

625-EMD-016

EOSDIS Maintenance and Development Project

Training Material for the EMD Project Volume 16: Science Software Integration and Test

Revision 02

July 2006

Raytheon Company
Upper Marlboro, Maryland

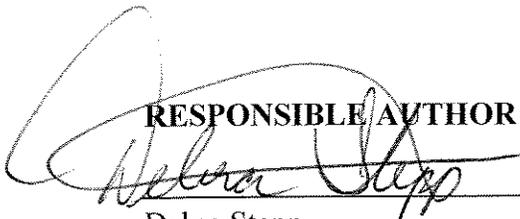
Training Material for the EMD Project Volume 16: Science Software Integration and Test

Revision 02

July 2006

Prepared Under Contract NAS5-03098
CDRL Item 23

RESPONSIBLE AUTHOR



Debra Stepp
EOSDIS Maintenance and Development Project

Date

July 28, 2006

RESPONSIBLE OFFICE



Mary Armstrong, Program Manager
EOSDIS Maintenance and Development Project

Date

7/28/06

Raytheon Company
Upper Marlboro, Maryland

625-EMD-016, Rev. 02

This page intentionally left blank.

Preface

This document is a formal contract deliverable. It requires Government review and approval within 45 business days. Changes to this document will be made by document change notice (DCN) or by complete revision.

Any questions should be addressed to:

Data Management Office
The EMD Project Office
Raytheon Company
1616 McCormick Drive
Upper Marlboro, Maryland 20774-5301

Revision History

Document Number	Status/Issue	Publication Date	CCR Number
625-EMD-016	Original	July 2004	04-0384
625-EMD-016	Revision 01	July 2005	05-0320
625-EMD-016	Revision 02	July 2006	06-0377

This page intentionally left blank.

Abstract

This is Volume 16 of a series of lessons containing the training material for the Earth Observing System Data and Information System (EOSDIS) Maintenance and Development (EMD) Project. This lesson provides a detailed description of the process required for installing and integrating science software Product Generation Executives (PGEs) into the EOSDIS environment, and once installed, test the new science software to verify its operability.

Keywords: training, instructional design, course objective, SSI&T, ESDT, DPREP, DPS, Production Rules.

This page intentionally left blank.

Contents

Preface

Abstract

Introduction

Identification	1
Scope	1
Purpose	1
Status and Schedule	1
Organization	1

Related Documentation

Parent Documents	3
Applicable Documents	3
Information Documents	3
Information Documents Referenced	3
Information Documents Not Referenced	4

Science Software Integration and Test Overview

Lesson Overview	7
Lesson Objectives	7
Importance	10

Science Software Configuration Management

ClearCase Overview	11
--------------------------	----

Creating a View in ClearCase	12
Setting a View in ClearCase	15
Creating a New Directory	16
Importing Files into ClearCase	18
Importing Multiple Files into ClearCase	20
Checking Out a File from ClearCase	23
Checking a Modified Element into ClearCase	24

Science Software Integration and Test (SSI&T) Preparation, Processes and Setup

Key Operator Roles	27
COTS Software Tools	27
Operations Tools Manual	27
General Process	28
Records	32
Performance Measurements	32
Science Software Integration & Test Flow	33
The Science Server Archive Package (SSAP)	36
ODL File(s)/PGE Metadata	39

Science Software Integration and Test (SSIT) Manager

SSIT Manager Overview	41
General Set Up of the SSIT Manager	48
SSIT Manager Tools	49
Using the SSIT Manager	49

Acquiring and Unpacking the Delivered Algorithm Package (DAP)

Acquiring the Algorithm Package via FTP	51
Unpacking a DAP	52

SSIT Software Operating Instructions	54
Subscribing to the DAP	54
DAP Insert	56
Performing a DAP Insert	56

DAP Acquire

Performing a DAP Acquire Using SSIT Manager	59
Performing a DAP Acquire Using Ingest	60

Inserting a Science Software Archive Package (SSAP)

Inserting an SSAP into PDPS	63
-----------------------------------	----

Updating a Science Software Archive Package (SSAP)

Updating an SSAP	67
------------------------	----

Standards Checking of Science Software

Standards Checking Overview	69
Checking FORTRAN 77 ESDIS Standards Compliance	69
Checking for ESDIS Standards Compliance in Fortran 90	71
Checking for ESDIS Standards Compliance in C or C++	74
Prohibited Function Checker	76
Checking for Prohibited Functions: Command-Line Version	76
Checking Process Control Files	78
Checking Process Control Files (Command-Line Version)	80
Extracting Prologs	81

Compiling and Linking Science Software

Updating the Process Control Files (PCFs)	85
---	----

Setting Up an SDP Toolkit Environment	88
Mandatory Tools	88
Optional Tools	88
Setting Up the SDP Toolkit Environment	90
Compiling Status Message Facility (SMF) Files	91

Building Science Software with the SCF Version of the SDP Toolkit

Building Science Software with the DAAC Version of the SDP Toolkit	97
--	----

Running a PGE in a Simulated SCF Environment

Setting Up the Environment for Running the PGE	101
Running and Profiling the PGE	103
Checking the PGE for Memory Leaks	104

The ECS Assistant Functionality Replaced in Part by Scripts and Monitor GUI Whazzup

Using Scripts to Start Up/Shut Down Servers	105
Using ECS Assistant	106
Bringing Up ECS Assistant	106
Using ECS Assistant to View the Science Data Server Database	110
Monitoring the System Using WHAZZUP Web GUI	111

ESDT Management

Inspecting ESDTs	115
Removing ESDTs	116
Adding ESDTs	121
Adding an ESDT Using the Science Data Server GUI	121
Updating ESDTs	123
Updating an ESDT Using the Science Data Server GUI	124

ESDT Volume Group Configuration	125
Modifying an ESDT's Volume Group Information	126
Adding an ESDT's Volume Group Information	130

Production Rules

Production Rules	133
------------------------	-----

Data Preprocessing (DPREP)

Data Preprocessing (DPREP)	135
----------------------------------	-----

Updating the Orbit Model

Updating the Orbit Model	137
--------------------------------	-----

PGE Registration and Test Data Preparation

PGE Registration	139
PGE ODL Preparation	139
ESDT ODL Preparation	142
SSIT Operational Metadata Update GUI	146
Operational Metadata	146
Test Data Preparation and Insertion of Data Granules	149
Generating a Metadata Configuration File (Source MCF)	150
Creating a Target MCF (.met) for a Dynamic/Static Granule	151
Inserting Static Data Granules into the Science Data Server	153
Inserting Dynamic Data Granules to the Science Data Server	155
Alternate Method of Inserting a Granule into the Science Data Server	157
Alternate Method of Acquiring a Granule from the Science Data Server	159
Placing the Science Software Executable (SSEP) onto Science Data Server	162
Assembling a Science Software Executable Package	162
Inserting the Science Software Executable Package onto Science Data Server	165

PGE Planning, Processing, and Product Retrieval

Production Planning and Processing	167
--	-----

Postprocessing and General Investigation

Examining PGE Log Files	169
Examining PDPS-Related Scripts and Message Files	170
Mechanisms for Capturing Information about a PGE's Execution	170
Production History (PH)	170
Failed PGE Tar File	172
Examining AutoSys JIL Scripts	172
Examining Application Log Files (ALOG)	173

File Comparison and Data Visualization

File Comparison Overview	175
Using the HDF File Comparison GUI	175
Using the hdiff HDF File Comparison Tool	175
Using the ASCII File Comparison Tool	176
Using the Binary File Difference Assistant	178
Data Visualization	180
Viewing Product Metadata with the EOSView Tool	180
Viewing Product Metadata Using the SSIT Manager	181
Viewing HDF Image Objects	182
Viewing HDF-EOS Grid Objects	184
Viewing HDF-EOS Swath Objects	185
Viewing HDF SDS Objects	186
Viewing Product Data with the IDL Tool	188
Creating an Image Display Using IDL	188
Saving an Image Display Using IDL	191
Creating a Plot Display Using IDL	192

Raster Mapping Fundamentals194

Miscellaneous

Setting Up an Environment for Direct PDPS Database Access197

Examining Application Log Files198

Practical Exercise

Science Software Integration and Test201

Slide Presentation

Slide Presentation Description203

This page intentionally left blank.

Introduction

Identification

Training Material Volume 16 is part of Contract Data Requirements List (CDRL) Item 23, which is a required deliverable under the Earth Observing System Data and Information System (EOSDIS) Maintenance and Development (EMD) Contract (NAS5-03098).

Scope

Training Material Volume 16 describes the procedures for science software integration and test. This lesson is designed to provide the operations staff with sufficient knowledge and information to satisfy all lesson objectives.

Purpose

The purpose of this Student Guide is to provide a detailed course of instruction that forms the basis for understanding science software integration and test. Lesson objectives are developed and will be used to guide the flow of instruction for this lesson. The lesson objectives will serve as the basis for verifying that all lesson topics are contained within this Student Guide and slide presentation material.

Status and Schedule

This lesson module provides detailed information about training for the current baseline of the system. Revisions are submitted as needed.

Organization

This document is organized as follows:

- | | |
|------------------------|--|
| Introduction: | The Introduction presents the document identification, scope, purpose, and organization. |
| Related Documentation: | Related Documentation identifies parent, applicable and information documents associated with this document. |
| Student Guide: | The Student Guide identifies the core elements of this lesson. All Lesson Objectives and associated topics are included. |
| Slide Presentation: | Slide Presentation is reserved for all slides used by the instructor during the presentation of this lesson. |

This page intentionally left blank.

Related Documentation

Parent Documents

The parent documents are the documents from which the EMD Training Material's scope and content are derived.

423-41-01	Goddard Space Flight Center, EOSDIS Core System (ECS) Statement of Work
423-46-03	EMD Task 101 Statement of Work For ECS SDPS Maintenance
423-46-02	Contract Data Requirements Document for EMD Task 101 ECS SDPS Maintenance

Applicable Documents

The following documents are referenced within this EMD Training Material, or are directly applicable, or contain policies or other directive matters that are binding upon the content of this document:

420-05-03	Goddard Space Flight Center, Earth Observing System (EOS) Performance Assurance Requirements for the EOSDIS Core System (ECS)
423-41-02	Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) (ECS F&PRS)
423-46-01	Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) Science Data Processing System (EMD F&PRS)

Information Documents

Information Documents Referenced

The following documents are referenced herein and amplify or clarify the information presented in this document. These documents are not binding on the content of the EMD Training Material.

313-EMD-001	Release 7.10 Internal Interface Control Document for the EMD Project
333-EMD-001	Release 7 SDP Toolkit Users Guide for the EMD Project

423-16-01	Data Production Software and Science Computing Facility (SCF) Standards and Guidelines
609-EMD-001	Release 7.10 Operations Tools Manual for the EMD Project
611-EMD-001	Release 7.10 Mission Operation Procedures for the EMD Project
625-EMD-006	Training Material for the EMD Project Volume 6: Production Planning and Processing
910-TDA-022	Custom Code Configuration Parameters for ECS

Information Documents Not Referenced

The following documents, although not referenced herein and/or not directly applicable, do amplify or clarify the information presented in this document. These documents are not binding on the content of the EMD Training Material.

305-EMD-001	Release 7.10 Segment/Design Specification for the EMD Project
311-EMD-001	Release 7.10 Data Management Subsystem (DMS) Database Design and Database Schema Specifications for the EMD Project
311-EMD-002	Release 7.10 INGEST (INS) Database Design and Schema Specifications for the EMD Project
311-EMD-003	Release 7.10 Planning and Data Processing Subsystem Database Design and Schema Specifications for the EMD Project
311-EMD-004	Release 7.10 Science Data Server Database Design and Schema Specifications for the EMD Project
311-EMD-005	Release 7.10 Storage Management and Data Distribution Subsystems Database Design and Database Schema Specifications for the EMD Project
311-EMD-006	Release 7.10 Subscription Server Database Design and Schema Specifications for the EMD Project
311-EMD-007	Release 7.10 Systems Management Subsystem Database Design and Schema Specifications for the EMD Project
311-EMD-008	Release 7.10 Registry Database Design and Schema Specifications for the EMD Project
311-EMD-009	Release 7.10 Product Distribution Subsystem (PDS) Database Design and Database Schema Specifications for the EMD Project
311-EMD-010	Release 7.10 NameServer Database Design and Schema Specifications for the EMD Project
311-EMD-011	Release 7.10 Order Manager Database Design and Database Schema Specifications for the EMD Project

311-EMD-012	Release 7.10 Spatial Subscription Server (SSS) Database Design and Schema Specifications for the EMD Project
311-EMD-013	Release 7.10 Data Pool Database Design and Schema Specifications for the EMD Project
500-EMD-001	Terra Spacecraft Ephemeris and Attitude Data Preprocessing
500-EMD-002	Aqua Spacecraft Ephemeris and Attitude Data Preprocessing
500-EMD-003	Aura Spacecraft Ephemeris and Attitude Data Preprocessing
508-EMD-001	ACRONYMS for the EOSDIS Maintenance and Development (EMD) Project
152-TP-003	Glossary of Terms for the EOSDIS Core System (ECS) Project
505-41-33	Interface Control Document Between EOSDIS Core System (ECS) and Science Computing Facilities (SCF)

This page intentionally left blank.

Science Software Integration and Test Overview

Lesson Overview

This lesson will provide you with the complete process by which a new science algorithm from the science community is received, integrated with the production system, tested and distributed.

Lesson Objectives

Overall Objective - The overall objective of the Science Software Integration and Test (SSI&T) lesson is for maintenance and operations personnel to develop proficiency in the procedures that apply to science software integration and test activities for the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS).

Condition - The student will be given oral or written information and requirements for performing science software integration and test activities, access to the planning and production processing systems, a copy of 609-EMD-001, *Release 7.10 Operations Tools Manual for the EMD Project*, a copy of 611-EMD-001, *Release 7.10 Mission Operation Procedures for the EMD Project*, and a copy of 625-EMD-006, *Training Material for the EMD Project, Volume 6: Production Planning and Processing*.

Standard - The student will perform science software integration and test activities in accordance with the prescribed procedures without error.

Specific Objective 1 - The student will perform routine running of scripts to start up/shut down servers.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 2 - The student will put text files from the DAP under configuration management using ClearCase to create a view, create a new directory, import files, check files in, and check files out.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 3 - The student will review the SSIT preparation and setup procedures.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 4 - The student will perform routine operation of the SSIT Manager Graphical User Interface (GUI)

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 5 - The student will acquire the delivered algorithm package (DAP) using file transfer protocol (ftp) and unpack if the DAP is delivered as a tar file.

Condition - The student will be given an operational system with supporting equipment. The student will be directed to review the README Text File to ascertain information that is directed to a particular PGE.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 6 - The student will verify that FORTRAN 77, Fortran 90, C and Ada source files are compliant with the ESDIS Data Production Software and Science Computing Facility (SCF) standards and guidelines document and will search source files for prohibited functions. The student will also run the Process Control File (PCF) syntax checker and extract the prologs from the source code.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 7 - The student will compile Product Generation Executives (PGEs) and link SCF and DAAC version of Science Data Processing (SDP) toolkits and update files by configuring the SDP Toolkit environment, updating the process control file and compiling status message facility files.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 8 - The student will run a PGE executable in a simulated SCF environment and determine resource requirements for the PGE.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 9 - The student will examine the PGE-produced log files from the run in the simulated SCF environment.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 10 - The student will perform Hierarchical Data Format (HDF), ASCII, and binary file comparison using the GUI and/or hdiff and will use EOSView to examine data products. Refer to the procedures in the File Comparison and Data Visualization section.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 11 - The student will identify and register Earth Science Data Types (ESDTs) for use in the system.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 12 - The student will insert a Science Software Algorithm Package (SSAP) into the Planning and Data Processing Subsystems' database (PDPS database).

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 13 - The student will review the production rules and production requests for compliance with the particular PGE being worked on and also DPREP as an additional PGE effort where necessary.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 14 - The student will update the PGE, ESDT, and Orbit ODL Templates as necessary to comply with the specific PGE being prepared.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 15 - The student will update the PDPS Database by inserting science ESDT metadata, science PGE metadata and operational metadata into it and will update the data server by inserting science data granules, static files and executables.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 16 - The student will subscribe to input and output data granules; submit a Production Request (PR) to run a PGE; use the Planning Workbench to plan, schedule and activate the PR; and monitor the processing under AutoSys.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Specific Objective 17 - The student will use the procedures to locate and examine data products and the Production History File.

Condition - The student will be given an operational system with supporting equipment.

Standard - The student will use the tools in accordance with prescribed methods and complete required procedures without error.

Importance

The Science Software Integration and Test lesson provides a review of the process that allows the SSIT team, Production Planner and Production Monitor to retrieve, install, check, compile, test and monitor science software.

Science Software Configuration Management

The CM Administrator and System Administrator are key players in the SSI&T process. The CM Administrator receives the science software from the Science Data Specialist, places these files into a directory and request that the System Administrator place the files under configuration control by using the ClearCase tool. The science software is then tested by the SSI&T team and once the science software has successfully been tested, and upon direction from the CCB, the files are distributed to the Production Planner for placement on production server.

The CM and System Administrator need a good understanding of the ClearCase tool. ClearCase will be used to create a view, create a new directory, import files into the temporary subdirectories, and check in and check out files.

ClearCase Overview

All data managed under ClearCase are stored in Versioned Object Bases (VOBs), which are the “public” storage areas and Views, which are the “private storage areas. VOBs are data structures that can only be created by the CM administrator using the mkvob (“make vob”) command. A VOB is mounted as a file system and when viewed through a view, it appears as a standard UNIX directory tree structure. This file system, accessed through its mount point, has a version-control dimension that contains file elements and versions of file elements. Once reviewed, the System Administrator will place these files under configuration control. In order to accomplish this task, a view must be created in ClearCase. A view is necessary in order to make visible and accessible files and directories that have been checked in to a VOB.

Data that are under configuration management in ClearCase are said to be “checked in”. In order to alter a checked-in data element (e.g. a file) to make a newer version of it, the data element must first be “checked out”. Once the change has been made to the checked- out version, it is checked in again. The VOB will then contain both versions of the data element and either can be retrieved at a later date.

In general, executable binary files, object files, and data files should not be checked into ClearCase. Binary and object files are not stored efficiently in ClearCase; data files for software may be extremely large and a VOB is typically not sized for this.

Files that should be checked into ClearCase include source code, scripts, makefiles, assorted build and run scripts, documentation and other ASCII files.

A Versioned Object Base is defined by the following characteristics:

- A mountable file system that stores version-controlled data, such as source files, binary files, object libraries, WYSIWYG documents, spreadsheets and anything that can be stored in the UNIX file system.
- Can be mounted on some or all workstations
- Several VOBs may exist on a machine or on different machines on a network.
- When mounted as a file system of type MFS, a VOB can be accessed with standard UNIX and ClearCase tools.
- The ClearCase file system is transparent.
- Created by the CM administrator

A VOB is comprised of:

- Storage area for versioned files, derived objects and clear text files.
- Database (live, shadow and log file).

Creating a View in ClearCase

In order to make files and directories that are in a ClearCase VOB visible and accessible a ClearCase view must set. A ClearCase view need only be created once. Once created, the view can be set at the beginning of each user session. Multiple views for a single user may be created.

