference pointer numbers are not greater than zero, the program will warn you and a (bad) physical file will still be created.

If any of the parameters are omitted, you will be prompted for them.

A "man page" for **renumber** is available (see Section 3.8). [SILVERNALE]

17 Bugs and updates

17.1 Reporting bugs

After you use the National PDES Testbed, you may want to make some comments about it; some of your comments may even help other Testbed users. If you want to register complaints, describe bugs, suggest improvements, or make observations, you can phone or send e-mail to the PDES Testbed Hotline (see Appendix B).

17.2 The PDES Testbed Hotline

NIST has created the PDES Testbed Hotline to provide better response times and tracking of your problems. It is staffed according to the following schedule:

Monday	1:00 PM to 4:30 PM
Tuesday Wednesday Thursday	9:30 AM to 4:30 PM
Friday	1:00 PM to 4:30 PM

All times are Eastern. The PDES Hotline is closed on federal holidays and occasionally on other days as scheduled and announced in advance.

You may also send e-mail to the Hotline. Sending e-mail is the preferred method when explicit error messages or code needs to be included in reporting difficulties. (See Appendix B). E-mail may be sent at any time but perhaps may not be read until regularly scheduled Hotline hours.

Any concerns or questions regarding the PDES Testbed Hotline should be directed to the PDES Testbed Hotline Administrator or to the National PDES Testbed Administrator.

17.3 Getting updates

Updates to this manual will be posted in the news category called nist.pdes in Emacs gnus. See Section 4.3 on how to get news.

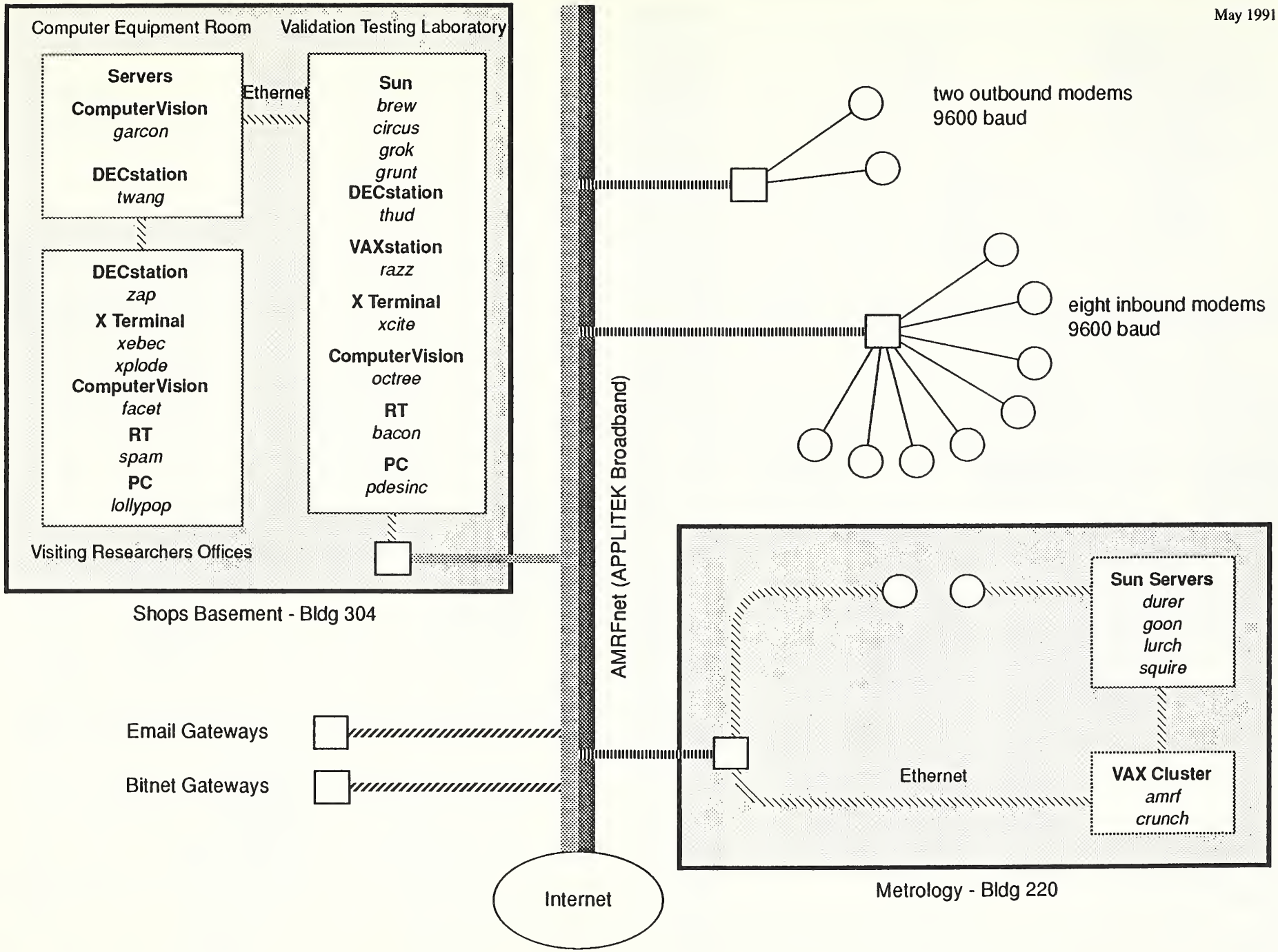

Fig-1

Figure 1. FASD/PDES Computers

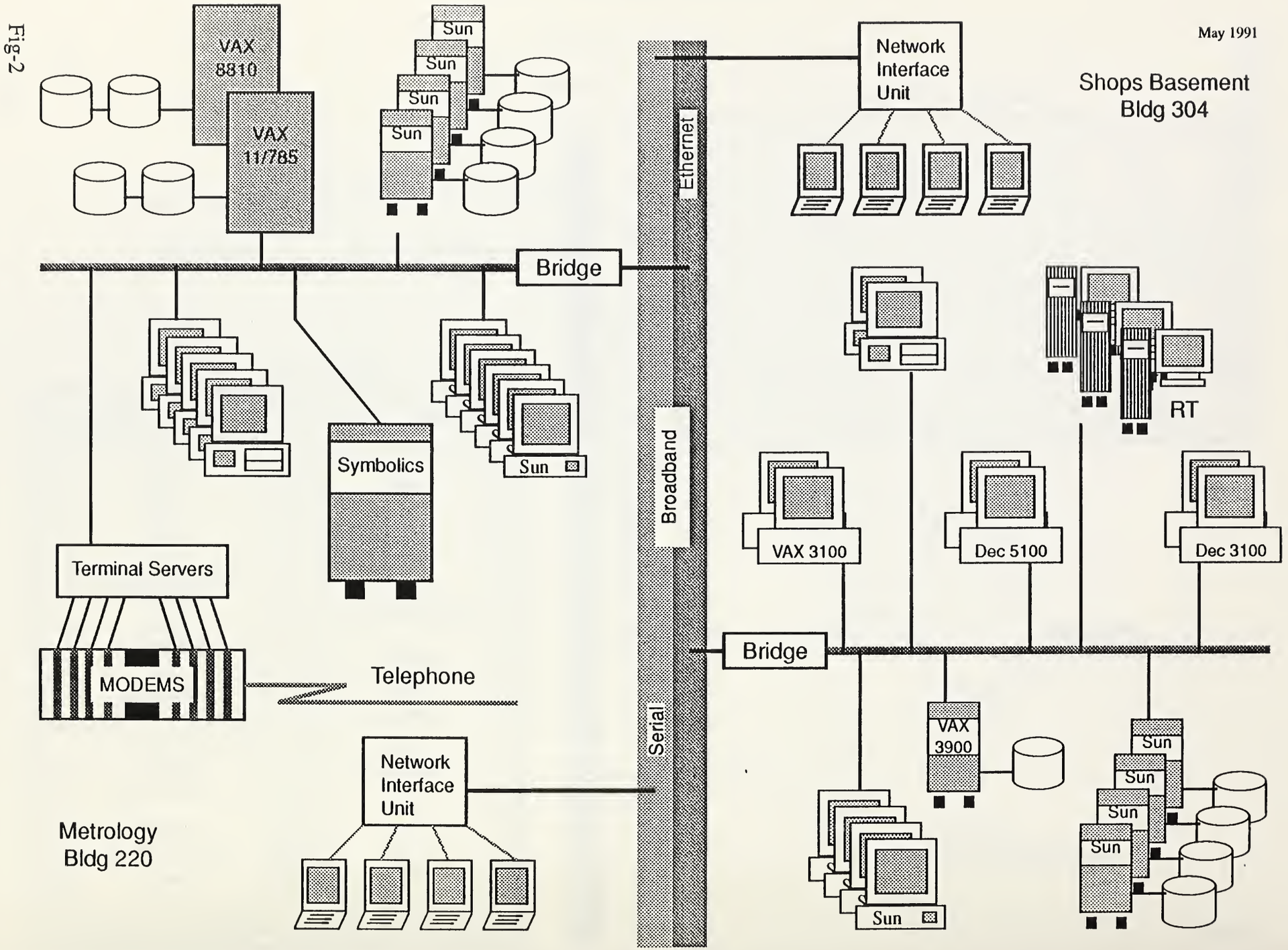