In order for the SSI&T tools under the SSIT Manager to have access to the ClearCase VOB, the ClearCase view must be set before the SSIT Manager is run.

A view is defined by the following characteristics:

- A working context for an individual developer or closely coordinated group.
- Can be used to access any VOB or multiple VOBs.
- Selects versions of VOB directories and files to display.
- Allows developer to work without interfering with other developers.
- Not a set of files but a way of seeing shared elements.
- Each user may have multiple views for new development, bug fixing or porting activities.

A view is comprised of:

- View storage area (typically in a local machine) - private storage for checked-out files, derived objects and private files.
- Configuration Specification - set of rules that determine the version of a file the view will see.

- View-tag - Name given to the view (e.g., `angies_view`). View-tags are registered in `/urs/adm/atria/view_tags`.
- Objects stored in a view:
- Checked-out versions of file elements.
- Unshared derived objects.

The ClearCase procedures can either be run from the UNIX command line or from the File Browser Screen. The SSI&T Training will only cover the UNIX command line procedures. The corresponding GUI procedures are included in the Training Material for future reference.

The following procedure not only will create a view, but will also allow creation of a subdirectory where new science software files may be stored.

Assumptions:

- ClearCase is available.
- A Versioned Object Base (VOB) has been created.

Create a View in ClearCase Using Command Lines

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - 2 Enter the **password** then press the **Enter** key.
 - 3 At a UNIX prompt type **cleartool lsview**, then press the **Enter** key.
 - The `lsview` command displays the pathname to the storage location of the views.
 - 4 At a UNIX prompt type **cleartool mkview -tag *ViewName* *ViewPath/ViewName.vws***, then press the **Enter** key.
 - The *ViewPath* is the full path to the directory where views are stored.
 - The *ViewName* is the user-selected name for the view. The file name for the view must end in “.vws”.
 - 5 Displays the directory name of the current VOB, just below the toolbar.
 - 6 Displays the content of the directory in the space below the directory’s name.
-

For future reference, the corresponding ClearCase GUI procedures are included in the following section. Selecting a view listed in the View Tag Browser screen brings up the File Browser, or main screen, shown in Figure 1.

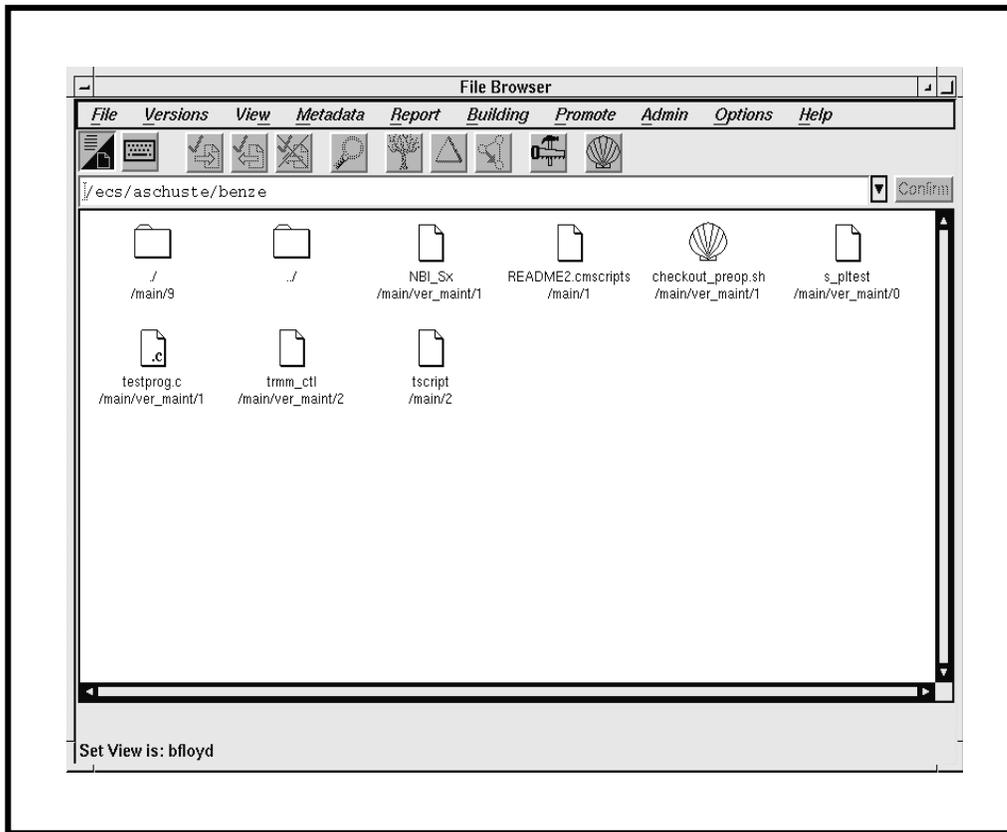


Figure 1. ClearCase File Browser Screen (Main Screen)

Create a View in ClearCase Using the File Browser Screen

- 1 The user should log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - Cursor moves to the Password field.
- 2 Type the **password** then press the **Enter** key.
- 3 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
 - The ClearCase Transcript screen is displayed as the View Tag Browser loads.
 - The ClearCase View Tag Browser screen is displayed listing available views.
- 4 To create a view for checking in the software change package, select a known View and press the **Enter** key.
 - The File Browser window is displayed.

- 5 Select **File**→**Execute**→**Single Command**.
 - The **String Browser** window is displayed.
 - The prompt **Enter shell command to run** is displayed.
 - 6 Invoke the make view command by typing **mkview [filename]** on the UNIX command line and press the **Enter** key.
 - The **tempdisp** window appears.
 - The **View [filename] Created Successfully** and the **Cache Updated for View [filename]** prompts are displayed.
 - 7 Close the **tempdisp** window by clicking on the window and press the **Enter** key.
 - The **tempdisp** window closes.
 - 8 Select **View** →**List** from the menu.
 - The **View Tag Browser** is displayed.
 - 9 Find the new view by scrolling through the list until the new view is observed.
-

Setting a View in ClearCase

In order to make files and directories that are in a ClearCase VOB visible and accessible, a ClearCase view must be set. Only one view can be set (active) at a time.

Set a View in ClearCase Using Command Lines

- 1 Log into one of the AIT Sun workstations by typing: *username* then press the **Enter** key.
 - 2 Enter the *password* then press the **Enter** key.
 - 3 At a UNIX prompt type **cleartool setview ViewName** where *ViewName* is the user's view created in the previous section, then press the **Enter** key.
-

Set a View Using the File Browser Screen in ClearCase

- 1 Log into one of the AIT Sun workstations by typing: *username* then press the **Enter** key.
 - Cursor moves to the Password field.
- 2 Type the *password* then press the **Enter** key.

- 3 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
 - The ClearCase Transcript screen is displayed as the View Tag Browser loads.
 - The ClearCase View Tag Browser screen is displayed listing available views.
 - 4 To set a view, select a known View and press the **Enter** key.
 - The File Browser window is displayed.
 - 5 Select **File→Execute→Single Command**.
 - The String Browser window is displayed.
 - The prompt Enter shell command to run is displayed.
 - 6 Invoke the set view command by typing **setview ViewName** on the UNIX command line and press the **Enter** key.
 - ViewName is the name of the view to set.
-

Creating a New Directory

In cases where a new directory needs to be created and placed in ClearCase, the user will activate ClearCase and create a new directory. This type of procedure is necessary only if a new directory is required.

The following is a list of tools and/or assumptions:

- A VOB has been created at the UNIX directory.
- A view has been created.

Create a New Directory in ClearCase Using Command Lines

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
- 2 Enter the **password** then press the **Enter** key.
- 3 At a UNIX prompt type **cleartool setview ViewName**, then press the **Enter** key.
 - The *ViewName* is the user's view.
- 4 At a UNIX prompt type **cleartool lsvo**, then press the **Enter** key.
 - This command lists all the VOBs and allows the identification of the SSI&T VOB.

- 5 At a UNIX prompt type **cd *pathname***, then press the **Enter** key.
 - The *pathname* is the full path name of the parent directory in the VOB in which the new directory is to be added.
 - 6 At a UNIX prompt type **cleartool checkout -nc .** then press the **Enter** key.
 - This command checks out the current directory. Note the dot for the directory.
 - The **-nc** is a keyword used when no comments are to be made for this action.
 - 7 At a UNIX prompt type **cleartool mkdir -nc *dirname***, then press the **Enter** key.
 - The *dirname* is the name of the new directory being created.
 - 8 At a UNIX prompt type **cleartool checkin -nc *dirname***, then press the **Enter** key.
 - This command checks in the new directory named *dirname*.
 - 9 At a UNIX prompt type **cleartool checkin -nc .** then press the **Enter** key.
 - This command checks in the current directory.
-

Enter a New Directory Using the File Screen Browser into ClearCase

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - Cursor moves to the Password field.
- 2 Type the **password** then press the **Enter** key.
- 3 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
 - The ClearCase Transcript screen is displayed as the View Tag Browser loads.
 - The ClearCase View Tag Browser screen is displayed listing available views.
- 4 Select **File→Execute→Single Command**.
 - The String Browser window is displayed.
 - The prompt Enter shell command to run is displayed.
- 5 Invoke the make directory element by typing **mkdir [filename]** on the UNIX command line and press the **Enter** key.
- 6 Invoke the make element command by typing **mkelem *directory*** on the UNIX command line and press the **Enter** key.

- 7 Type into the directory input box of the **File Browser** the name of the directory in the VOB to be checked out, press the **Enter** key, then follow the menu path **Version→Checkout→Reserved: no comment**.
 - In order to add new files to ClearCase, the directory in which the files are to be added must be checked out first.
 - ClearCase forces the checkout onto a maintenance branch to isolate the maintenance activity.
 - If someone else has already checked out the directory, permission to check out the directory is denied.
 - A separate shell window is displayed.
 - 8 Cancel the checkout of the element if it is decided that no changes are to be made by typing into the directory input box of the **File Browser** the name of the directory to be checked in, press the **Enter** key, then follow the menu path **Version → Uncheckout → Unreserved: no comment**,
 - 9 On the **File Browser** screen, follow the menu path **File→Exit**.
 - The ClearCase Graphical User Interface session is closed.
-

Importing Files into ClearCase

Once the user has created a directory to place the science software files, ClearCase can be used to place a single file or multiple files in a UNIX directory structure under CM.

The following is a list of tools and/or assumptions:

- A VOB and subdirectory are created to hold these files.
- No object files or executables exist in the source code directory.
- The PGE was received with a directory structure that contains various types of files.
- These files will be entered into ClearCase and will maintain the same directory structure as the delivery structure.

The first procedure to review will be importing a single file. The following procedure review covers importing multiple files in a UNIX directory structure.

Import a Single File into ClearCase

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - Cursor moves to the Password field.

- 2 Type the **password** then press the **Enter** key.
- 3 At a UNIX prompt, type **cleartool setview *ViewName*** then press the **Enter** key.
 - The *ViewName* is the name of the ClearCase View.
- 4 At the UNIX prompt, type **cd *pathname*** then press the **Enter** key.
 - The *pathname* is the full path name of the subdirectory in the VOB into which the file is to be checked in.
 - If the desired directory cannot be seen, it could mean that the view has not been set or the properties of the view do not allow the directory to be seen; check with the CM Administrator.
- 5 At a UNIX prompt, type **cp *pathname/filename* .** then press the **Enter** key (note the space and then “**dot**” at the end of the command).
 - The *pathname* is the full path name to the directory where the file to be checked in exists and *filename* is the file name of the file to be checked in.
 - This command copies a file over into the VOB area in preparation for checking it in.
- 6 At the UNIX prompt, type **cleartool checkout -nc .** then press the **Enter** key (note the space and then “**dot**” at the end of the command).
 - This command checks out the current directory (represented by the “**dot**”) from ClearCase.
 - Adding a new file (or element) to a directory represents a modification of the directory. Hence, the directory must be checked out before a file can be checked in.
- 7 At a UNIX prompt, type **cleartool mkelem -nc *filename***, then press the **Enter** key.
 - The *filename* is the name of the file that was copied over in Step 5 and is the file that will be checked into ClearCase.
 - This command creates a ClearCase element from the file in preparation for checking it in.
 - The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the make element step.
- 8 At the UNIX prompt, type **cleartool checkin -nc *filename***, then press the **Enter** key.
 - The *filename* is the name of the file to be checked into ClearCase.
 - This command performs the check in of the file.
 - The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the checkin step.

- 9 At the UNIX prompt, type **cleartool checkin -nc .** then press the **Enter** key (note the space and then “**dot**” at the end of the command).
- This command checks in the current directory (represented by the “**dot**”) into ClearCase.
 - The adding of an element (**here, a file**) represents a modification to the directory and hence, the new version of the directory must be checked back in.
 - The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the checkin step.
-

Importing Multiple Files into ClearCase

The importing of multiple files into ClearCase will not be performed during the SSI&T Training Lesson. The DAP for the synthetic PGE contains only one source code module and a minimal number of other files. A real PGE will generally contain many source files, header files, and multiple other types of files stored in a standard type of directory structure that is retained when the PGE is packed into the tar file. The script provided by ClearCase is used for the purpose of making another load script to enter all of the DAP files along with the directory structure at one time. The final step of running the load script can only be performed by the DAAC Administrator.

The following procedure explains how to place the entire contents of a UNIX directory structure under ClearCase. A UNIX directory structure refers to all the files and subdirectories under some top-level directory.

This procedure is geared toward science software deliveries. In such cases, science software is delivered in the form of a UNIX *tar* files. A *tar* file has been unpacked (**untarred**) and the contents are to be placed under ClearCase configuration management.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

The following is a list of tools and/or assumptions:

- A VOB and subdirectory are created to hold these files.
- A ClearCase view is not required to perform this procedure.

Import Multiple Files into ClearCase

- 1 At a UNIX prompt, type **cd *ParentPathname***, then press the **Enter** key.
- The ***ParentPathname*** is the path name of the directory that ***contains*** the directory structure to be brought into ClearCase. This is ***not*** the VOB.

- 2 At the UNIX prompt, type **clearcvt_unix -r *DirName*** then press the **Enter** key.
 - The ***DirName*** is the name of the directory in which it and everything below it is to be brought into ClearCase.
 - A conversion script will be then be created. The **-r** causes all subdirectories to be recursively included in the script created.
- 3 Contact the VOB Administrator and request that the utility script **cvt_script** be run on the script created in Step 2.
 - The VOB Administrator is the only one who can run the **cvt_script** because it modifies the VOB.
- 4 At this time the user logs out from this workstation. The VOB Administrator completes the procedure.
 - The remaining steps are accomplished by the VOB Administrator.
- 5 The VOB Administrator logs into the AIT Sun workstation by typing **username** then press the **Enter** key.
 - Cursor moves to the Password field.
- 6 Type the **password** then press the **Enter** key.
- 7 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
 - The ClearCase Transcript screen is displayed as the View Tag Browser loads.
 - The ClearCase View Tag Browser screen is displayed listing available views.
- 8 To create a view for checking in the software change package, select a known View and press the **Enter** key. If you are using an existing view, select the desired existing view and proceed to Step 14.
 - The File Browser window is displayed.
- 9 Select **File → Execute → Single Command**.
 - The String Browser window is displayed.
 - The prompt Enter shell command to run is displayed.
- 10 Invoke the make view command by typing **mkview [filename]** on the UNIX command line and press the **Enter** key.
 - The tempdisp window appears.
 - The **View [filename] Created Successfully** and the **Cache Updated for View [filename]** prompts are displayed.

- 11 Close the **tempdisp** window by clicking on the window and press the **Enter** key.
 - The tempdisp window closes.
 - 12 Select the VOB where the software change package is to be imported then press the **Enter** key.
 - 13 To create a subdirectory for the software change package in that VOB, which is a modification to the parent directory (for the VOB) the parent directory must be checked out by following the menu path **Version→Checkout→Reserved: no comment**.
 - In order to add new files to ClearCase, the directory in which the files are to be added must be checked out first.
 - ClearCase forces the checkout onto a maintenance branch to isolate the maintenance activity.
 - If someone else has already checked out the directory, permission to check out the directory is denied.
 - A separate shell window is displayed.
 - 14 Start a shell process in a separate window by clicking on the shell icon button of the **File Browser** toolbar.
 - A separate shell window is displayed.
 - 15 To run the script type **cvt_script** then press the **Enter** key.
 - The VOB Administrator is the only person who can run the **cvt_script** because it modifies the VOB.
 - 16 To check in the new directory type *path* into the directory input box of the **File Browser** screen then press the **Enter** key.
 - *path* is the full path identification for the new directory.
 - 17 Select **Versions→Checkin** from the menu.
 - 18 To check in the parent directory (for the VOB), type *VOBpath* into the directory input box of the **File Browser** screen then press the **Enter** key.
 - *VOBpath* is the full path identification for the parent directory.
 - 19 Select **Versions→Checkin** from the menu.
 - 20 On the **File Browser** screen follow menu path **File → Exit**.
 - The ClearCase Graphical User Interface session is closed.
-

Checking Out a File from ClearCase

If a configured file requires modification, then the file needs to be checked out of the configured directory and placed in a user directory. This will allow the file(s) to be modified.

The following is a list of tools and/or assumptions:

- The file or directory must be an element created in ClearCase.
- The view should be configured to ensure the correct version of the file or directory is seen.

Check Out an Element/File from the Command Line

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - 2 Enter the password then press the **Enter** key.
 - 3 At a UNIX prompt type `cleartool setview ViewName`, then press the **Enter** key.
 - The *ViewName* is the name of the user's view.
 - 4 At a UNIX prompt type `cleartool checkout -nc element` then press the **Enter** key.
 - The *element* is the name of the file or directory that is to be checked out.
 - The **-nc flag** means “no comment” which will suppress the ClearCase prompting for a comment to be associated with the check out step.
-

Check Out an Element/File from the File Screen Browser

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
 - Cursor moves to the Password field.
- 2 Type the **password** then press the **Enter** key.
- 3 Invoke ClearCase GUI by typing `xclearcase &` on the UNIX command line then press the **Enter** key.
 - The ClearCase Transcript screen is displayed as the View Tag Browser loads.
 - The ClearCase View Tag Browser screen is displayed listing available views.

- 4 To check out the directory where the controlled files were place, type *path* into the directory input box of the **File Browser** screen then press the **Enter** key.
 - *path* is the full path identification for the directory.
 - 5 Select **Versions**→**Checkout** from the menu.
 - 6 Select **File**→**Execute**→**Single Command**.
 - The String Browser window is displayed.
 - The prompt Enter shell command to run is displayed.
 - 7 To determine editing privileges, type **ls -l**, then press the **Enter** key.
 - A prompt displaying read/write/execute privileges will be displayed. There will be three groupings; i.e., **User**, **Group**, and **Others**.
 - Codes for read/write/execute privileges are as follows:
 - **r** = read.
 - **w** = write.
 - **x** = execute.
 - 8 If you have editing/execute privileges, you can revise the contents of the file with any text editor.
 - 9 To check in a controlled file, select **Versions**→**Checkin** from the menu.
 - The file/directory will be checked in to ClearCase and the version will be updated.
-

Checking a Modified Element into ClearCase

This procedure explains how to check in a modified element to ClearCase. An element refers to a directory or file in ClearCase, that is, under configuration management. Modifications made to a file or directory cannot be saved in ClearCase unless the file or directory had been checked out first.

The following is a list of tools and/or assumptions:

- A VOB exists and is mounted at a known UNIX directory.
- A ClearCase view exists for the SSI&T operator.
- The element or file has been checked out and modified.
- The modified file is now in the user's directory on the VOB from which it was checked out.

Check a Modified Element/File into ClearCase

- 1 Log into one of the AIT Sun workstations by typing: *username* then press the **Enter** key.
 - Cursor moves to the **Password** field.
 - 2 Type *password* then press the **Enter** key.
 - 3 At a UNIX prompt, type **cleartool setview *ViewName*** then press the **Enter** key.
 - The *ViewName* is the name of the user's view.
 - 4 At the UNIX prompt, type **cleartool checkin -nc *filename***, then press the **Enter** key.
 - *filename* is the name of the file (full path name allowed) that is to be checked out (and later modified).
 - The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
 - This command checks in the current directory.
 - 5 This step is optional; it is performed when ClearCase does not accept a checkin because the element was not modified. In this case, the check out must be canceled. At a UNIX prompt, type **cleartool uncheckout -nc *filename***, then press the **Enter** key.
 - *filename* is the name of the file or directory (**full path name allowed**) checked out.
 - This command cancels the check out of an element/file.
-

This page intentionally left blank.

Science Software Integration and Test (SSI&T) Preparation, Processes and Setup

Key Operator Roles

Science Coordinator: Provide support to Instrument Teams for the integration and testing of science software in the production system at the DAAC. Perform standard checking on all delivered software including source code, scripts, process control files and related documentation.

Science Data Specialist: Serves as a point-of-contact for planning, integrating, testing, and operating science software.

CM Administrator: Record, report, manage and distribute new and updated science software.

Science Software I&T Support Engineer: Provide support to Instrument Teams for the development, integration, test and problem resolution of science software.

Production Planner: Populate, maintain and schedule the production planning database for science software.

COTS Software Tools

ClearCase: This tool is used as the custom software configuration management tool. ClearCase provides a mountable file system that is used to store version-controlled data, such as source files, binary files, object libraries and spreadsheets.

Distributed Defect Tracking System (DDTS): This tool is used to electronically process configuration change requests (CCRs). DDTS will prompt the user for relevant information, identify the request and will mail these requests to pre-designated personnel.

MODIS Science Data Processing Software Version 2.0 System Description Manual: This manual should be referred to for more detailed information on how to perform the SSI&T operational procedures as they apply to MODIS PGEs. It covers the specific attributes for each individual PGE and setup criteria.

Operations Tools Manual

The SSI&T operational procedures are given in many of the sections that follow. They are organized by activity. The order in which the procedures appear loosely follows the order in which they will usually be performed.

These procedures present the use of GUIs. Some procedures may have a command line equivalent; these are documented in the corresponding GUI help screens but are not presented here in the interest of simplicity. Refer to 609-EMD-001, *Release 7.10 Operations Tools*

Manual for the EMD Project, for more detailed information on how to use GUIs and command line equivalent usage.

General Process

The SSI&T process consist of two activities:

- Pre-SSI&T activity - During this activity the Delivered Algorithm Package (DAP) is inspected, and tested in a non-production environment.
- Formal SSI&T activity - During this activity, the Product Generation Executives (PGEs) are integrated with the DAAC version of the SDP Toolkit and executed on the PDPS platform.

Key Terms:

- **Product Generation Executives (PGEs)** - The smallest scheduled unit of science software.
- **Delivered Algorithm Package (DAP)** - An ensemble of PGE source code, makefile, documentation, and other related files delivered in a package from the SCF to the DAAC for SSI&T.
- **Process Control File (PCF)** - Relate logical identifiers to physical files and other parameters required by the PGE.
- **Strings** - The processing hardware on which the science software runs.
- **Archive** - A File Storage Type indicating that granules that will be inserted to Data Server are intended for long-term storage and acquisition for distribution.
- **Collection** - A related group of data granules.
- **Granule** - The smallest data element, which is identified in the inventory, tables.
- **Product** - A set of output values generated by a single execution of a PGE for archiving by the production system. A PGE may generate one or more products whose attributes are defined by the data provider.
- **Reliability** - Software reliability means that the software runs to normal completion repeatedly over the normal range of data inputs and running conditions.
- **Safety** - Software safety means that the software executes without interfering with other software or operations.

The science software in the DAPs will be integrated onto the PDPS and be used to produce the output data as determined by the algorithms. The refined and updated DAPs and data produced by the science software will eventually be provided to the subscribing user. Before the PGE is integrated into a production environment, extensive testing on the software must be performed.

Science Software Integration and Test (SSI&T) is the process by which the science software is tested for production readiness in the DAACs in order to assure its (1) reliability and (2) safety.

Prior to the delivery of the software to the DAACs, SSI&T Checkout is conducted on early versions of the Product Generation Executives (PGEs) using separate system modes in the Performance Verification Center (PVC), Verification and Acceptance Test Center (VATC), or the DAAC environments.

SSI&T activities can be broadly separated into two categories: pre-SSI&T and formal SSI&T. Pre-SSI&T activities are those which do not involve the Planning and Data Processing Subsystems (PDPS) or the Science Data Server (SDSRV), but the formal SSI&T activities do involve the full production system including the PDPS and the SDSRV.

Most steps in the SSI&T process are inter-related and some steps may assume that another step has been completed. The ordering of the steps is very important but it cannot, however be interpreted as a detailed, step-by-step guide to SSI&T activities. Refer to the Science Office Project Instructions in the EMD Process Assets Library (PAL). The Process Assets Library is accessible on the web at the following URL:

http://dmserver.hitc.com/EMD_PAL/index.html

Perform Pre-SSI&T Activities

- 1** As the DAP is delivered to a DAAC by the Instrument Team for SSI&T, the PGE listing documentation is reviewed.
- 2** The DAP is acquired and unpack and the documentation (i.e., packing list, readme, etc.) checked. The DAP contents are further checked by the Science Data Specialist to verify that the contents match the packing list, agreed-upon directory structures are employed, location of files are correct, and all intended files and directories are present.
- 3** The Science Data Specialist requests that the CM Administrator place the DAP under Configuration Management control using ClearCase.
- 4** The SSI&T team checks the science software for standards compliance using the Process Control File Checker to check process control files (PCF), and the Prohibited Function Checker to check source files. Extract and check prologs.
- 5** The SSI&T team builds the science software into PGEs using the SCF version of the SDP Toolkit. Compile all source code. Link object code with appropriate libraries. . If the SMF files compile successfully, then proceed to Step 11 below; otherwise, the problem needs to be fixed and a successful compile must occur before proceeding further. This may require one or more of the following:
 - Note any error messages and review the included documentation to ensure a proper compile;
 - Check environmental variables;
 - Review the setup files for proper directories and variables;
 - Make corrections and recompile.

- If the executable builds successfully, proceed to Step 12. If the build fails, it may be necessary to do one or more of the following before proceeding:
 - Check environmental variables;
 - Make corrections and repeat the build.
- 6** Run the PGE from the Command Line.
- If it the execution is successful, then the output files (products) are checked using the SSIT Manager file comparison tools; otherwise, one or more of the following needs to be done before proceeding:
 - Check error logs;
 - Check environmental variables;
 - Review and source the setup files;
 - If necessary files are missing, then review the PCF file to ensure the existence of correct reference directories.
- 7** The SSI&T team runs and profiles the PGEs from the UNIX command line on the SGI, saving the profiling results. They will be used later when entering operational metadata into the PDPS.
- 8** The SSI&T team collects performance statistics for the PGEs.
- 9** The SSI&T team examines the output log files from the PGE runs for any anomalous message. The SSI&T team compares the output product data with the delivered test data using the file comparison tools. If the products do not match the delivered test outputs (expected outcome), the outputs should be analyzed and the PGE must be re-run. If the products match the delivered test outputs then
- NOTE:** Steps 10 through 12 are repeated once using the DAAC Toolkit. If the products generated with the DAAC Toolkit match the delivered test output, formal SSI&T may begin.
- 10** SSI&T team reports any science software problems using the DDTS NCR process.
- 11** The SSI&T team reports any production system problems using the DDTS NCR process.
- 12** The SSI&T team collects and logs all lessons learned.
-

Perform Formal SSI&T Activity

- 1** For each ESDT used by the PGE, construct an ESDT ODL file for updating the PDPS or verify that they already exist. ESDT ODL files are also needed for all input and output data granules.

- 2 Construct a PGE ODL file for updating the PDPS database. This involves using the delivery PCF to construct an initial PGE ODL template file, which must then be hand edited to add required metadata. A mapping between logical IDs in the PCF and ESDT ShortNames must be known before this step is done.
- 3 Install ESDTs on the Science Data Server if verification indicates that they do not already exist. Installation links the PGE to all input and output ESDTs that allows the PGE to run within the PDPS. Note: While installing ESDTs the SDSRV intermittently core dumps. To clean up you must remove the ESDT from SBSRV and DDICT and then try again.
- 4 The Science Metadata.met is updated (PGE & ESDT Object Description Language or ODLs are created). This supplies metadata to the PDPS database
 - If the Science Metadata update is successful, then the Operational Metadata is updated; otherwise, the ESDT ODL files may have to be checked for correctness before updating the Operational Metadata.
- 5 Register the PGEs with associated data in the PDPS database. This step uses the PGE ODL from Step 2 above.
- 6 For each input dynamic data granule needed by the PGE, construct a Target MCF and insert both granule and .met files into the Science Data Server.
- 7 For each input static granule needed by the PGE, construct a Target MCF.met file and insert both to the Science Data Server.
- 8 Assemble the SSEP (as a tar file) and insert it to the Science Data Server.
- 9 Initiate a Production Request (PR) that will result in one or more DPRs.
- 10 Use the Planning Workbench to plan the PR and hence, run the PGE.
- 11 Monitor the PGE run using AutoSys. The PGE's progress is monitored using the AutoSys COTS.
- 12 If any of the steps in the execution is not successful, then each failure must be identified and corrected before proceeding to the next step.
- 13 Examine the output Production History File from the PGE runs for any anomalous messages. Compare the output product data with the delivered test data using the file comparison tools. . If any of the steps in the execution is not successful, then each failure must be identified and corrected before proceeding to the next step.
- 14 If the output files match the test output files and they are in Hierarchical Data Format (HDF), they are visualized using the EOSView tool, or the Interactive Display Language (IDL) tool. If the files are not HDF, then IDL is used.
- 15 Using the Planning Subsystem, initiate more complex Production Requests if chaining is required.

- 16 Using electronic or hard media transfer methods, distribute the data products to the Instrument Teams for their review.
-

Records

A weekly SSI&T status report is provided to NASA. This report contains the Performance Measurement Data.

Performance Measurements

SSI&T PGEs planned vs. actually delivered, pre-tested, and integrated is the metric used to monitor the effectiveness of the process described in the Procedure. Additionally, the Duration of Effort Required to Integrate in Work Days is used.

Prepare and Set Up to Use the SSI&T Manager Tool

- 1 Log into one of the AIT Sun workstations by typing: *username* then press the **Enter** key.
- 2 Type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - You may need to setup the terminal so that the remote host is displayed on your screen. (Sun machine) This is done by clicking on the Application Manager icon (the file drawer located at the bottom of the screen), followed by the Desktop Tools icon, followed by the Terminal Console icon.
- 3 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - The **Enter Password** prompt is displayed.
- 4 Type *password* then press the **Enter** key.
- 5 Enter the directory where the setup script is located by typing **cd *directory_name*** then press the **Enter** key.
- 6 To source the setup script type **source *script_name*** then press the **Enter** key.
 - The setup script contains directory paths, sets of alias commands, and tools for SSI&T.
 - For example, to source the SSI&T script type **source /usr/ecs/*MODE*/CUSTOM/utilities /.buildrc** then press the **Enter** key.

NOTE: This step only needs to be done once per login.

NOTE: To ensure access to the multi server environment when needed, the following generic login command has been established and should be used routinely:

- 7 From a terminal type **xterm -n host &** then press the **Enter** key.
- 8 From the xterm invoked type **ssh host** then press the **Enter** key.
- 9 At the **Password** prompt type **password** then press the **Enter** key.
- 10 Type **cd /usr/ecs/MODE/CUSTOM/utilities/.buildrc** then press the **Enter** key.

NOTE: Listed are some of the GUI tools, typical servers and their hosts that need to be considered for activation when conducting SSI&T:

- ECS Assistant, DM, x0ins02
 - ECS Assistant, SDSRV/DSS, x0acs03
 - ECS Assistant, DPS, x0sps06
 - ECS Assistant, SBSRV/CSS/, x0ins01
 - SSIT Manager tools, AITTL/DPS x0ais01
 - Production Request, PLS, x0pls01
 - Planning Workbench, PLS, x0pls01 (**Note:** NETSCAPE should be closed to allow for a full screen GUI to be activated.)
 - Monitor PGE, DPS, x0sps06 using AUTOSYS
- 9 A second xterm should be activated with the same login procedures so as to monitor the (log files) when entering SSI&T files from GUI's.
 - 10 Servers can be brought down in any order. To bring them backup requires that they be brought up in a **sequential order to ensure connectivity**, the order is listed as follows:
STMGT, MSS, DDIST, SDSRV, PDPS
 - 11 The above servers have unique hosts assigned. Each host needs to be logged into before activating ECS Assistant to bring down and bring up servers assigned to their respective hosts.
-

Science Software Integration & Test Flow

Software is developed at the SCFs using the SDP Toolkit. It is shipped to the DAAC for testing in a package that includes all source code, make files, a version of the Process Control File (for the Toolkit), a Metadata control file (for the Toolkit), some files describing the PGE Profile (this data will be turned into the ODL files that are ingested by AIT to form the PGE registration), documentation, test plans, and test files. It may be shipped via ftp, hard media, or some combination there of. Note that some of the data (particularly large test data files) may be

shipped separately from the rest of the stuff. Since all the DAPs are retrieved by the SSIT staff, there shouldn't be any problem integrating a partial DAP with the software, test files, etc.... that go along with it. Note that it is possible that the SCFs will want to create the ODL files and ship them with the DAP. While this is possible, the PGE Metadata ODL file template is created via an SSIT tool by reading the PCF. Then this template and the PGE Metadata ODL file would have to be merged some how. There may have to be some kind of turnover form or PGE worksheet for the Instrument Teams to specify the information needed in the ODL files.

- Software is ingested into the system (by INGEST) and placed on the Data Server (in TS1 mode). The AIT workstation has a subscription on all new DAPs that is created by the Subscription Editor.
- A ClearCase work area is created on the SSIT machine(s) for testing and correcting the software. The source code will be maintained in ClearCase to support versioning. The actual compilation and execution of the source code may be done on the SSIT workstation (currently slated to be a Sun), but will eventually have to be done on the target software (one of the science processors—currently slated to be an SGI). To test software on a science processor, the machine must be “signed out” by the Resource Planner.
- The DAP (Delivery Archive Package) is acquired from the Data Server. This is done through an SSIT script tool. It is unpacked (via “tar” commands) at the SSIT machine. This is currently a manual process, though we could automate it in the same script that is used to retrieve the DAP.
- The files of the DAP are placed in ClearCase for configuration management. Since the DAP is just a big tar file, or series of tar files, it has (included within it) the directory structure of the code. When the code is extracted, it will be placed (via untar) into the correct directory structure.
- The PGE is compiled using SDP Toolkit tools, and verified using a combination of Toolkit and SSIT custom functions (Prohibited Function checker, Fortran77 checker). SSIT tools are brought up from the SSIT GUI menu. First the SCF version of the Toolkit is used, then the DAAC version of the Toolkit is used to verify the compilation and tests. The compilation of the PGE will take place on the target hardware (science processors), not the SSIT workstation(s).
- Tests are performed as they were at the SCF. Test input is put into local directory on the machine where the test is being performed and then the PGE is run from the command line using the delivered PCF (with minor modifications).
- Any problems found can be fixed on the local copy of the source code. Tests will be rerun to verify completeness of the fix. Errors found are entered into DDTS for tracking. Once the command line tests of the PGE have completed successfully, then the executables must be stored at the Data Server (running in SSIT mode) so that they can be retrieved by PDPS.

The PGE executable and any associated binary files (Toolkit files or any binaries called by the main program of the PGE) are stored on the Data Server as tar files. DPS will stage and extract the tar file when the PGE is scheduled to run. The tar file is untarred in one directory, with the profile (any environment variables that must be set), top-level shell (main program of the PGE) and the Toolkit SMFs (message files) in the main directory. This is used both when testing the PGE in a non-operational PDPS system (with Data Server in “SSIT” mode) and in the production system (in “OPS” mode). There is a script to insert the exe tar file into the Data Server, as well as any static files needed by the PGE.

When standalone tests have completed successfully, information about the PGE is entered into the PDPS Database (a copy resides on the AIT hardware) so that it can be tested in the Production Environment. Input and Output Data Types, Production Rules, etc. are supplied by the ODL files. The PGE ODL file is generated from the PCF but must be completed by the SSIT personnel. ODL files must also be created for the ESDTs used by the PGE and any production rules. A SSIT tool parses the ODL files and places the data in the PDPS database. The PGE Registration GUI is used to finish the information about the PGE, allowing the input of Performance Statistics and Resource Requirements (collected via EcDpPrRusage) and any text description for User Parameters.

The original PCF (that was part of the DAP) is NOT used by DPS when the PGE is run in PDPS. All information from the original PCF is captured in the PDPS database (as part of the PGE profile). DPS uses the entries in the PDPS database to create a PCF each time the PGE is run. The original PCF will be saved in the SSAP. There is an SSIT Tool that takes as input the PCF of the PGE and then generates a template ODL file.

The original PCF must be changed before the ODL files are created. Any entries of Attitude/Ephemeris/DEM data in the PCF are removed if the source code is found to NOT have the corresponding Toolkit calls to access the data. Also, any PCF entries for binary metadata output files (called “.met” files) will be removed and replaced with a Runtime Parameter (at the same logical id) that tells the Toolkit the logical id of the binary output for which the ASCII “.met” file is for. This allows the Toolkit to name the “.met” file after the binary output and helps DPS later.

PGE is tested in the production system. The PLS and DPS software will coexist on the SSIT hardware and will be brought up there in “SSIT” mode. SSIT hardware will also include AutoSys for processing. A science processor will have to be signed out to allow the tested PGE to be executed.

DAAC and SCF personnel agree that the PGE performs properly. It is accepted as “product ready” software.

The Science Software Archive Package for the PGE is defined. All source code, documentation, test plans, scripts, test data (maybe only a pointer to this) must be put into the SSAP. The SSAP GUI allows the SSIT operators to add files to the SSAP and then insert it all at once. The SSAP is actually two different data types at the Data Server. The Algorithm Package is just metadata, information about the SSAP that must be filled in by the SSAP GUI. Then each SSAP component (source code, test data, etc.....) must be inserted into the Data Server separately. To “copy” the files into the SSAP, the user will have to know their location on the SSIT workstation

(and in ClearCase) so that they can be selected on the GUI. Once the SSAP is complete, it is inserted to the Data Server (many different inserts by the GUI itself). Note that the GUI also allows the user to modify an existing SSAP (retrieve it and then re-insert it) and delete an existing SSAP.