Figure 2. FASD Global Systems

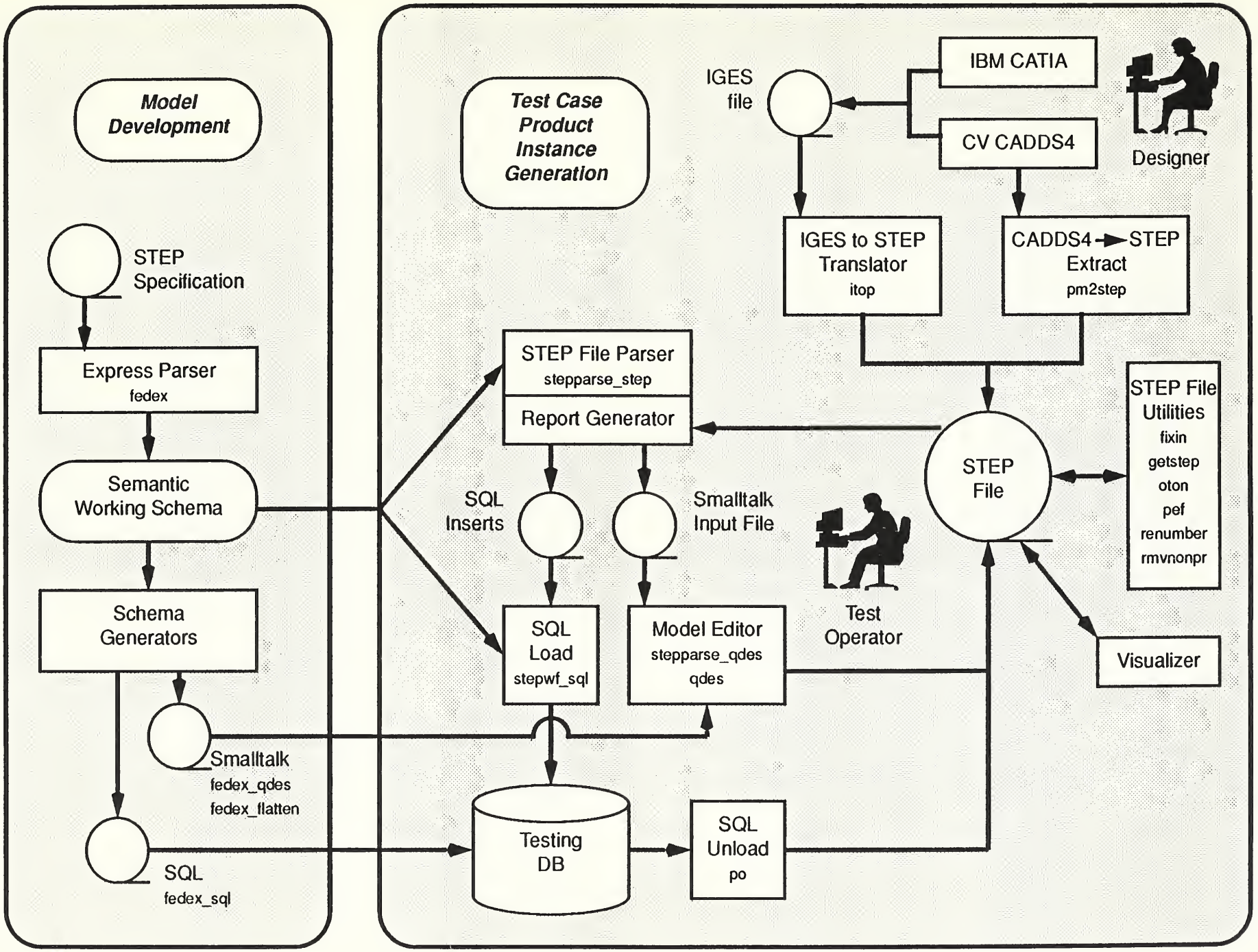

Fig-3

Figure 3. Testbed STEP Software (BPR2.1/VTS 2.0)

Appendix A Dial-in to the National PDES Testbed at NIST

How To Use the Dial-In Terminal Server

There are 8 high-speed V.32 dial-in modems attached to the National PDES Testbed Ethernet. Users can dial in at speeds of 300, 1200, 2400, 4800, 9600, or 19200 bps. These modems are attached to two separate 4-line rotaries, and may be contacted by dialing:

(301) 963-2612
or
(301) 258-8996

When the modems connect, the user should see a "CONNECT" message from their local modem. Several seconds after a connection has been established, the Ethernet terminal server (to which our modems are attached) will respond to a carriage return with the following prompt:

Welcome to the FASD Network Terminal Server

telnet>

NOTE WELL: It may take some time after the CONNECT before the terminal server will recognize a carriage return. DO NOT send a carriage return before the CONNECT message because this will cause the modem to which you are connected to disconnect. After the CONNECT message wait a few seconds, then try a carriage return. If the response shown above is not immediate, then wait a few more seconds and try the carriage return again. Repeat this process for as long as 30 seconds total before giving up. If there is no response to a carriage return after 30 seconds from the CONNECT message, please report a communications failure to the PDES Hotline (see Appendix B). Successful communication is dependent on many details; so, please be patient with us if we need specific information from you to resolve problems.

At the telnet> prompt the user can connect to a host by issuing the telnet "open" command followed by a host name, as shown below. The host names are resolved by the terminal server through network domain name service.

EXAMPLE:

telnet> **open *hostname*<return>**

The modems incorporate MNP level 5, a standard for error correction and data compression allowing data transfer rates that are higher than the baud rate. This feature is only available when both modems in the connection use MNP level 5.

The modems and the terminal server ports have been configured to do CTS/RTS flow control so that the complete ASCII character set is available to the user for computer/application control. Emacs users may now use <ctrl>-S and <ctrl>-Q freely without having to re-bind their associated functions. For the line to be transparent, your own modem must be set to pass <ctrl>-S and <ctrl>-Q also.

Questions and problems should be addressed to the PDES Testbed Hotline (see Appendix B).

Appendix B Contacts for functional titles

PDES Testbed Hotline		(301) 975-3699	hotline@cme.nist.gov
PDES Testbed Hotline Administrator	J. Newton Breese	(301) 975-6670	breese@cme.nist.gov
National PDES Testbed Administrator	Michael McLay	(301) 975-4099	mclay@cme.nist.gov
Standards Testing Center Manager	Mary Mitchell	(301) 975-3538	mitchell@cme.nist.gov
Database Administrator (DBA)	J. Newton Breese	(301) 975-6670	breese@cme.nist.gov
Standards Testing Center Laboratory -			
NIST, Bldg 304/Room 06		(301) 975-6392 (301) 975-6543	
NIST, Bldg 304/Room 12		(301) 975-5488	
PDES, Inc. Liaison	Chuck Stark	(301) 975-3416	stark@cme.nist.gov
Information Services Center Manager	Susan Katz	(301) 975-3519	katz@cme.nist.gov
Configuration Control Administrator	Gaylen Rinaudot	(301) 975-3386	rinaudot@cme.nist.gov
Configuration Management System Administrator	Scott Bodarky	(301) 975-4262	pdescmsa@cme.nist.gov
South Carolina Research Authority Testbed Manager	John Hodges	(803) 760-3283	0003887022@mcimail.com
CALS PDES Project Manager	Chuck McLean	(301) 975-3511	mclean@cme.nist.gov

Appendix C Current host assignments

Testing team	Host name	Database instance name
CDIM-A	zap	oracle
CDIM-B	thud	oracle
Sheetmetal		
Prototype Implementation		

Only one Sun (hostname "brew") is currently available for storing the personal directories and files of PDES, Inc. technical team members and to users logging in from remote locations.