Note that the SSIT personnel may want to save the SSAP to both the “SSIT” and “OPS” Data Servers. To allow this, the SSAP GUI must be able to save SSAPs to the local disk and recreate them so they can be “copied” from one mode of the Data Server to the other.

The PGE executables are stored separately from the SSAP, as are any static files needed during PGE execution. There is a tar file that contains the exe and any Toolkit files that the executable needs. A script tool performs the Insert of this to the Data Server. The static files are also stored via a script tool.

The PDPS database (on the AIT hardware) that was populated during testing is then copied into the “operational” PDPS database on other hardware. This is done by re-running the tool that populates the PDPS database from the ODL files in the “OPS” mode. This means that the ODL files need to be saved (possibly *checked into ClearCase?*) so that the production database can be populated after SSIT.

Descriptions of the various IDs that SSIT creates:

- **PgeId**: This is the identifier of the PGE, used as a key into the PDPS database. SSIT takes the PGE name, and concatenates the PGE Version and Profile Ids. The maximum length is 20 characters, with the maximum name 10 characters, maximum version 10 characters and the profile Id 3 characters. ‘#’ is used to pad the end of the name or version if it is not equal to the maximum length.
- **SswId**: This is the identifier of the PGE executable, also used as a key (for different tables) in the PDPS database. SSIT takes the PGE name, and concatenates the SSW Version (version of the source code). The maximum length is 20 characters, with the maximum name 10 characters (as in PgeId), and the maximum SSW version 10 characters. ‘#’ can be used to pad the end of the name if it is not equal to the maximum length.
- **DataTypeId**: This is the identifier of Data Types, and is used as a key in the PDPS database. SSIT forms it in one of two ways: for normal Data Types, it takes the Data Type Name (ESDT Short Name) and concatenates it with the Data Type Version (ESDT Version). The maximum length is 20 characters, but since the maximum name is 10 characters and the maximum version 10 characters, a ‘#’ can be used to pad the end of the name.

The Science Server Archive Package (SSAP)

This is a grouping of science software, documentation, and other related files that is stored at the DAAC. These are accessed and updated through an SSIT SSAP GUI. It is supposed to be a record of the software, complete with source code, documentation, what and how it was tested, etc..... This is both for recording purposes and so that the tests can/could be repeated later.

Note that the executables and any static files needed by the PGE are stored separately from the SSAP.

The SSAP is not what is received from the SCF. This is currently being called the DAP (Delivery Algorithm Package) and it is similar to the SSAP in that it contains test data, source code, etc.... Much of what is in the DAP will make it into the SSAP (although it may be modified, i.e. code may have bug fixes).

The SSAP will be made up of two different Data Types at the Data Server. The Algorithm Package is metadata about the SSAP, such as the name of the PGE, name of the instrument, date accepted, etc.... Each part of the SSAP (source code, documentation, and test data) will be stored as an SSAP component, with its own metadata in addition to the files. SSAP components such as source code will be tarred to retain the directory structure (so they must be untarred when retrieved). The executables and static files are stored separately from the SSAP, and thus have their own Data Types (ESDTs).

What follows is a list of items in the SSAP. See also the Core Metadata model under DAP for a graphical representation of the SSAP.

The following make up an SSAP:

- Documentation
 - Delivery Memo.
 - Summary Information for each PGE.
 - System Description Document (SDD).
 - Operations Manual.
 - Processing Files Description Document.
 - Test Plans (these include the test cases).
 - Scientific documents.
 - Interface Definition Document.
 - Detailed Performance Testing Results
 - Detailed design/implementation documents.
 - COTS User or Programmer Guides.
- Software & Control files:
 - Science software source code (including make files & scripts).
 - Testing software source code (including make files & scripts).
 - Test Data Input (this may only be the UR for this).
 - Expected Test Output.

- Coefficient Files.
- Process Control File.
- Metadata Configuration File.
- ODL files. These define the PGE, and its related Data Types to the PDPS database. They currently don't have official names.
- Other files:
 - There is a log file that is created by the SSAP GUI to describe any changes made to the SSAP.

The current list of File List types in the SSAP code:

- Change Log.
- Context Diagram. This would be the context diagram(s) for the PGE.
- Delivery Contents.
- A copy of the delivery memo.
- Manual. Any manuals about the PGE, test code, etc.
- Performance Test Results. The results of any performance tests.
- Processing File Description.
- Programmers Guide.
- A pointer or a copy to the Programmers guide used to write the software.
- Test Plan. The plan for testing the software.
- Software Standard. The software standard used to write the software.
- System Description. Description of the system used to run the software.
- Compilation Information. Information on how to compile the software.
- Description. A description of the PGE/software. Note that this might not be an actual part of the SSAP. It might only exist in metadata.
- PGE Error Log. Log of PGE errors.
- PGE Executables. The executable(s) that make up the PGE. These are what are executed to run the PGE. Note that the executables that are used by processing are currently retrieved from somewhere else.
- Dependencies.
- Test Site Configuration. The configuration needed to test the software.

- Version. PGE version. Note that this might not be an actual part of the SSAP. It might only exist in metadata.
- External Data. Data needed when running the PGE.
- Engineering Data. More data needed when running the PGE.
- Science Data. Scientific data (temporary products) needed to test the PGE.
- Metadata Control File. Needed by the Toolkit to allow for updates to the metadata of a product by the PGE.
- Ancillary Data. More data needed to run the PGE.
- Control parameters and Resource files. More information on running the PGE. This might include runtime parameter values.
- Results Report. Report of tests run.
- Results Product files. Files created during testing. These also could be comparison files for use after a test is run (to compare current output with expected output).
- Results Temporary files. Temporary files created during testing.
- Test Scripts. Scripts needed when testing the software.
- Test Source. Source code for test drivers, stubs, etc.
- Compilation Scripts. Scripts needed to compile the software.
- Software Source. The source code for the software.
- Software Scripts. Scripts needed to develop or run the software.

Data Server had to write a special interface for SSAPs. AIT will use standard Data Server commands to access the SSAPs, INSERT (add), ACQUIRE (Retrieve), QUERY/INSPECT (give me a list of them). The Algorithm Package will be inserted as just metadata, while the rest of the SSAP will be insert with both metadata and component file(s). There is currently a requirement for creation of HTML pages to allow users to access and retrieve parts of an SSAP. These HTML page requirements may go away, since all of that can be performed by the SSAP GUI. The SSAP Editor is covered in more detail in 313-EMD-001, *Release 7.10 Internal Interface Control Document for the EMD Project*. Additional information on the SSAP Editor is also available in 609-EMD-001, *Release 7.10 Operations Tools Manual for the EMD Project*.

ODL File(s)/PGE Metadata

These are parameter = value files that describe the PGE, its inputs and outputs, along with other information needed in the PDPS database. The current plan is for these files to be kept in ClearCase with the PGE, and thus to be configuration managed along with the software. They could also be stored as part of the SSAP.

There are currently five types of ODL files:

- PGE Metadata (information about the PGE)
- ESDT Metadata (information about each input/output of the PGE),
- ORBIT Metadata (information about the orbit of the spacecraft supplying the data for the PGE),
- TILE Metadata (information about the geographic tiles that the PGE will produce) and
- PATHMAP_ODL (information about mapping between Absolute Path Number and a Sequential numbering ranging from 1-233).

There are “.template” files for each type of Metadata/ODL file that describe each field and give examples of how it’s used. A PGE Metadata ODL file must exist for each PGE Profile in the system, and an ESDT Metadata ODL file must exist for each input/output of those PGEs. ESDT Metadata ODL files can be shared by PGEs (that have the same input/output). The ORBIT Metadata ODL File is only required for PGEs that are scheduled by “Orbit” rather than time, and it will only exist for each type of Spacecraft for which there is data to be processed. The TILE Metadata ODL File will exist for each definition of Tiles, where a PGE wants to be scheduled to produce data for a certain geographical location. The PATHMAP_ODL is required if the PGE_ODL specifies a PATHMAP. The PATHMAP definition ODL files must reside in directory \$DPAT_RULE_SCIENCE_MD. Each file must be named PATHMAP_<Pathmap_Name>.odl.

The PGE Metadata ODL file can be created from a PGEs PCF by running the CreateODLTemplate tool or it can be created from scratch by copying the PGE_ODL.template file. In either case the file must be edited to add ESDT information, correct numbers of input and output files, type of scheduling for the PGE, and many other things. The ESDT Metadata ODL File must be created from either an existing ODL File or the ESDT_ODL.template file. The Tile and Orbit Metadata ODL Files are also created from the template files (TILE_ODL.template, ORBIT_ODL.template) are copied from an existing ODL file and edited. Once all ODL files for a PGE are created (PGE Metadata ODL File, any ESDT Metadata ODL Files needed to describe inputs and outputs, an ORBIT Metadata ODL File to describe spacecraft orbits and a TILE Metadata ODL File to describe geographic outputs), then the information about the PGE can be stored in the PDPS database so it can be scheduled and executed by Planning and Processing.

Science Software Integration and Test (SSIT) Manager

SSIT Manager Overview

The principal tool used during SSI&T is the SSIT Manager. The SSIT Manager is the top-level graphical user interface (GUI) environment presented to SSI&T personnel. Its purpose is to bring together the tools needed for SSI&T into a single, graphical environment. Please refer to 609-EMD-001, *Release 7.10 Operations Tools Manual for the EMD Project*, for the latest update on expanded uses of the SSIT Manager.

The SSIT Manager is the starting point for use by the SSI&T specialist to check in and verify the science software delivered by the instrument teams at the SCFs. The SSIT Manager application provides access to all COTS tools and custom applications that are part of the SSI&T environment. The SSIT Manager GUI is capable of kicking off instrument-specific compilation and execution scripts, configuration management scripts, custom code checking, file display and comparison tools, and COTS tools such as analysis environment programs. The SSIT Manager GUI contains a checklist of SSI&T steps in delivery and testing of science software, and a display of a log file recording SSI&T events. The checklist and log are the only inherent functionality to SSIT Manager; all other programs run from the Manager are also accessible from the Unix command line.

The terms Process Control File (PCF) and Object Description Language (ODL) are used in the following sections. The PCF is a file that tells an executable where to find its various inputs and outputs, as well as the values for any specific runtime parameters. Several of these files are used both by the SSIT Manager and by PGEs. ODL is a parameter = value format for input files. It is used to define the PGEs to the Planning and Data Processing Database.

Table 1 presents a summary of the capabilities provided via the SSIT Manager GUI.

Table 1. Common Operator Functions Performed through the SSIT Manager GUI (1 of 3)

Operating Function	Command/Script or GUI	Description	When and Why to Use
Prepare SSI&T checklist.	SSIT Manager GUI	<ul style="list-style-type: none"> • SSIT Manager GUI requires a checklist which it can access • the checklist contains all the operational procedures for science software integration and test 	<ul style="list-style-type: none"> • during normal SSIT operations, used to keep track of activities pending and completed • checklist of operational procedures must first be prepared using a text editor. • SSIT Manager must then be linked to the checklist before invoked
Change SSI&T Checklist.	SSIT Manager GUI	<ul style="list-style-type: none"> • SSIT Manager GUI requires to change the checklist • the checklist contains all the operational procedures for science software integration and test 	<ul style="list-style-type: none"> • Update tracking of activities pending and completed • checklist of operational procedures must first be prepared using a text editor. • SSIT Manager must then be linked to the checklist before invoked
open xterm session.	<ul style="list-style-type: none"> • open xterm window via Tools pull-down menu • also can be opened via Unix command <i>xterm</i> 	Standard Unix command line window.	As needed for ad hoc use.
Code analysis.	select SPARCworks via Tools → Code Analysis pull-down menu	Used for ad hoc analysis of science software.	used to debug problems (e.g., memory leaks)
check for standards compliance.	select the following via Tools → Standards Checkers pull-down menu: <ul style="list-style-type: none"> • FORCHECK • Prohibited Function Checker • Process Control File (PCF) Checker • Prolog extractor 	<ul style="list-style-type: none"> • check FORTRAN 77 science software • check if certain functions are used in the science software which conflict with the production environment • check the syntax of the data in the Process Control File • Extract prologs from science software 	<ul style="list-style-type: none"> • to ensure that science code conforms to EMD project standards • to ensure that the delivered PCF is of the proper syntax • To extract prologs from science software

Table 1. Common Operator Functions Performed through the SSIT Manager GUI (2 of 3)

Operating Function	Command/Script or GUI	Description	When and Why to Use
Product Examination.	select the following via Tools → Product Examination pull-down menu: <ul style="list-style-type: none"> • IDL • EOSView 	Opened via Tools pull-down menu.	<ul style="list-style-type: none"> • Ad-hoc graphical analysis • For viewing an arbitrary file (e.g., standard product) in HDF format
File Comparison.	select the following via Tools → Product Examination File Comparison <ul style="list-style-type: none"> • ASCII files • Binary files • HDF files (GUI) • HDF files (hdiff) 	Compares the outputs of the science software between the DAAC and SCF.	Ensures that the output that was generated at the SCF when running the science software is the same output that is generated at the DAAC.
Edit text.	select the following via Tools → Text Editors pull-down menu: <ul style="list-style-type: none"> • Emacs • Xedit 	Text editors.	Edit arbitrary text file.
PDPS Database	select the following via Tools → PDPS Database pull-down menu: <ul style="list-style-type: none"> • PCF ODL Template • Check ODL Files • SSIT Science Metadata Update • SSIT Operational Metadata Update GUI 	<ul style="list-style-type: none"> • creates an ODL file template from the science software PCF • Check the ODL file updates PGE and ESDT SCIENCE metadata in the PDPS /SSIT database • updates PGE OPERATIONAL metadata via GUI in the PDPS /SSIT database 	To initialize and update the Planning/Production (PDPS) databases: <ul style="list-style-type: none"> • SSIT version • Production version

Table 1. Common Operator Functions Performed through the SSIT Manager GUI (3 of 3)

Operating Function	Command/Script or GUI	Description	When and Why to Use
Data Server	select the following via Tools → Data Server pull-down menu: 1. Acquire DAP 2. Insert Static 3. Insert Test Dynamic 4. Insert EXE TAR 5. Edit SSAP 6. Get MCF	1. Acquires DAP. 2. Inserts static input file. 3. Inserts test dynamic input file. 4. Inserts tar file with files needed for Processing. 5. Edits Science Software Archive Package (SSAP) components. 6. Acquires Metadata Configuration Files from the Data Server.	1. After DAP notification received by email. 2. After ESDT is registered in Data Server, before test PGE run. 3. After ESDT is registered in Data Server, before test PGE run. 4. After PGE compilation, before test PGE run. 5. After PGE testing is complete, at time of promotion to Production, as needed to edit/review SSAP components. 6. As needed to retrieve MCF.

Quick Start Using SSIT Manager

The SSIT Manager provides a common interface to the SSI&T tools. A more detailed discussion of the tools accessed via this GUI is provided in subsequent sections.

The following assumptions are made with regard to the use of the SSIT Manager application.

- The operator is located at a workstation or server to which the SSIT Manager has been configured.
- The operator has proper authorization to access the PDPS/SSIT database and the Data Server.
- To access files in ClearCase, the operator has a ClearCase view already set.
- The operator's environment has been configured as documented in the pertinent sections of the Help menu, available from the main window of the SSIT Manager.
 - The Index submenu of the Help menu provides access, through the Netscape browser, to a number of topics that help the operator in the environment configuration.
 - The Help menu contains a list of topics that can be searched through.

Invoke SSIT Manager from the Command Line Interface

- 1 To start SSIT Manager at the Unix command line type **DpAtMgr ConfigFile** *config_filename* **ecs_mode** *MODE* & then press the **Enter** key.
 - *config_filename* is the name of the user's personal Process Framework configuration file for this program, as customized from **\$ECS_HOME/CUSTOM/cfg/**.
 - *MODE* is the mode of operation (e.g., OPS, TS1, or TS2).
-

Sun Platform

Table 2 lists the SSI&T command line interfaces for the Sun workstation.

Table 2. Command Line Interfaces (Sun) (1 of 2)

Command Line Interface	Description and Format	When and Why Used
EcDpAtMgr	Startup of SSIT Manager.	To do SSI&T, and record items accomplished in the log.
EcDpAtMgrLogDump	Used to dump/print a log file to the screen.	As needed.
Xterm	Open a Unix command line window.	As needed.
Sparcworks	Ad hoc code analysis	As needed.
Ghostview	Postscript file viewer	As needed.
Netscape	WWW browser Netscape	As needed.
Acroread	PDF file viewer Adobe Acrobat	As needed.
DpAtMgrForcheck	FORTRAN 77 code analysis	Determine whether FORTRAN 77 science software adheres to standards.
EcDpAtBadFuncGui	Prohibited function checker (GUI)	Determine whether science software adheres to standards.
EcDpAtBadFunc	Prohibited function checker (command line)	Determine whether science software adheres to standards.
EcDpAtCheckPCF	Process Control File checker (GUI)	Determine whether PCF is valid.
EcDpAtMgrPrologs	Prolog extractor	Extract science software code prologs.
/data/IDL/idl_4/bin/idl	IDL	As needed.
EOSView	EOSView	HDF file viewer.
EcDpAtMgrXdifff	ASCII file comparison	Compare 2 text files.
EcDpAtBinDiffGui	Binary file difference environment	Compare 2 binary files.
DpAtCheckHdfFile	HDF file comparison (GUI)	Compare 2 HDF files.

Table 2. Command Line Interfaces (Sun) (2 of 2)

Command Line Interface	Description and Format	When and Why Used
DpAtHdiff	HDF file comparison (command line)	Compare 2 HDF files.
xedit	Text editor	As needed.
emacs	Text editor	As needed.
EcDpAtCreateODLtemplate	Create PGE metadata ODL template file.	Before running EcDpAtDefinePGE.
EcDpAtVerifyODL	Verify PGE metadata ODL template file.	Before Running EcDpAtDefinePGE.
EcDpAtDefinePGE	Update PDPS/SSIT database with SCIENCE metadata.	Before executing PGE in SSIT environment.
DpAtOpDbGui	Update PDPS/SSIT database with OPERATIONAL metadata.	Before executing PGE in SSIT environment.
EcDpAtStageDAP	Acquires DAP from the Data Server.	After email subscription notification received.
DpAtInsertStatic	Inserts static input file into the Data Server.	Before executing PGE in SSIT or Production environment.
DpAtInsertTest	Inserts test dynamic input file into the Data Server.	Before executing PGE in SSIT environment.
DpAtInsertExeTar	Inserts tar file of executables, etc. needed to run PGE file into the Data Server.	Before executing PGE in SSIT or Production environment.
EcDpAtSSAPGui	Edit and inserts a single SSAP component into the Data Server.	After SSI&T is finished, before official promotion to Production.
netscape <html page name>	HTML pages for acquiring SSAP components from the Data Server, including test outputs.	During SSI&T, to get test outputs; After SSI&T is finished.
EcDpAtaCQUIREMCF	Get ESDT from the Data Server and insert MCF.	Before inserting MCF in the Data Server.

SGI Platform

It is intended that the SSI&T tools be most often run from the SSIT Manager. A small number of SSIT tools run only on the SGI platform. Because of security considerations, these tools cannot be run from the SSIT Manager on the Sun. They may only be run from the Unix command line on the SGI platform as indicated in Table 3.

Table 3. Command Line Interfaces (SGI)

Command Line Interface	Description and Format	When and Why Used
usr/sbin/cvproj	ProDev Workshop: Used for ad hoc analysis of science software.	Used to determine causes of problems (e.g., memory leaks).
DpAtRusage	Measures PGE performance.	Output of this tool is to be typed into the "Performance Statistics" section of the PROFILE screen of the PDPS/SSIT Database Update GUI.

Table 4 lists SGI platform tools associated with the SSIT process.

Table 4. SGI Tools Description

Categories/ Tools	Tool Description & Use	Further Information
ProDev Workshop	<ul style="list-style-type: none"> • ProDev Workshop is a COTS package developed by SGI • This tool is targeted within the production system for applications running on the SGI science processors • ProDev Workshop is a software development support tool which includes several tools that may have applicability to SSIT • Among these tools is the capability to perform static code analysis to aid in the detection of memory leaks 	<ul style="list-style-type: none"> • ProDev Workshop includes online documentation describing its features • Other ProDev Workshop documentation is delivered with the production system software. • ProDev Workshop is not available from the SSIT menu. This tool must be started from the Command Line Interface: see Table 3
PGE Performance	<ul style="list-style-type: none"> • DpAtRusage is a custom tool developed for the Data Processing Subsystem • It measures performance parameters such as CPU time used for a PGE linked to the SDP Toolkit, SCF version. 	A help message is printed if the tool is invoked without input parameters.

Across the top of the SSIT Manager GUI are the toolbar items **F**ile, **T**ools, and **R**un. Clicking on each of these invokes a pull-down menu.

Under the **F**ile pull-down menu, the only item is **E**xit. Clicking on this causes the SSIT Manager to terminate.

The **T**ools pull-down menu has most of the SSIT Manager's tools. The menu functions are:

The **R**un pull-down menu initially contains no menu items. Its purpose, however, is to allow a place for SSI&T personnel to place their own custom tools and scripts SSIT Manager GUI.

This GUI (Figure 2) is the starting point for SSI&T activities. It provides access to a collection of tools that will be useful for this purpose.

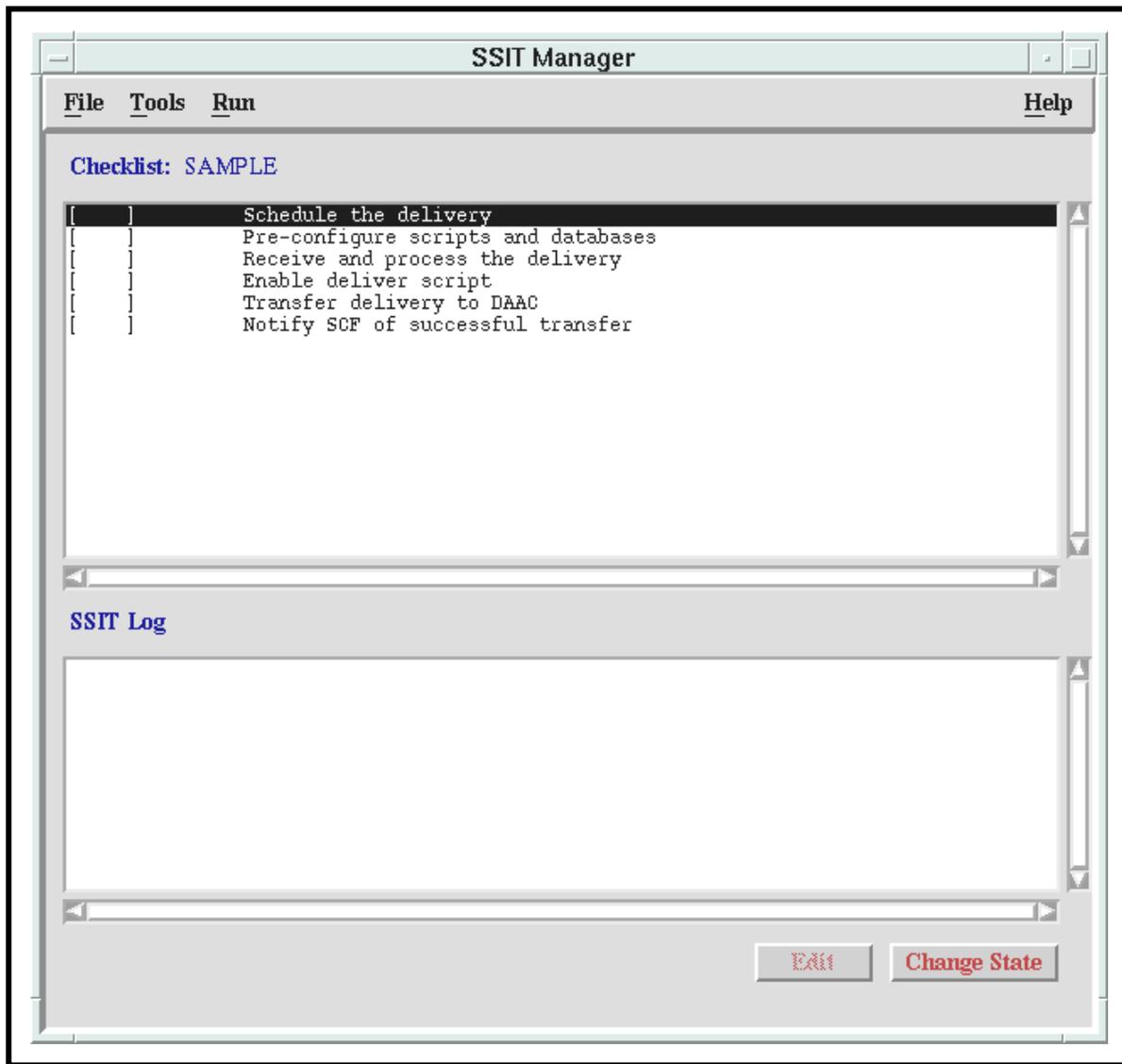


Figure 2. SSIT Manager Window

General Set Up of the SSIT Manager

The SSIT Manager requires a configured environment within which to run; it runs only on the AIT Suns. The set up steps described in this section need only be done the first time a SSI&T operator uses the SSIT Manager

Set Up the Environment for the SSIT Manager

- 1 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
 - 2 Type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - 3 Type **cd /usr/ecs/*MODE*/CUSTOM/utilities** then press the **Enter** key.
 - *MODE* is the mode in which you are operating. This mode should be **TS1** or another mode assigned beforehand to operate in.
 - For example, type **cd /usr/ecs/TS1/CUSTOM/utilities** and then press the **Enter** key.
 - This directory should contain scripts pertaining to setting the environment for SSIT Manager.
 - 4 Type **EcDpAtMgrStart *MODE*** then press the **Enter** key.
 - This invokes the **SSIT Manager GUI** that should be displayed.
-

SSIT Manager Tools

There are several tools that are accessible through the SSIT Manager GUI. After selecting the TOOLS menu option of the menu bar, a set of options is available. See Figure 3, which indicates the use of the Tool menu item.

Using the SSIT Manager

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The source file(s) are available, accessible, and have read permissions.
- The below listed formatted text (ASCII) files containing the list of prohibited functions exist in the directory stored in the environment variable DPATMGR_DAT:
 - prohibitedFunctions.Ada
 - prohibitedFunctions.C++
 - prohibitedFunctions.C

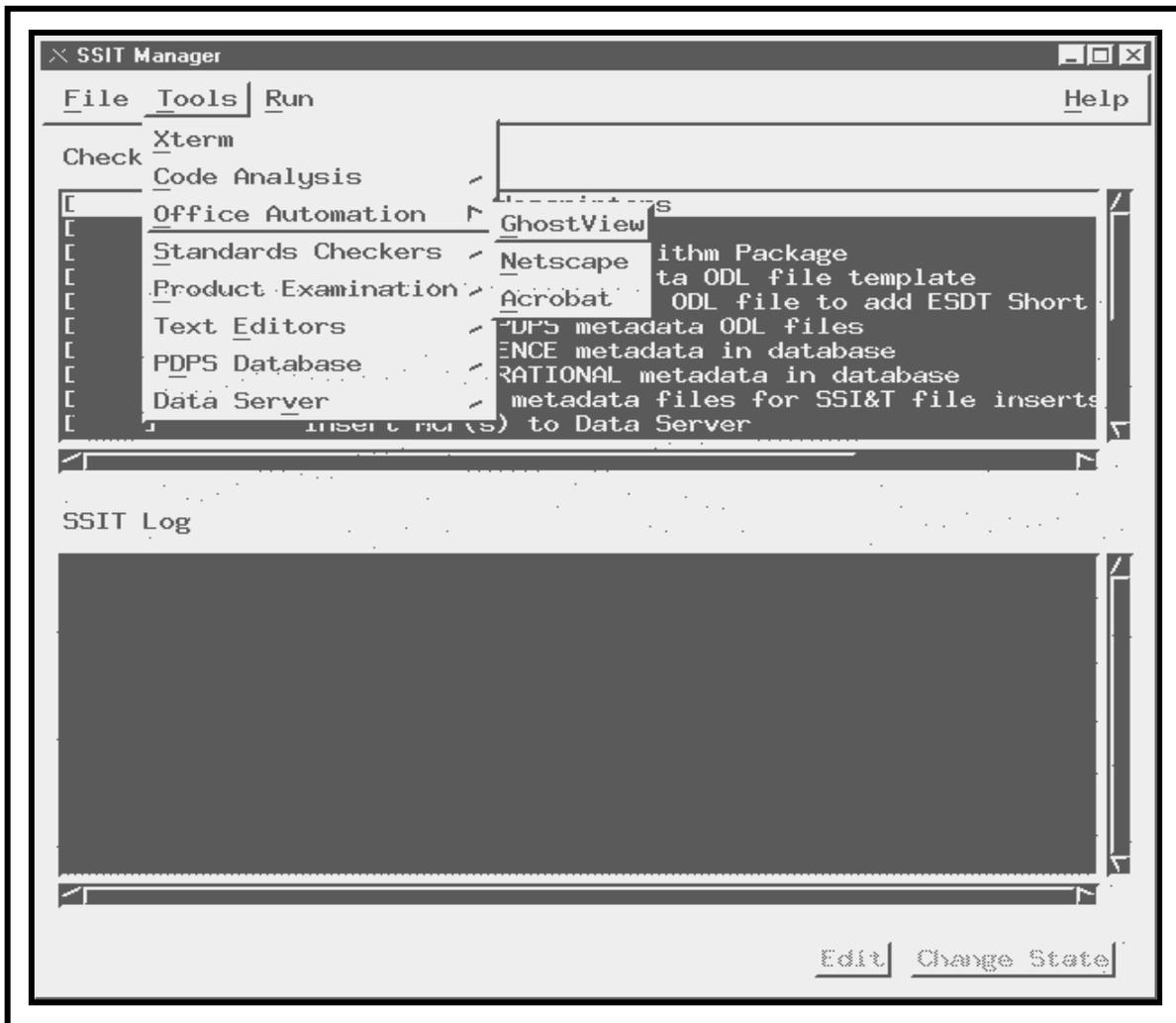


Figure 3. SSIT Manager Window - Tools Menu

- prohibitedFunctions.F77
- prohibitedFunctions.F90
- If the source code files to be checked are in a VOB in ClearCase, a view has been set before the SSIT Manager was started.

Acquiring and Unpacking the Delivered Algorithm Package (DAP)

The SSIT team receives the science software from the science community instrument teams. The following procedures will provide the SSIT team with the procedural steps that are used to acquire files.

Acquiring the Algorithm Package via FTP

Acquiring the science software via FTP is another method that the SSIT team will use in order to receive the science software. For this training class, a simple synthetic product generation executive (PGE) has been prepared for use by the students. This example will be used to demonstrate the FTP of the tar file from a remote machine. The students will then copy the tar file from the instructor's home directory.

Acquire the Algorithm Package via FTP

- 1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
- 2 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your **user id** and **password**.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
- 3 At a UNIX prompt, type **cd DeliveryPathname**, then press the **Enter** key.
 - The **DeliveryPathname** is the full path name to the directory that has been set aside for ftp pull of DAPs from the Instrument Team. For example, **cd /home/user** where **user** is the user's login directory, then press the **Enter** key.
 - If the DAP is to be copied into a subdirectory, change to this subdirectory.
- 4 At a UNIX prompt, type **ftp machineIPAddress**, then press the **Enter** key.
 - The **machineIPAddress** is the IP address or fully qualified domain name of the remote SCF machine. For example, **ftp 192.116.53.2**, then press the **Enter** key.
- 5 At the ftp prompt on the remote machine, enter **username**, then press the **Enter** key.
 - The remote machine will typically respond with **331 Password required for username**:

- 6 At the ftp prompt on the remote machine, enter user password, then press the **Enter** key.
 - The remote machine will typically respond with **230** User *username* logged in and display the ftp> prompt for further ftp commands.
 - 7 At the ftp prompt on the remote machine, type **cd *DAPpathname*** then press the **Enter** key.
 - The *DAPpathname* is the full path name to the directory on the remote machine containing the DAP to retrieve. For example, type **cd /home/mac** then press the **Enter** key. The directory location should be known.
 - 8 At the ftp prompt on the remote machine, type **binary**, then press the **Enter** key.
 - The binary command causes subsequent file transfers to be in binary mode, preserving the integrity of the file to retrieve without interpretation (as would be done in ASCII mode).
 - The system will typically respond with the message **200** Type set to I indicating that binary mode has been set.
 - 9 At the ftp prompt on the remote machine, type **get *DAPfilename***, then press the **Enter** key.
 - The *DAPfilename* is the file name of the DAP to retrieve.
 - For example, type **get TestPGE.tar** then press the **Enter** key.
 - The user may need to type **dir** then press the **Enter** key to display a listing of the files in the current directory. The system will likely display several lines of messages once the transfer has completed. For large files, this may take a long time (minutes to hours depending upon the size of the DAP and the bandwidth of the connection).
 - 10 At the ftp prompt on the remote machine, repeat Step 8 or type **bye** then press the **Enter** key.
 - Typing bye and pressing Enter closes the ftp connection with the remote machine.
 - Retrieve other DAP files by repeating Step 8. The DAPs retrieved will reside in *DeliveryPathname* on the local machine.
-

Unpacking a DAP

Once a DAP has been acquired via electronic means or physical media, it typically needs to be unpacked before its contents are accessible for SSI&T. Several mechanisms are available under standard UNIX for packing and unpacking files to and from a file archive, the most common being UNIX *tar*.

Unpack a DAP

- 1** On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
 - 2** Log into one of the AIT Sun workstations where the retrieved DAP resides by typing: username then press the **Enter** key.
 - 3** Enter the **password** then press the **Enter** key.
 - 4** At a UNIX prompt, type **cd UnpackPathname**, then press the **Enter** key.
 - The UnpackPathname is the path name of the directory that has been set aside for unpacking of DAPs.
 - This directory now contains the DAP tar file. For example, cd /home/user, where user is the user's login directory, then press the Enter key.
 - 5** If the tar file is compressed, at a UNIX prompt, type **uncompress PackedDAP.Z**, then press the **Enter** key.
 - The PackedDAP.Z is the file name of the compressed DAP file.
 - The file name extension of .Z is a convention indicating UNIX compressed files. The uncompress utility expects this file name extension by default. A resulting error may indicate that the DAP file was not compressed or that another compression utility was used. If the file name extension was .Z, the uncompressed version will have the same file name but without the .Z, for example PackedDAP.
 - The tar file for the SSI&T Training will not be compressed.
 - 6** At the UNIX prompt, type **tar xvf PackedDAP**, then press the **Enter** key.
 - The PackedDAP is the file name of the uncompressed DAP file.
 - The tar archive will be unpacked in the current directory. If the archive contained directories and subdirectories, these will be created by the tar utility and populated by the files that belong.
-

SSIT Software Operating Instructions

Start the SSIT Manager GUI

- 1 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
 - 2 Type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - 3 Type **cd /usr/ecs/*MODE*/CUSTOM/utilities** then press the **Enter** key.
 - This directory should contain scripts pertaining to setting the environment for SSIT Manager.
 - 4 Type **EcDpAtMgrStart *MODE* &** then press the **Enter** key.
 - This invokes the **SSIT Manager** GUI, which should be displayed.
-

What must be done via SSIT tools:

- Since SSIT is just a calibration of various tools, there is no specific order for which they must be run. Most tools can be brought up from the SSIT Manager GUI as well as started on their own.
- The **File** menu provides the capability to exit the manager. The **Tools** menu provides access to the various tools that make up SSIT. The **Run** menu is customizable (allowing you to add your own scripts and tools) by editing the file *ssit_run_menu* in the *data/DPS* directory.
- The checklist (first window on the GUI) allows you to check off various activities by double clicking on them. You may enter a commentary on the activity in the second window when checking off a particular item. The file *checklist.sample* in the *data/DPS* directory can be edited to change the items in the checklist or its' location.

Subscribing to the DAP

The following Servers/Services must be up and operational:

- Data Server.
- Subscription Server.

The following must have occurred between those Servers/Services:

- Data Server must have inserted the DAP ESDT.

Prepare for Trying SSIT Functionality

- 1 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
- 2 Type **setenv DISPLAY ipaddress:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
- 3 Perform a remote login by typing **ssh hostname** then press the **Enter** key.
 - *hostname* refers to the Subscription Server GUI host (e.g., x0dms01).
- 4 At the Password prompt type *password* then press the **Enter** key.
- 5 Type **cd /usr/ecs/MODE/CUSTOM/utilities** then press the **Enter** key.
- 6 Type **EcSbSubServerGUIstart MODE** then press the **Enter** key.
 - The Subscription Service GUI should appear.
- 7 Select **Add Subscription...**
 - Add/Edit Subscription GUI appears.
- 8 Use **Browse Events** to choose *Event ID*.
- 9 Place *Event ID* into *User ID* slot.
- 10 Type *address* in the **Email Address** slot provided.
 - The subscription notification will be returned via email.
- 11 Type *text* in the **Email Text Identification** slot.
 - Limited to one Word.
- 12 Verify that **Start Date Group** is the current date.
- 13 Verify that **Stop Date** is a date in the future (or if running this day, use current date).
- 14 Select **Action**.

- 15 Type the Data Type Id (**DAP + # + VersionID**) of the ESDT that you wish to search for (e.g., DAP#1).
 - Do not override the provider.
 - 16 Submit the subscription using the default service (i.e., Notification of the Insert of a granule).
 - A good status message should be displayed.
 - Email will be returned when a DAP is Ingested (by Ingest)
-

DAP Insert

The **DAP** (Delivered Algorithm Package) is the vehicle by which the PGE, source code, supporting files, documentation, etc. are delivered to a DAAC for SSI&T. Typically, the DAP is a compressed TAR file with a file name of form **string.tar.Z**. After initial processing, the DAP is broken apart into its components and those components will be subsequently processed and used based on their intended function.