Appendix D Current hardware and corresponding software versions

Sun 3 workstations ("brew", "circus", "grok", "grunt")

- SunOS 4.1, 4.3 BSD UNIX
- X-windows Version 11R4
- Oracle Relational DBMS Version 6.0
- Ontos Object-Oriented DBMS
- Smalltalk-80 Version 2.4
- PDES Testbed software/PDES, Inc. Toolkit
- FrameMaker Version 2.1 desktop publishing software

DECstation 3100 workstations ("thud", "zap")

- ULTRIX 4.0 and 4.1 and 4.3 BSD UNIX
- DEC-windows
- Oracle Relational DBMS
- Smalltalk-80 Version 2.5
- PDES Testbed software/PDES, Inc. Toolkit

IBM RT workstations ("bacon", "spam")

- AIX UNIX
- CATIA
- X-windows Version 11R3

DEC VAX Cluster ("AMRF")

- OS 5.2 VMS
- Oracle Relational DBMS Version 6.0
- DACOM Leverage data modeling software
- PDES, Inc. Parametric Evaluator
- PDES, Inc. IGES to PDES Translator

DEC VAXstation 3100 workstations ("razz", "zing")

- OS 5.2 VMS
- PDES, Inc. Parametric Evaluator
- PDES, Inc. IGES to PDES Translator

Macintosh IIx

- MacDraw II software and a plotter
- Word Perfect 5.0

Current hardware and corresponding software versions

Compaq 386 and IBM PC/AT computers

- Word Perfect 5.1
- MCI Mail/Lotus Software
- FTP
- Sun PC NFS
- IGES verification software
- DACOM Model Pro 2.0 data modeling tool
- McDonnell-Douglas' Express compiler and tools (N496)

ComputerVision CAD workstation ("facet","octree")

- CADDS4X solids modelling software

NCD X-terminal ("xcite")

Tectronix X-terminal ("xebec", "xplode")

Appendix E Locations of selected hardware at NIST

Bldg 304 (shops)/Room 06, The Validation Testing Laboratory

- four Suns, "brew", "circus", "grunt", "grok"
- IBM RT workstation, "bacon"
- DECstation 3100, "thud"
- Compaq 386, "pdesinc"
- two DEC VAXstation 3100s, "razz", "zing"
- ComputerVision (CV) cad station, "octree"
- NCD X-terminal, "xcite"
- Macintosh IIx
- Symbolics, "farm"
- Sun/Apple LaserWriter, "lw-pdes"
- IBM plotter

Bldg 304 (shops)/Room 12, PDES, Inc. Visiting Researchers Offices

- IBM RT workstation, "spam"
- DECstation 3100, "zap"
- ComputerVision (CV), "facet"
- IBM PC/AT, "lollypop"
- Zenith 286
- two Tectronix X-terminals, "xebec", "xplode"

Appendix F Current paths to and directories of selected files

~pdes	the home directory for the PDES project.
~pdes/arch/bin	executables - this subdirectory is symbolic and depends on which hardware the user is using - included in the generic .login path
~pdes/bin	other executable files (such as database user scripts) - included in the generic .login path
~pdes/docs	PDES related documents
~pdes/data	Express data, CDIM schemas, other information; see the README files in the subdirectories
~pdes/qdes	location of the QDES tutorial file, tutorial.st
~pdes/generic.env/ proto.files	location of some prototypical files that can be copied and used

Current paths to and directories of selected files

Appendix G Current formats for input and output from PDES Testbed tools

Fed-X	fedex fedex_flatten fedex_sql fedex_qdes	N496, May 1990 - Working Draft 10303-11 (input)
STEPparse	stepparse_step stepwf_sql stepparse_qdes	N279 or St. Louis (input)
IGES to PDES	itop	N279 (output) from geometry 1B
CV Extractor	pm2step	N279 (output) from geometry 1B
Visualizer	visualizer	N279 (output) from geometry 1B
PDES Out	po fixin	N279 (output)
Old to New	oton	convert from geometry 1B to Sweden/St. Louis

References

[BODARKY] Bodarky, Scott, *A User's Guide to PDES Software Development Under RCS*, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, in publication.

[CLARKa] Clark, Stephen Nowland, *The NIST Working Form for STEP*, NISTIR 4351, National Institute of Standards and Technology, Gaithersburg, MD, June 1990.