The **insert** service is used to put the DAP into the Data Server. Once the DAP is in the Data Server, the **acquire** service is used to retrieve it.

Performing a DAP Insert

The method described here, for performing a DAP insert, is by command-line prompts and responses.

Assumptions:

- The DAP ESDT has been installed on the Data Server.
- The Target MCF for the DAP has been created for the Insert.
- The SSIT Manager is running.

Insert a DAP into the Science Data Server

- 1 From the **SSIT Manager** select **Tools** → **Data Server** → **Insert Test Dynamic** from the pull-down menu.
 - An xterm with title **SSIT: PGE Test Dynamic Input File Insertion** is displayed.
- 2 At the program prompt **Configuration filename? (enter for default: ../.cfg/EcDpAtInsertTestFile.CFG)** and then press the **Enter** key.
- 3 At the program prompt **ECS Mode of operations?** type **MODE** then press the **Enter** key.

- 4 At the program prompt **ESDT short name for the file(s) to insert?** type *ESDTShortName* then press the **Enter** key.
 - The *ESDTShortName* is the ShortName for the DAP file, which is DAP.
 - 5 At the program prompt **ESDT Version for the file(s) to insert?** type *version* then press the **Enter** key.
 - 6 At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?** press the **Enter** key to accept the default.
 - 7 At the program prompt **Single Filename to Insert? (including FULL path)** type *pathname/GranuleFilename* then press the **Enter** key.
 - *pathname/GranuleFileName* is the full path name and DAP file name.
 - 8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** type *pathname/DAP.tar.met* then press the **Enter** key.
 - *pathname* is full name of the path and *DAP.tar.met* is the name of the associated **.met file**.
 - 9 At the program prompt **Hit enter to run again, 'q <enter>' to quit:** type **q** and press **Enter** to quit or just press **Enter** to insert additional dynamic granules.
 - If continuing, repeat Steps 2 through 8.
-

This page intentionally left blank.

DAP Acquire

The **DAP** (Delivered Algorithm Package) is the vehicle by which the PGE, source code, supporting files, documentation, etc. are delivered to a DAAC for SSI&T. Typically, the DAP is a compressed TAR file with a file name of form *string.tar.Z*. After initial processing, the DAP is broken apart into its components and those components will be subsequently processed and used based on their intended function.

- The insert service is used to put the DAP into the Data Server. Once the DAP is in the Data Server, the acquire service is used to retrieve it.
- DAP is acquired from Data Server and placed in the specified directory. Note there will be two files, the DAP itself (a big tar file) and the metadata associated with the DAP. The metadata may be helpful in the creating the SSAP.

Performing a DAP Acquire Using SSIT Manager

Generally, the preferred approach to accomplishing a DAP **acquire** will be through the use of the SSIT Manager GUI.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The following servers/services are up and operational:
 - Data Server.
 - Subscription Server.
 - Storage Management.
- The following must have occurred between those Servers/Services:
 - Ingest must have ingested DAP and Inserted it into the Data Server.
 - Subscription Server must have gotten notification from the Data Server of the Insert.
 - Subscription Server must send email to the SSIT operator notifying him/her of DAP Insertion and giving him (in the email) the UR of the DAP.
- The SSIT Manager is available.
- The X Window **DISPLAY** environment variable is pointing to your screen

Perform a DAP Acquire

- 1 If not already on an AIT Sun, log onto one from your current machine.
 - 2 Bring up the SSIT Manager GUI at the UNIX prompt by typing **mgr**.
 - After a short while, the SSIT Manager GUI will appear.
 - 3 From the **SSIT Manager** select **Tools** → **Data Server** → **Acquire DAP** from the pull-down menu.
 - See Figure 4.
 - If the SSIT Manager GUI is used to initiate the DAP processing, Steps 4 and 5 can be skipped.
 - 4 Alternately, one can initiate the DPA processing sequence from the command line: type **source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc** then press the **Enter** key.
 - Note: This step only needs to be done once per login.
 - 5 Type **/usr/ecs/TS1/CUSTOM/bin/DPS/DpAtStageAlgorithmPackage.sh** then press the **Enter** key.
 - 6 At the prompt **** DAP Staging Tool ** Configuration filename? (enter for default: DpAtAA.CFG)** press the **Enter** key.
 - 7 At the prompt **ECS Mode of operations? (enter for default: OPS)** type **MODE** then press the **Enter** key.
 - 8 At the prompt **Name of email message file (including path)?** type *path/filename* (e.g., **/home/diascone/emessage01.asc**) then press the **Enter** key.
 - 9 At the prompt **Directory to receive staged file?** type *directory* (e.g., **/home/diascone/staged**) then press the **Enter** key.
-

Performing a DAP Acquire Using Ingest

This program is used to retrieve the Delivered Algorithm Package (DAP) from the Data Server.

Before its use, the following events must have occurred:

- A subscription for the DAP must have been registered with the Data Server. The subscription delivery option will be sent to email, to a specified DAAC operator or maildrop.

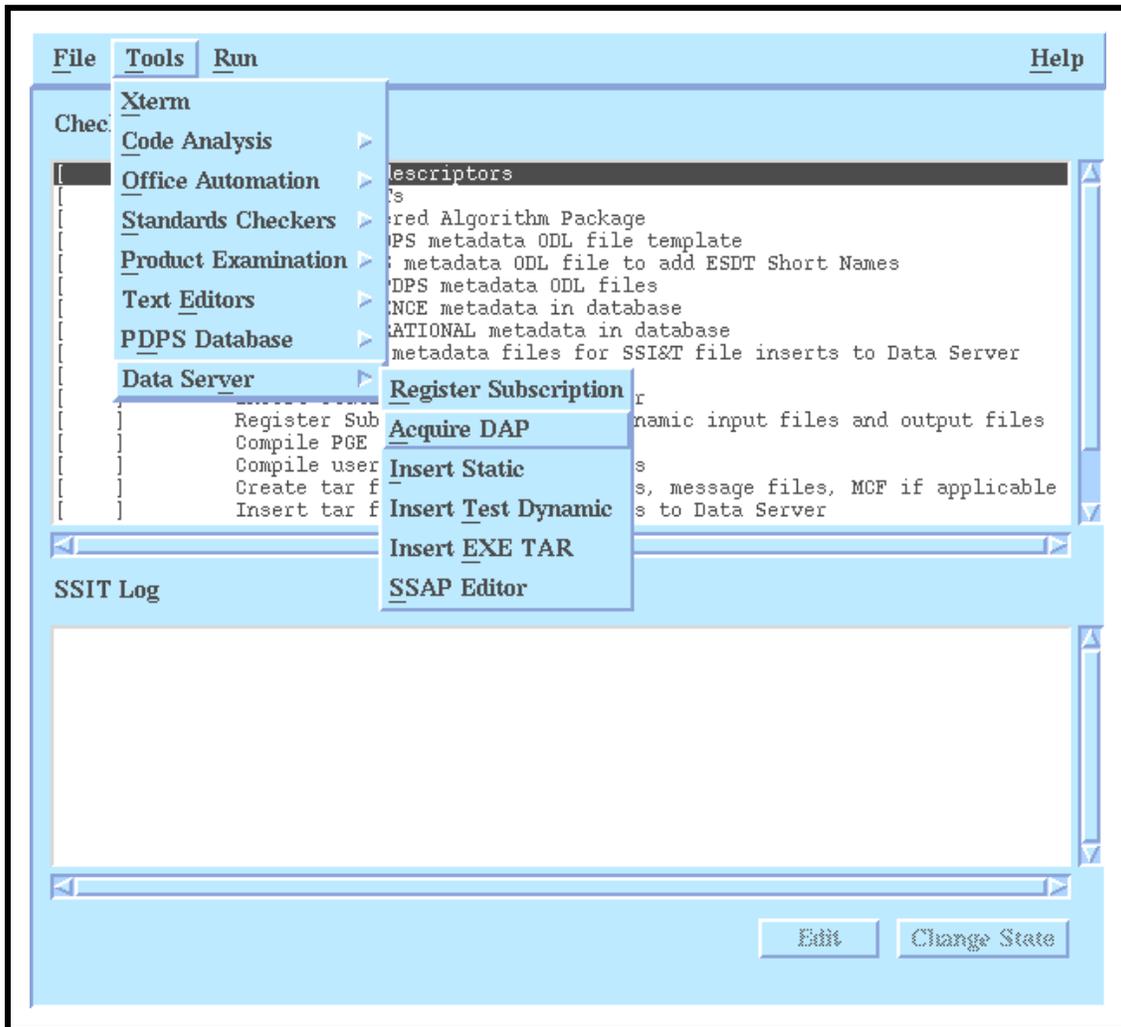


Figure 4. Selecting DAP Acquire from SSIT Manager

- The DAP must have been Ingested. There are two ways for this to occur. First, the DAP may be processed by Ingest. When this occurs, Ingest inserts the DAP into the Data Server, triggering subscription notification. Second, the DAP may be inserted with the Insert Test File.
- After the DAAC operator received the email subscription notification, he or she must have saved it to a file.

To run the program, select **Acquire DAP** from the **Data Server** submenu. The program prompts for input parameters Process Framework configuration filename, email message filename and directory to receive staged file.

The program reads the DAP UR from the e-mail file, acquires the DAP from the Data Server, and stages the DAP on the local disk. The name of the staged file is written to standard output.

Note that this function may be used to acquire any type of granule with a known UR.

After the file is staged, the operator may unpack it and install its components in ClearCase or in other places as appropriate.

Use of this program is optional, in the sense that the DAP may arrive at the DAAC through other means than Ingest, e.g., simple ftp.

Inserting a Science Software Archive Package (SSAP)

The Science Software Archive Package (SSAP) is a grouping of science software, documentation, and other related files that is stored at the DAAC. They are accessed and updated through an SSIT SSAP GUI. The SSAP is supposed to be a record of the software, complete with source code, documentation, what and how it was tested, etc., and is created both for recording purposes and so that the tests can/could be repeated later. Note that the executables and any static files needed by the PGE are stored separately from the SSAP.

The SSAP is similar, but not identical to the DAP (Delivered Algorithm Package), in that it contains test data, source code, etc. Much of what is in the DAP will make it into the SSAP (although it may be modified, i.e., code may have bug fixes). Basically, the DAP is what arrives at the SSIT doorstep, while the SSAP is a similarly packaged finished product of the SSIT process, and so contains some things that were not in the DAP (and vice versa).

The SSAP will be made up of 2 different Data Types at the Data Server. The Algorithm Package is metadata about the SSAP, such as the name of the PGE, name of the instrument, date accepted, etc. Each part of the SSAP (source code, documentation, test data) will be stored as an SSAP component, with its own metadata in addition to the files. SSAP components such as source code will be tarred to retain the directory structure (so they must be untarred when retrieved). The executables and static files are stored separately from the SSAP, and thus have their own Data Types (ESDTs).

The items in the SSAP are described in the section on **Science Software Integration & Test Flow** (previous section of this lesson). (See also the Core Metadata model under DAP for a graphical representation of the SSAP.)

Inserting an SSAP into PDPS

This procedure describes how to insert an SSAP into PDPS.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- All required servers are up running.
- SSI&T is up and running.
- The C shell (or a derivative) is the current command shell.
- FORCHECK is available on the AIT Suns only.

Create the SSAP

- 1 If not already on an AIT Sun, log into one from your machine.
- 2 Launch the SSIT Manager.
- 3 From the **SSIT Manager** select **Tools → Data Server → SSAP Editor** from the pull-down menu.
 - The GUI starts. Note that it will first query Data Server for a list of SSAPs that have previously been created. This list will appear in the window at the center (if any SSAPs already exist). Current SSAP field will be blank, and only Refresh and Create buttons will be active. All three tabs (Main, Files, and Metadata) will be active.
- 4 Click on **Create** to create a new SSAP.
 - The Create SSAP window appears. If no OK button is visible, resize the window so that the OK button is visible.
- 5 Enter the *name* of the SSAP in the first field.
- 6 Enter *SSAP version* in the second field. Note that version has a limit of 20 characters.
- 7 Click *OK* and the window disappears.
 - On the main GUI, the SSAP created (what was entered in the step above) will appear. Current SSAP is now set to that value. All buttons are now active.
- 8 Click on the **File List** tab to set up SSAP components.
 - The File List Tab displays files in the local directory to the left and files in the selected SSAP component to the right. At the bottom left are a directory listing and a method to move through the directory tree on the local machine. Delete and Reset buttons -- both active -- are to the right.
- 9 Click on the **File Type** button to select the SSAP component to manipulate.
- 10 Choose **one** of the menu items.
- 11 Select **a file (or files)** from the left window to add to the component.
- 12 Click the **Add** button to add the files. They will appear in the right window because they are now part of that SSAP Component.
- 13 Now select the **Metadata** tab to set the metadata for the new SSAP. The Metadata Tab displays the metadata for the new SSAP. Only the Name and Version will be filled in automatically. The rest of the fields will have default information. While the SSAP can be submitted with the default information, it is wise to fill in valid values. To change a value click the mouse in the field you wish to change and type in a new value. For dates click in the first box or use the up/down arrows to move the date up or down. When

finished entering a date, click the *OK* button. For text fields just hit the *Enter* key. The button marked “Edit Assoc Collections” on the bottom of the window must be hit and an Associated Collection entered for the SSAP.

- 14 Click the **Edit Assoc Collections** button.
 - The Edit Associated Collections window displays a list of associated collections and fields for the entry of new ShortNames and Versions (which make up an Associated Collection).
 - 15 Enter a **shortname** (of an ESDT that has been installed in the Data Server) — must be eight or fewer characters. Note that the Data Server will verify if the Shortname exists.
 - 16 Enter the **version** (of the installed ESDT).
 - 17 Click **OK** and the new entry to the collection should appear in the window.
 - 18 Click **Done** to close the window.
 - 19 Click on the **Metadata** tab.
 - 20 Click **Save** to save the updated metadata.
 - 21 Click **Main** tab to get back to the Main tab.
 - 22 Click **Submit** to send the new SSAP to Data Server. When finished, a message should pop up that says, “**SSAP Successfully inserted to the Data Server**”.
-

This page intentionally left blank.

Updating a Science Software Archive Package (SSAP)

The Science Software Archive Package (SSAP) is a grouping of science software, documentation, and other related files that is stored at the DAAC. For a discussion of the SSAP and its contents

Updating an SSAP

This procedure describes how to update an existing SSAP in PDPS.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- All required Servers are up running.
- An SSAP must already have been inserted into the Data Server.
- The C shell (or a derivative) is the current command shell.

FORCHECK is available only on the AIT Suns.

Update the SSAP

- 1 If not already logged into an AIT Sun, perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the Subscription Server GUI host (e.g., x0dms01).
- 2 At the **Password** prompt type ***password*** then press the **Enter** key.
- 3 Launch the SSIT Manager using the procedure to **Invoke SSIT Manager from the Command Line Interface** (previous section of this lesson).
- 4 From the **SSIT Manager** select **Tools → Data Server → SSAP Editor** from the pull-down menu.
 - The GUI starts. Note that it will first query Data Server for a list of SSAPs that have previously been created. This list will appear in the window at the center (if any SSAPs already exist--if not, one will, of course, have to be created before the remainder of this procedure can be performed). **Current SSAP** field will be blank, and only **Refresh** and **Create** buttons will be active. All three tabs (**Main**, **Files**, and **Metadata**) will be active.
 - There is currently missing functionality in the SSAP Editor, so before updating the new SSAP you must hit the **Refresh** button to refresh the data about the new SSAP.

- 5 Click on the **Metadata tab** to update the SSAP.
 - The Metadata Tab displays the metadata for the SSAP. All fields will be set to the values entered when the SSAP was created, and the **Algorithm Name** field will be grayed out (because it may not be updated). If you want to create a new SSAP from an existing one, go back to the **Main** tab and hit the **Create With** button.
 - 6 Click on the **Algorithm Version** field (currently called **Algorithm Description**) and enter a new version (different from what is in the field when the tab is clicked).
 - 7 Update any other fields that you wish to change.
 - 8 Before you leave the **Metadata** tab, click **Save** to save the updated metadata.
 - 9 Click on the **File List** tab to set up new SSAP components.
 - The **File List** tab displays files in the local directory to the left and files in the selected SSAP component to the right. On the bottom left are a directory listing and a method to move through the directory tree on the local machine. **Delete** and **Reset** buttons -- both active -- are to the right.
 - 10 Click on the **File Type** button to select the SSAP component to manipulate.
 - 11 Choose one of the menu items.
 - 12 Select a **file** (or files) from the left window to add to the component.
 - 13 Click the **Add Arrow** button to add the files. They will appear in the right window because they are now part of that SSAP Component.
 - 14 Click **Main** to get back to the **Main** tab.
 - 15 On the **Main** tab, click **Submit** to send the new SSAP to Data Server.
 - When finished, a message should pop up that says, “SSAP Successfully inserted to the Data Server”. The SSAP has been updated at the Data Server.
-

Standards Checking of Science Software

Standards Checking Overview

The purpose of standards checking is to verify that the source files of the science software are compliant with the ESDIS Data Production Software Computing Facility (SCF) Standards and Guidelines document.

Checking FORTRAN 77 ESDIS Standards Compliance

The ESDIS Data Production Software Computing Facility (SCF) Standards and Guidelines document requires all FORTRAN 77 code to be compliant with the ANSI FORTRAN 77. The COTS used for this task is FORCHECK.

The following is a list of tools and/or assumptions:

- The FORTRAN 77 science software source code is available, accessible, and has “read” permissions for the user.
- SSIT Manager is available for use.
- FORCHECK is available on the AIT Suns only.

Check for ESDIS Standards Compliance in FORTRAN 77 Code

- 1** If not already logged into an AIT Sun, perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the Subscription Server GUI host (e.g., x0dms01).
- 2** At the **Password** prompt type ***password*** then press the **Enter** key.
- 3** If required, at the UNIX prompt on the AIT Sun, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - The ***ViewName*** is the name of a view allowing the FORTRAN 77 source files to be accessible.
 - This step is only necessary if any of the FORTRAN 77 source files are in ClearCase (in the VOB under configuration management).
- 4** If your general environment setup doesn't include transparent access to the SSIT Manager GUI, then you need to set that up. One way to do it is as follows:
 - Set up an alias, manually or from shell script, to set up preliminary environment: at the UNIX prompt, type **alias do_buildrc "source /usr/ecs/*MODE*/CUSTOM/bin/DPS/.buildrc"** then press the **Enter** key.

- Set up an alias, manually or through shell script, to invoke the SSIT Manager: at the UNIX prompt, type **alias do_ssit_man**
“/usr/ecs/MODE/CUSTOM/bin/DPS/EcDpAtMgr ConfigFile
/usr/ecs/MODE/CUSTOM/cfg/EcDpAtMG.CFG ecs_mode TS1& “then press the **Enter** key.
- 5** To set up the preliminary environment type **do_buildrc** then press the **Enter** key.
- **do_buildrc** is one of the aliases set up in Step 4.
 - This only needs to be done once per session.
- 6** To run SSIT Manager type **do_ssit_man** then press the **Enter** key.
- **do_ssit_man** is one of the aliases set up in Step 4.
- 7** From the **SSIT Manager** select **Tools → Standards Checkers → FORCHECK** from the pull-down menu.
- See Figure 5 for a screen snapshot of this step.
 - A separate FORCHECK window opens.
- 8** At the prompt **Global Option(s) and List File?** type the appropriate response then press the **Enter** key.
- In order to understand what the proper response should be, the user is encouraged to find hardcopy documentation for FORCHECK or to use the UNIX **man** utility and type **man forchk** then press the **Enter** key.
- 9** At the prompt **Local Option(s) and File(s)?** type the appropriate response then press the **Enter** key.
- In order to understand what the proper response should be, the user is encouraged to find hardcopy documentation for FORCHECK or to use the UNIX **man** utility and type **man forchk** then press the **Enter** key.
 - The prompt is repeated until there is a blank line and carriage return.
- 10** At the UNIX prompt on the AIT Sun, type **vi FORCHECKoutput** then press the **Enter** key.
- **FORCHECKoutput** is the file name for the output file produced in Step 9.
 - The **FORCHECKoutput** file will contain any warnings, errors, and other messages from FORCHECK. A summary will be at the bottom of the file.
 - Any text editor (not just **vi**) may be used for this procedure step.

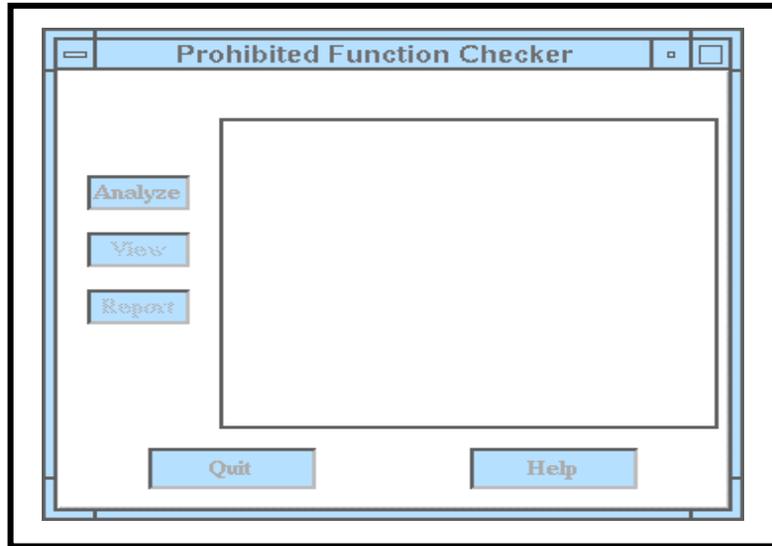


Figure 5. Prohibited Function Checker

- 11 At the UNIX prompt on the AIT Sun, type **vi *ListFile*** and then press the **Enter** key.
- *ListFile* is the file name for the list file specified at the FORCHECK prompt.
 - The *ListFile* file will contain FORCHECK messages similar to the *FORCHECKoutput* file embedded in the source code listing.
 - Any text editor (not just **vi**) may be used for this procedure step.
-

Checking for ESDIS Standards Compliance in Fortran 90

This procedure describes how to use the Fortran 90 compiler flags on the SPR SGI machines to check science software written in Fortran 90 for ESDIS standards compliance.

Unlike with FORTRAN 77, no COTS tool is used to check Fortran 90 science software. Instead, this procedure describes how to use the compiler to perform the checking (ESDIS standards for Fortran 90 are ANSI). Since the Fortran 90 compiler is used, the checking for standards compliance can be naturally tied in with building the science software (since this procedure will produce object files suitable for linking). However, in this procedure, the building of the software (compiling *and* linking) is deferred to a later procedure.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The Fortran 90 science software source code is available, accessible, and has “read” permissions for the user.
- Required Status Message Facility (SMF) files have been compiled.
- The C shell (or a derivative) is the current command shell.
- The Fortran 90 compiler is available on the SPR SGI.

Check for ESDIS Standards Compliance in Fortran 90 Code

- 1** From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 2** Perform a remote login by typing **ssh hostname** then press the **Enter** key.
 - **hostname** refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
- 3** To set up the proper environment for the compiler to be used type **source ToolkitPathname /bin/sgiXX/pgs-dev-env.csh** then press the **Enter** key.
 - **ToolkitPathname** is the home directory of the desired SDP Toolkit version.
 - The directory **sgiXX** should be replaced with **sgi32** or **sgi64** as appropriate for the specific compiler desired.
- 4** On workstation **x0spg##**, at the UNIX prompt in a terminal window, type **source /data3/ecs/MODE/CUSTOM/toolkit/bin/sgi64/pgs-dev-env.csh** then press the **Enter** key. This will set up the various environment parameters, such as **PGSHOME**, to enable the 64-bit version of the FORTRAN 90 compiler to be run.
 - The **x** in the workstation name will be a letter designating your site: **g** = GSFC, **m** = SMC, **l** = LaRC, **e** = EDC (LP DAAC), **n** = NSIDC. The **##** will be an identifying two-digit number (e.g., **e0ais02** indicates an AIT Workstation at the LP DAAC).
- 5** Type **source /data3/ecs/MODE/CUSTOM/toolkit/bin/sgi64/pgs-dev-env.csh** then press the **Enter** key.
 - This will set up the various environment parameters, such as **PGSHOME**, to enable the 64-bit version of the FORTRAN 90 compiler to be run.
- 6** If required, at the UNIX prompt on the SPR SGI, type **cleartool setview ViewName** and then press the **Enter** key.
 - The **ViewName** is the name of a view allowing the Fortran 90 source files to be accessible.

- This step is only necessary if any of the Fortran 90 source files are in ClearCase (in the VOB under configuration management).
- 7 At the UNIX prompt on the SPR SGI, type **cd *SrcPathname*** and then press the **Enter** key.
- The ***SrcPathname*** is the full path name to the location of the Fortran 90 source files to be checked.
 - The ***SrcPathname*** will be in the ClearCase VOB is the Fortran 90 source files are checked into ClearCase.
- 8 At the UNIX prompt on the SPR SGI, type **f90 -c -ansi [-I\$PGSINC] [-I\$HDFINC] [/-IOtherIncFiles]...] *SourceFiles* >& *ReportFile*** and then press the **Enter** key.
- The terms in square brackets (*I I*) are used to optionally specify locations of include and module (.mod) files. The **\$PGSINC** already contains the SDP Toolkit include directory and **\$HDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include or module directories.
 - The ***SourceFiles*** is a list (space delimited) of Fortran 90 source files or a wildcard template (*e.g.* *.f90).
 - The **>&** is a C shell construct that causes standard error (where the output from the Fortran 90 compiler normally emerges) to be redirected to a file.
 - The ***ReportFile*** is the file name under which to save the results of the compile process.
 - The **-c** flag causes only compilation (no linking).
 - The **-ansi flag** enables ANSI checking.
 - Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
 - Do not use the **-I option** for include or module files that are in the standard directories or in the current directory.
 - The makefile for the science software may contain the names of additional include files needed by the software.
 - For example, type **f90 -c -ansi-I\$PGSINC -I\$HDFINC -I/ecs/modis/pge5/include/*.f90 >& pge10.report** and then press the **Enter** key.
- 9 At the UNIX prompt on the SPR SGI, type **vi *ReportFile*** and then press the **Enter** key.
- The ***ReportFile*** is the file name for the compilation results as produced in Step 8.
 - Any text editor may be used for this procedure step.
-

Checking for ESDIS Standards Compliance in C or C++

This procedure describes how to use the C compiler flags on the SPR SGI machines to check science software written in C for ESDIS standards compliance.

Unlike with FORTRAN 77, no COTS tool is used to check C science software. Instead, this procedure describes how to use the compiler to perform the checking (ESDIS standards for C are essentially ANSI). Since the C compiler is used, the checking for standards compliance can be naturally tied in with building the science software (since this procedure will produce object files suitable for linking). However, in this procedure, the building of the software (compiling *and* linking) is deferred to a later procedure.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The C science software source code is available, accessible, and has read permissions for the user.
- Required Status Message Facility (SMF) files have been compiled.
- The C shell (or a derivative) is the current command shell.
- The C compiler is available on the SPR SGI.

Check for ESDIS Standards Compliance in C Code

- 1 From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 2 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
- 3 At the UNIX prompt on the SPR SGI, type **setenv PGSHOME *ToolkitPathname*** then press the **Enter** key.
- 4 Type **source \$PGSHOME/bin/*sgiX*/pgs-dev-env.csh** then press the **Enter** key.
 - The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version.
 - ***sgiX*** refers to the appropriate processor. For example, type **source \$PGSHOME/bin/*sgi*/pgs-dev-env.csh** and then press the **Enter** key.
- 5 If required, at the UNIX prompt on the **SPR SGI**, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - ***ViewName*** is the name of a view allowing the C source files to be accessible.

- This step is only necessary if any of the C source files are in ClearCase (in the VOB under configuration management).
- 6 At the UNIX prompt on the SPR SGI, type **cd *SrcPathname*** and then press the **Enter** key.
- ***SrcPathname*** is the full path name to the location of the C source files to be checked.
 - ***SrcPathname*** will be in the ClearCase VOB is the C source files are checked into ClearCase.
- 7 At the UNIX prompt on the SPR SGI, type **cc -c -ansiposix [-ISPGSINC] [-ISHDFINC] [-IOtherIncFiles]...] *SourceFiles* >& *ReportFile*** and then press the **Enter** key.
- The terms in square brackets (*I*) are used to optionally specify locations of include and module (.mod) files. The **SPGSINC** already contains the SDP Toolkit include directory and **SHDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include directories.
 - ***SourceFiles*** is a list (space delimited) of C source files or a wildcard template (*e.g.* *.c or c++).
 - The **>&** is a C shell construct that causes standard error (where the output from the C compiler normally emerges) to be redirected to a file.
 - ***ReportFile*** is the file name under which to save the results of the compile process.
 - The **-c** flag causes only compilation (no linking).
 - The **-ansiposix** flag enables ANSI and POSIX checking.
 - Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
 - Do not use the **-I option** for include files that are in the standard directories (*e.g.* /usr/include) or in the current directory.
 - The makefile for the science software may contain the names of additional include files needed by the software.
 - For example, type **cc -c -ISPGSINC -ISHDFINC -Iecs/modis/pge5/include/ *.c >& pge10.report** and then press the **Enter** key.
- 8 At the UNIX prompt on the SPR SGI, type **vi *ReportFile*** and then press the **Enter** key.
- The ***ReportFile*** is the file name for the compilation results as produced in Step 7.
 - Any text editor may be used for this procedure step.
-

Prohibited Function Checker

The use of certain functions in the PGE is prohibited. The **Prohibited Function Checker** (Figure 5) is used to check C, FORTRAN 77 and FORTRAN 90, language source files for the occurrence of functions that are prohibited in the DAAC production environment.

Checking for Prohibited Functions: Command-Line Version

This procedure describes using the command-line version of the Prohibited Function Checker to check science software for the prohibited functions.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The source files to be checked are available, accessible, and have read permissions for the operator.
- Source files to be checked are C, FORTRAN 77, Fortran 90, C shell, Korn shell, Bourne shell, or Perl and have recognizable file name extensions.

Check for Prohibited Functions in Delivered Source Files

- 1 If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - The *ViewName* is the name of a view allowing the source files to be accessible.
 - This step is only necessary if any of the source files are in ClearCase (in the VOB under configuration management).
- 2 At the UNIX prompt on the AIT Sun, type **cd *SrcPathname*** and then press the **Enter** key.
 - The *SrcPathname* is the full path name to the location of the source files to be checked.
 - The *SrcPathname* will be in the ClearCase VOB if the source files are checked into ClearCase.
 - The *SrcPathname* can contain other directories that contain source files and/or more directories. The Prohibited Function Checker will search out all source files in subdirectories recursively.

- 3 At the UNIX prompt on the AIT Sun, type
/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrBadFunc ConfigFile
/data3/ecs/TS1/CUSTOM/cfg/EcDpAtBA.CFG *FilesOrDirectories* > *ResultsFile* and then press the **Enter** key.
 - The *FilesOrDirectories* is a list of source file names or directory names of directories containing source files.
 - *ResultsFile* is the file name for the results that are output.
 - For example, type **/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrBadFunc ConfigFile /data3/ecs/TS1/CUSTOM/cfg/EcDpAtBA.CFG main.c utils/ >** myOutput and then press the **Enter** key. Here, **main.c** is a source file and **utils/** is a directory that contains other source files.
 - 4 At the UNIX prompt on the AIT Sun, type **vi ResultsFile** and then press the **Enter** key.
 - *ResultsFile* is the file name for the output results as produced in Step 3.
 - Any text editor may be used for this procedure step.
-

Check for Prohibited Functions Using the Prohibited Function Checker GUI

- 1 From the **SSIT Manager** select **Tools** → **Standards Checkers** → **Prohibited Function Checker** from the pull-down menu.
 - The Prohibited Function Checker GUI will be displayed.
- 2 In the Prohibited Function Checker GUI, click on the **Analyze** button.
 - The File Selector GUI will be displayed.
- 3 Within the **Directories** subwindow, double-click on the desired directory.
 - Repeat this step until the directory with the source files to be checked are displayed in the **Files** subwindow.
- 4 Within the **Files** subwindow, click on the source files to be checked. Each file clicked on will be highlighted.
 - To choose groups of contiguous files, hold down the left mouse button and drag the mouse.
 - To choose non-contiguous files, hold down the **Control** key while clicking on file names.
- 5 In the File Selector GUI, click on the **OK** button.
 - The File Selector GUI will disappear.

- The files selected in Step 4 will be displayed in the Prohibited Function Checker GUI window as they are being checked.
- 6 In the Prohibited Function Checker GUI, click on the **Report** button.
- The Report GUI will be displayed.
 - For each file, a list of prohibited functions found will be displayed.
- 7 Optionally, click on the **Print** button or the **Save** button.
- Choose **Save** to save the results to a file; choose **Print** to have the results printed on the default printer.
 - Choosing **Save** will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.
- 8 Optionally, in the Prohibited Function Checker GUI, highlight one of the source files listed. Then click on **View**.
- The Source Code GUI will be displayed.
 - Occurrences of prohibited functions found in that source file will be highlighted.
 - Click on the **Next** button to bring into the window successive occurrences of prohibited functions (the **Next** button does not bring in the next source file).
 - Click on the **Done** button to close the Source Code GUI. Other source files may be examined similarly, one at a time.
- 9 In the Prohibited Function Checker GUI, click on the **Quit** button.
- The Prohibited Function Checker GUI will disappear.
 - This ends the session.
-

Checking Process Control Files

The next task to accomplish is to check that the PCFs are syntactically correct and contains all the necessary information for PGEs to run within the DAAC production environment. Only one PCF can be associated with a PGE. The following procedure describes how to check PCFs for valid syntax and format.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The Process Control File(s) are available, accessible, and have read permissions.
- If the source code files to be checked are in a VOB in ClearCase, a view has been set before the SSIT Manager was started.

Check Process Control Files Using the Process Control File Checker GUI

- 1** From the **SSIT Manager** select **Tools** → **Standards Checkers** → **Process Control File Checker** from the pull-down menu.
 - The Process Control File Checker GUI (Figure 6) is displayed.
 - 2** In the **Directories** subwindow, double click on the desired directory.
 - Repeat this step until the directory with the PCF(s) to be checked is displayed in the Files window.
 - Use the Filter subwindow to limit which files are displayed.
 - 3** Within the **Files** subwindow click on the PCF to be checked.
 - The file clicked on will be highlighted.
 - Only one PCF can be checked at a time.
 - 4** Click on the **Check PCF** button.
 - A GUI labeled PCF Checker Results will be displayed.
 - Results will be displayed in this window.
 - 5** Optionally, click on the **Save** button or on the **Print** button.
 - Choose Save to save the results to a file; choose Print to have the results printed on the default printer.
 - Choosing Save will bring up a GUI labeled Save To File. Specify the directory and file name in which to save the results file.
 - Choosing Print and then clicking on the **OK button** will send the results to the default printer.
 - 6** Click on the **Check Another** button or on the **Quit** button.
 - Choosing Check Another allows another PCF to be checked. Repeat Steps 2 through 5.
 - Choosing Quit causes the Process Control File Checker GUI to disappear and ends the session.
-

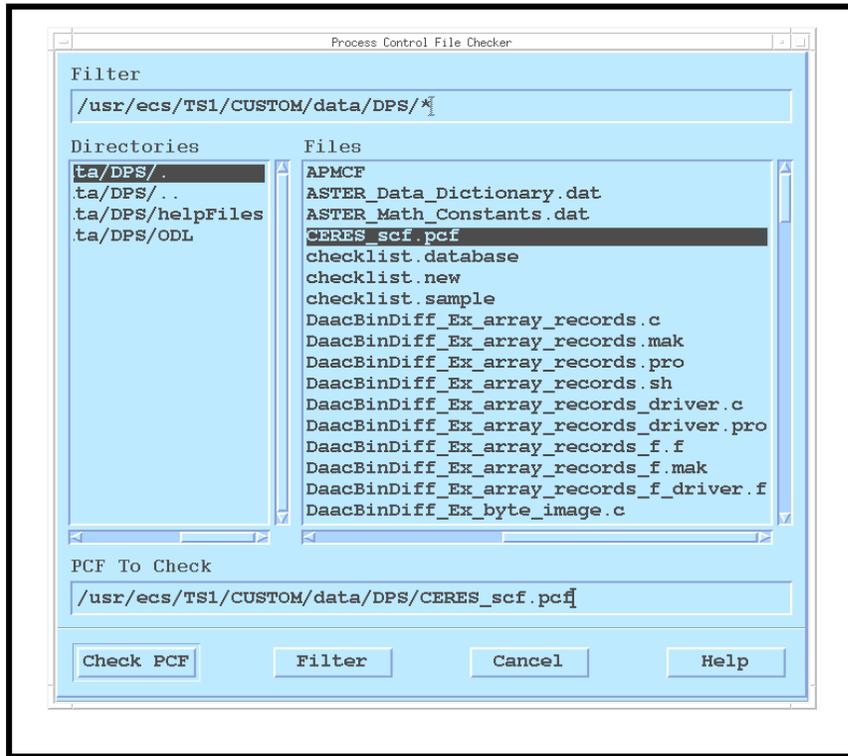


Figure 6. Process Control File Checker GUI

Checking Process Control Files (Command-Line Version)

This procedure describes using the command-line version of the Process Control File Checker to check process control files delivered with the science software.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The PCF files to be checked are available, accessible, and have “read” permissions for the operator.
- You will need the command **pccheck.sh**. One way to see if this is available is to type **which pccheck.sh** and then press the **Enter** key. If a path is displayed, then the directory is in your path. On the PVC Sun platform **p0ais01**, the pathname for the command is **/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh**. In this case, you will have to set a ClearCase view to access that area.