[CLARKb] Clark, Stephen Nowland, *QDES User's Guide*, NISTIR 4361, National Institute of Standards and Technology, Gaithersburg, MD, June 1990.

[CLARKc] Clark, Stephen Nowland, *An Introduction to the NIST PDES Toolkit*, NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[CLARKd] Clark, Stephen Nowland, *Fed-X: The NIST Express Translator*, NISTIR 4371, National Institute of Standards and Technology, Gaithersburg, MD, June 1990.

[CLARKe] Clark, Stephen Nowland, *QDES Administrative Guide*, NISTIR 4334, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[EDS] Electronic Data Systems Corporation, *C4-OASIS User Interface Programmer's Guide*, July 1989.

[FUHR] Fuhr, Richard, *Conversion Between Rational B-Spline and Parametric Spline Representations*, Boeing Corporation internal report #B-7137-DF-WP-631, January 1985.

References

[ISO] International Standards Organization, Working Draft 10303-42, *Part 42 Changes Resulting from Washington Meeting*, ISO TC184/SC4/WG1 Document Number N498, R. Goult, Ed., May 29, 1990.

[KATZ] Katz, Susan, *STEP On-Line Information Service User's Guide*, NISTIR 4491, National Institute of Standards and Technology, Gaithersburg, MD, January 1991.

[MCGILTON] McGilton, Henry and Rachel Morgan, *Introducing the UNIX System*, McGraw Hill, New York, 1983.

[METZ] Metz, W.P., and K.C. Morris, *Translation of an Express Schema into SQL*, PDES, Inc. internal document, November 1989.

[MITCHELL] Mitchell, Mary, *Testbed Development Plan: Validation and Testing System Development*, NISTIR 4417, National Institute of Standards and Technology, Gaithersburg, MD, January 1990.

[MORRIS] Morris, K.C., *Translating Express to SQL: A User's Guide*, NISTIR 4341, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[NICKERSON] Nickerson, D., *The NIST SQL Database Loader: STEP Working Form to SQL*, NISTIR 4337, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[OREILLY] O'Reilly, Tim and Valerie Quercia, *The X-Window System User's Guide*,O'Reilly and Associates, Inc., Sebastopol, CA, 1989.

[PTI018.01.00] Silvernale, Gerard, *PDES Inc. BLOCK POINT RELEASE 2.1 SYSTEMS MANUAL PTI018.01.00*, PDES, Inc., February 1,1990.

[RESSLER] Ressler, Sanford, *The National PDES Testbed Mail Server User's Guide*, NISTIR 4508, National Institute of Standards and Technology, Gaithersburg, MD, January 1991.

[SMITH] Smith, B. and G. Rinaudot, eds., *Product Data Exchange Specification First Working Draft*, NISTIR 88-4004, National Institute of Standards and Technology, Gaithersburg, MD, December 1988.

[STALLMAN] Stallman, Richard M., *GNU Emacs Manual, Fifth Edition, Emacs Version 1.8 for UNIX Users*, (email to) prep.ai.mit.edu, 1986.

References

Glossary

AMRF	Automated Manufacturing Research Facility
ANSI	American National Standards Institute
ANSI C	a computer programming language, ANSI C is a version of C that is a recognized standard of the American National Standards Institute
ASCII	American Standard Code for Information Interchange, a code for representing alphanumeric information
bison	a yacc work-alike program
CADDS	a proprietary mechanical CAD system with a proprietary database (ComputerVision) (Section 6)
CATIA	a proprietary mechanical CAD system with a proprietary database (IBM) (Section 8)
CMS	Configuration Management System, used to control revision and configuration of programs, documentation, etc. (Section 4.4)
DOS	Disk Operating System, typical operating system for PCs
Emacs	an extensible full-screen text editor - public domain (Section 4.1.2)
Express	a language information model, Express is used to build the schemas in which to store STEP data
FASD	Factory Automated Systems Division of the Manufacturing Engineering Laboratory of the National Institute of Standards and Technology

Fed-X Toolkit	the collection of programs developed by NIST for release into the public domain that are basic building blocks for implementing STEP-based software systems; this includes software for manipulating Express and STEP files
flex	a lex work-alike program
FSF	Free Software Foundation
FTP	File Transfer Protocol, a method of moving information from one computer to another across networks (Section 5.2.1)
Geometry 1A	STEP geometry schema working draft from the Tokyo December 1988 ISO meeting [SMITH]
Geometry 1B	STEP geometry schema working draft from the Reston 1990 IPO/ISO meeting [ISO]
GM/EDS C4	General Motors/Electronic Data Systems CAD CAM CAE CIM, a set of program tools for engineering application development [EDS]
gcc	an ANSI C compiler from the Free Software Foundation
IGES	Initial Graphics Exchange Specification, a standard for exchanging graphics data
Internet	a nationwide (physical) network system
Kermit	a set of computer file transfer and manipulation utilities - public domain (Section 5.2.2)
lex	a lexical analyzer, converts a language text into a series of tokens which are passed on to a yacc generated parser

NFS	Network File System, allows the sharing of passwords and files across all computers in the network (Section 1.3.1)
NIS	Network Information System
NIST	National Institute of Standards and Technology
NPT	National PDES Testbed
Part 42	Part 42 of STEP; the geometry model
PDES	Product Data Exchange using STEP
PDES BBS	Bulletin Board System, maintained at South Carolina Research Authority, for use by PDES, Inc. team members for project related documents (Section 1.4)
PDES, Inc.	a consortium of private industry companies dedicated to the advancement of STEP
PDES,Inc. BPR Toolkit	the collection of programs developed by PDES, Inc. and NIST for manipulating and evaluating data and schemas used in STEP
PostScript	a public domain format for printing
pre-St.-Louis geometry	STEP geometry schema, Part 42, from the Sweden 1990 ISO meeting
QDES	the Quick-and-Dirty Editor for STEP, an interim program for browsing and editing STEP data and for viewing an Express schema (Section 12)
RCS	Revision Control System, a Configuration Management System (Section 4.4)
SCRA	South Carolina Research Authority, provides general management services for PDES, Inc.

SQL	Structured Query Language, the user interface for interacting with a relational database
STEP	Standard for the Exchange of Product Model Data
STEP BBS	Bulletin Board System, maintained at NIST, containing STEP Parts, general information with respect to those Parts, and public domain PDES software (Section 1.4)
STEP Part	a subsection of the STEP standard
STEP physical file	a file of STEP data encoded by the rules specified in Part 21 of the STEP standard
telnet	program that allows host-to-host interactive communications
VTL	Validation Testing Laboratory
WordPerfect	a proprietary computer software tool for word processing produced by WordPerfect Corporation
X Windows	a computer windowing system based on a client/server model; the server supplies a client with windows in which to draw objects or text; the client and server do not have to reside on the same computer (Section 2)
yacc	Yet Another Compiler Compiler, a compiler to compile compilers

NIST-114A
(REV. 3-90)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NISTIR 4683

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
SEPTEMBER 1991

4. TITLE AND SUBTITLE

The Validation Testing Laboratory User's Guide

5. AUTHOR(S)

J. Newton Breese, Michael McLay, Gerard Silvernale

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

Office of the Secretary of Defense
CALS Program Office
PENTAGON
Washington, DC 20301-8000

10. SUPPLEMENTARY NOTES

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

Product Data Exchange Using STEP (PDES) is the United States activity supporting the Standard for the Exchange of Product Model Data (STEP), a proposed international standard for product data representation and exchange.

This document is a quick reference for those using the Validation Testing Laboratory of the National PDES Testbed located at the National Institute of Standards and Technology (NIST) in Gaithersburg, Maryland. It is not intended for those wishing to learn about the tools in the Testbed but is rather for those who already have a working knowledge of the programs and terminology.

The Testbed was initiated in 1988 under the sponsorship of the U.S. Department of Defense Computer-aided Acquisition and Logistic Support (CALS) program. A major goal of the Testbed is to provide technical leadership in a national effort to implement a complete and useful specification for the exchange of product data. This specification must be designed to meet the needs of U.S. industry and the CALS program.

The National PDES Testbed supports and actively participates in the international effort to develop the Standard for the Exchange of Product Model Data (STEP). The STEP development effort is led by the International Organization for Standardization (ISO) TC184/SC4.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

Fed-X; IGES; National PDES Testbed; QDES; PDES; STEP; SQL

13. AVAILABILITY

XX	UNLIMITED
	FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).
	ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.
X	ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
112 124

15. PRICE
A06

ELECTRONIC FORM