Check Process Control Files Using the Command Line Tool

- 1 If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName*** and then press the Enter key.
 - ***ViewName*** is the name of a view allowing the Process Control File(s) to be accessible.
 - This step is only necessary if any of the Process Control Files are in ClearCase (in the VOB under configuration management).
 - 2 At the UNIX prompt on AIT Sun, type **cd *PCFpathname*** and then press the **Enter** key.
 - ***PCFpathname*** is the full path name to the location of the Process Control File(s) to be checked.
 - The ***PCFpathname*** will be in the ClearCase VOB if the Process Control Files are checked into ClearCase.
 - 3 At the UNIX prompt on an AIT Sun, type **/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh -i *PCFfilename* > *ResultsFile*** and then press the **Enter** key.
 - ***PCFfilename*** is the full path name (directory and file name) to the Process Control File to check.
 - The ***ResultsFile*** is the file name for the results that are output.
 - The PCF Checker is also available on the SPR SGI machines. The easiest way to access it is to set a SDP Toolkit environment (any will do for purposes here) and type **\$PGSBIN/pccheck.sh -i *PCFfilename* > *ResultsFile*** and then press the **Enter** key.
 - 4 At the UNIX prompt on the SPR SGI, **p0spg01**, type **vi *ResultsFile*** and then press the **Enter** key.
 - ***ResultsFile*** is the file name for the output results as produced in Step 4.
 - Any text editor may be used for this procedure step.
-

Extracting Prologs

The Project standards and guidelines are contained in the document 423-16-01, *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines*. This ESDIS document mandates that science software delivered to the DAACs to be integrated into the production system contain prologs in the source files. Prologs are internal documentation containing information about the software. The details are specified in the ESDIS document. Prologs must be at the top of every function, subroutine, procedure, or program module.

This procedure describes using the Prolog Extractor to extract prologs into a file. Note that the prolog extractor only extracts the prologs it finds. It does not check the contents of prologs.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- Prologs are assumed to be delimited by particular delimiters depending on the language type. Delimiters are listed in Table 5.

Table 5. Prolog Delimiters

Language	Type	Delimiter
FORTRAN 77	source	!F77
Fortran 90	source	!F90
C	source	!C
FORTRAN 77	include	!F77-INC
Fortran 90	include	!F90-INC
C	include	!C-INC
Any Language	any	!PROLOG
All Languages	The end delimiter is always !END	

- The Prolog Extractor recognizes the language type of the file by its file name extension. Table 6 lists assumed file name extensions:

Table 6. Filename Extensions

Language	File Name Extensions
C	.c, .h
C++	.cpp, .h
Fortran 77	.f, .f77, .ftn
Fortran 90	.f90
C Shell	.csh
Korn Shell	.ksh
Bourne Shell	.sh
Perl	.pl

Extract Prologs

- 1 At the UNIX prompt on the AIT Sun type `/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrPrologs` then press the **Enter** key
-or-
From the **SSIT Manager** select **Tools** → **Standards Checkers** → **Prolog Extractor** from the pull-down menu.
 - An xterm is displayed on the AIT Sun.
 - Select the default ConfigFile. The output goes to a file called Prologs in the directory from which the SSIT Manager was started.
 - The Prologs file can be viewed by changing directories to the SSIT Manager directory and invoking a text editor. The file may also be sent to a printer.
 - 2 At the **Files(S)? (-h help)** prompt, type in the file names and/or directory names containing the files.
 - Separate items with spaces.
 - The contents of the directory will be search recursively for files with valid file name extensions.
 - Use `./` to indicate current directory.
 - The time needed for the Prolog Extractor could be very long for large numbers of files and directories.
 - When extraction is complete, the message **Output written to file: ./prologs** will be displayed. (Refer to Figure 7 for an example.)
 - 3 At the program prompt **Hit Enter for another, 'q <Enter>' to quit:** press the **Enter** key to repeat the process with another set of source files or type **q** and press **Enter** to quit.
 - The xterm will disappear.
 - 4 At a UNIX prompt on the AIT Sun, type **vi prologs**, then press the **Enter** key.
 - The extracted prologs file, named **prologs**, will be brought into the editor.
 - The default location of the prologs file is the directory from which the SSIT Manager was invoked.
 - 5 Once the extracted prologs file has been examined, exit the editor.
-

```
SOURCE CODE PROLOG EXTRACTOR
Configuration filename? (enter for default: ../../cfg/EcDpAtPrologs.CFG)
ECS mode? (enter for default: OPS)
TS1
File(s)? (enter -h for help)
/home/dps/ssit/*.c
Warning: Could not open message catalog "oodce.cat"
[Warning:
Invalid Resource Catalog directory path or no catalog installed
Applications can run with or without Resource Catalog
FYI : Values of ECS_HOME env variable and RC Directory path:/usr/ecs/ecsmode/CU
STOM/data/DPS/ResourceCatalogs
]
EcDpAtPrologs: Process Framework: ConfigFile ../../cfg/EcDpAtPrologs.CFG  ecs_m
ode ecsmode
Output written to file: /usr/ecs//TS1/CUSTOM/logs/prologs.txt
Hit return for another, 'q <return>' to quit:
□
```

Figure 7. Prolog Extractor Sample Run

Compiling and Linking Science Software

Science software is developed at independent Science Computing Facilities (SCFs) using the SDP Toolkit. The SDP Toolkit allows science software to be developed at independent SCFs for use in DAAC production systems. Once delivered to the DAACs for SSI&T, science software needs to be compiled and linked to one of the SDP Toolkit versions resident at the DAAC. The (PCFs) Process Control Files provide the interface between the science software and the production system. Since the process control files delivered to the DAACs for SSI&T were created and used at the SCFs, the path names in the PCF will need to be checked and revised to work at the DAACs.

To save time for the SSI&T Training Lesson, the compile and link with the SCF Version of the Toolkit will be omitted. The procedures are included in the student guide for future reference.

The next step is to set up a DAAC version SDP Toolkit environment, compile the PGE, and link to the DAAC Toolkit. This procedure will be performed at the SSI&T Training. The procedure steps for the two processes are the same except for the set up for the Toolkit environment and link with the corresponding Toolkit library.

Updating the Process Control Files (PCFs)

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- A PCF for the PGE has been delivered and is available, accessible, and has read permissions.

Update the PCF

- 1 From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 2 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
- 3 If required, at the UNIX prompt on the SGI, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - The ***ViewName*** is the name of a view allowing the PCF to be accessible.
 - This step is only necessary if the PCF is in ClearCase (in the VOB under configuration management).

- 4 At the UNIX prompt on the Sun or on the SGI, type **cd *PCFpathname*** and then press the **Enter** key.
 - The *PCFpathname* is the full path name to the location of the PCF. This location will be in the ClearCase VOB if the PCF is under configuration management.
- 5 At the UNIX prompt on the Sun or on the SGI, type **cleartool checkout -nc *PCFfilename*** and then press the **Enter** key.
 - The *PCFfilename* is the file name of the **PCF** that is to be checked out (and later modified). The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- 6 Run the Process Control File Checker on the delivered **PCF**.
 - This will verify that the delivered **PCF** is correct before editing.
- 7 At a UNIX prompt on the Sun, type **vi *PCFfilename*** and then press the **Enter** key.
 - The *PCFfilename* is the file name of the PCF to update.
 - Any text editor may be used such as *emacs*. For example, **emacs AST02.pcf** and then press the **Enter** key.
- 8 In the file, make changes to the default directories specified in each section of the **PCF**. All path names specified in the **PCF** must exist on the **SGI**.
 - Each section begins with a line consisting of a ? in the first column followed by a label:
 - ? **PRODUCT INPUT FILES**
 - ? **PRODUCT OUTPUT FILES**
 - ? **SUPPORT INPUT FILES**
 - ? **SUPPORT OUTPUT FILES**
 - ? **INTERMEDIATE INPUT**
 - ? **INTERMEDIATE OUTPUT**
 - ? **TEMPORARY I/O**
 - Each of the above section heading lines will then be followed (not necessarily immediately; there may be comment lines) by a line that begins with a ! in the first column. These lines specify the default path names for each section.
 - If the line reads:
 - ! **~/runtime**
 leave it unchanged. The **tilde (~)** is a symbol that represents **\$PGSHOME**.

- If another path name is listed instead, it will probably need to be changed to a path name that exists at the DAAC on the SGI. When specifying a path name, use an absolute path name, not a relative path name.
- 9** In the file, look for science software specific entries in each section and make changes to the path names (field 3) as necessary. All path names specified in the PCF must exist on the SGI.
- The science software specific entries will have logical IDs (first field) *outside* of the range 10,000 to 10,999.
 - Where necessary, replace the path names in the third field of each entry with the path names appropriate to the DAAC environment.
 - Do not alter file entries that are used by the SDP Toolkit itself. These have logical IDs *in* the range 10,000 to 10,999.
 - For example, if the following entry was found in the **PCF**:
100|A.granule|/MODIS/run/input|||1
change /MODIS/run/input to the appropriate path name in the DAAC where the file A.granule is stored.
 - When specifying a path name, use **an absolute path name**, not a relative path name.
 - Do not include the file name with the path name. The file name belongs in field 2 by itself.
- 10** In the file, verify that the **SUPPORT OUTPUT FILES** section contains an entry to the shared memory pointer file.
- Look for the entry: **10111|ShmMem|~/runtime|||1**
 - The third field may be blank; this will work too.
 - If this entry is not within this section, add it.
 - Once changes have been made to the **PCF**, save the changes and exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq** and then press the **Enter** key.
 - For other editors, refer to that editor's documentation.
- 11** Again, run the Process Control File Checker on the PCF.

- 12 If the PCF had been checked out of ClearCase, at the UNIX prompt on the SGI, type **cleartool checkin -nc *PCFfilename*** and then press the **Enter** key.
- The ***PCFfilename*** is the file name of the modified **PCF**. The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
-

Setting Up an SDP Toolkit Environment

The purpose of the SDP Toolkit is to allow science software to be developed at independent SCFs for use in DAAC production systems and to provide:

- An interface to the system, including PDPS and CSMS and information management.
- A method for Science software to be portable to different platforms at the DAAC.
- A method to reduce redundant coding at the SCF.
- Value added functionality for science software development.

The SDP Toolkit is divided into two groups of tools:

- Mandatory tools.
- Optional tools.

Mandatory Tools

- **Error and Status Message Facility (SMF)** - provides general error handling, status log messaging, and interface to CSMS services.
- **Process Control Tools** - provides the primary interface to the PDPS. Allows access to physical filenames and file attributes and retrieval of user defined parameters.
- **Generic Input/Output** - provides the means to open and close support, temporary and intermediate duration files.
- **Memory Allocation Tools** - simple wrappers on native C functions which track memory usage in the SDPS, and shared memory tools that enable the sharing of memory among executables within a PGE.

Optional Tools

- **Ancillary Data Access** - provides access to NMC data and Digital Elevation (DEM) data.
- **Celestial Body Position** - locates the sun, moon and the planets.
- **Coordinate System Conversion** - coordinate conversions between celestial references.

- **Constant and Unit Conversion** - physical constants and unit conversions.
- **IMSL** - mathematical and statistical support.

In the description of the Toolkit routines, descriptive information is presented in the following format:

TOOL TITLE

NAME: Procedure or routine name

SYNOPSIS: C: C language call

FORTRAN: FORTRAN77 or Fortran90 language call

DESCRIPTION: Cursory description of routine usage

INPUTS: List and description of data files and parameters input to the routine

OUTPUTS: List and description of data files and parameters output from the routine

ENTERS: List of returned parameters indicating success, failure, etc.

EXAMPLES: Example usage of routine

NOTES: Detailed information about usage and assumptions

REQUIREMENTS: Requirements from PGS Toolkit Specification, Oct. 93 which the routine satisfies

The science software delivered to the DAACs is expected to work with either the SCF SDP Toolkit or the DAAC SDP Toolkit, both of which are installed at each DAAC. During the pre-SSI&T initial testing, the SCF Toolkit should be used.

There are several versions of the SCF/DAAC SDP Toolkit installed on the SGI Power Challenges at the DAACs. The toolkit versions at the DAACs differ according to:

- **Object Type** - The operating system on the SGI is a 64-bit operating system. To be backward compatible, the SGI operating system will allow new 64-bit and 32-bit objects to be built as well as the older 32-bit machines. Each of these object types is designated by placing a cc flag on the command line to enable a particular mode with the SGI C compiler.
 - New 64-bit: cc flag = -64
 - New 32-bit: cc flag = -n32
 - Old 32-bit: cc flag = -32 (SCFs only)
- **Library Type** - The SDP Toolkit uses different libraries depending upon whether FORTRAN 77 or FORTRAN 90 source code is being linked. If C source code is to be linked, then either language version of the library will work.

NOTE: Each Toolkit library comes with a debug version, for example:
sgi32_daac_cpp/
sgi32_daac_cpp_debug/

The choice of which version of the SDP Toolkit to use depends upon two factors: the test being performed and the version required by the science software. For running a PGE in a simulated SCF environment (i.e., as if at the SCF), a SCF version of the Toolkit should be used (see Section 11). For running a PGE in the fully functional DAAC environment, the DAAC version should be used.

Among the DAAC versions, there are six choices. Most science software will likely require one of the 32-bit versions. If FORTRAN 77 code is being used (with or without C), then the FORTRAN 77 language version of the DAAC Toolkit must be used. Conversely, if Fortran 90 code is being used (again, with or without C), the Fortran 90 language version of the DAAC Toolkit must be used.

If both FORTRAN 77 and Fortran 90 are being used, the procedure becomes more complex. Under such circumstances, refer to document 333-EMD-001, *Release 7 SDP Toolkit Users Guide for the EMD Project*.

In addition to the SDP Toolkit interface to the DAAC environment, there are some other interfaces (e.g., MAPI for MODIS codes) and libraries (e.g., OCEAN library for MODIS OCEAN codes) designed to simplify the processes of building and running PGEs at DAACs. Follow the delivered documentation to build specific interfaces and libraries if necessary.

This procedure describes how to set up the appropriate SDP Toolkit environment. It involves two basic steps. First, set the SDP Toolkit home directory in the environment variable PGSHOME. The second step is to source (run) the set up script in the appropriate bin directory. This step results in a number of other environment variables getting set that will be needed.

Setting Up the SDP Toolkit Environment

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The C shell (or a derivative) is the current command shell.

Set Up an SDP Toolkit Environment

- 1 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname*** and then press the **Enter** key.
 - ***ToolkitPathname*** is the home directory of the particular SDP Toolkit version being used. (Refer to Table 4. Note that the setting of **PGSHOME** shown in this table may differ in your local DAAC.)

- Korn shell users, type **PGSHOME=ToolkitPathname**; export PGSHOME and then press the **Enter** key.
- 2 At the UNIX prompt on the SGI, type **source \$PGSHOME/bin/sgiX/pgs-dev-env.csh** and then press the **Enter** key.
- The *sgi* uses **sgi64** for 64-bit version of the Toolkit. Refer to the last column of Table 4, for path names to the file to source.
 - Korn shell users, type **\$PGSHOME/bin/sgiX/pgs-dev-env.ksh** then press the **Enter** key (note the “**dot**” and then space at the beginning of this command).
- 3 This step is optional. Edit the file \$HOME/.cshrc and add the line **alias aliasname ‘setenv PGSHOME ToolkitPathname; source \$PGSHOME/bin/sgiX/pgs-dev-env.csh; echo “textmessage” ‘**
- *aliasname* is the name of the alias. For example, to set up an environment for the DAAC version of the Toolkit for FORTRAN 77 (or C), you might use DAACf77 as an *aliasname*.
 - *ToolkitPathname* is the home directory of the particular **SDP Toolkit** version being used. Refer to Table 4. Note that the setting of **PGSHOME** shown in this table may differ in your local DAAC.
 - The *sgi* Toolkit uses **sgi64** for 64-bit version of the Toolkit.
 - *textmessage* is a message that will be echoed to the screen signifying that a new Toolkit environment has been set up. It must be enclosed within double quotes (“”). An example may be, “DAAC F77 Toolkit environment is now set.”
 - A complete example (it should be all on one line in the **.cshrc** file):


```
alias DAACf77 ‘setenv
PGSHOME/$CUSTOM/TOOLKIT/toolkit/bin/sgi64_daac_f77/; source
$PGSHOME/toolkit/bin/sgi64/pgs-dev-env.csh; echo “DAAC F77 Toolkit
environment is now set” ‘
```
 - Other aliases for other versions of the Toolkit can be set up similarly.
-

Compiling Status Message Facility (SMF) Files

Status Message Facility (SMF) files are used by the SDP Toolkit to facilitate a status and error message handling mechanism for use in the science software and to provide a means to send log files, informational messages, and output data files to DAAC personnel or to remote users.

Science software making use of the SMF need particular header (include) files when being built and also need particular runtime message files when being run. Both the header and message files are produced by running an SMF “compiler” on a message text file. These message text

files should be part of the science software delivery to the DAAC. They typically have a **.t** file name extension.

This procedure describes how to compile the SMF message text files to produce both the necessary include files and the necessary runtime message files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The C shell (or a derivative) is the current command shell.

Check Compile Status Message Facility (SMF) Files

- 1** From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 2** Perform a remote login by typing **ssh hostname** then press the **Enter** key.
 - **hostname** refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
 - It is recommended that this procedure begin within a new command shell on the **SGI**.
- 3** If required, at the UNIX prompt on the SGI, type **cleartool setview ViewName** and then press the **Enter** key.
 - The **ViewName** is the name of a view allowing the SMF files to be accessible.
 - This step is only necessary if any of the SMF files are in ClearCase (in the VOB under configuration management).
- 4** At the UNIX prompt on the SGI, type **setenv PGSHOME ToolkitPathname** and then press the **Enter** key.
 - The **ToolkitPathname** is the home directory of the desired SDP Toolkit version.
- 5** Type **source \$PGSHOME/bin/sgiX/pgs-dev-env.csh** and then press the **Enter** key.
 - **sgiX** refers to the appropriate processor. For example, type **source \$PGSHOME/bin/sgi/pgs-dev-env.csh** and then press the **Enter** key.
- 6** At the UNIX prompt on the SGI, type **cd pathname** and then press the **Enter** key.
 - The **pathname** is the full path name to the directory containing the SMF text files.
 - The SMF text files will typically have **.t file** name extensions.

- 7 At the UNIX prompt on the SGI, type **smfcompile -f *textfile.t* -lang** and then press the **Enter** key.
- **-lang** is a flag that indicates for what language to compile. This flag can be one of **-c** to produce C header files, **-f77** to produce FORTRAN 77 include files, and **-ada** to produce Ada include files. The default is for C include files. For example, type **smfcompile -f77 PGS_MODIS_39123.t** and then press the **Enter** key.
 - ***textfile*** is the file name of the SMF text file delivered with the science software.
 - The SMF text files will typically have **.t** file name extensions.
 - File names for SMF text files usually have the “**seed**” value used by the file as part of its file name (e.g. **PGS_MODIS_39123.t** where **39123** is the seed number).
 - Only one such SMF text file can be compiled at a time; wildcards cannot be used.
 - The SMF compiler may be run with the additional **flags -r and -i** as in, **smfcompile -f *textfile.t* -r -i**. The **-r** automatically places the runtime message file in the directory given by the environment variable **PGSMSG**. The **-i** automatically places the include file in the directory given by the environment variable **PGSINC**. For example, type **smfcompile -ada -r -i -f PGS_MODIS_39123.t** and then press the **Enter** key. Note that the **-f** flag must always be immediately followed by the name of the text file.
- 8 If necessary, at the UNIX prompt on the SGI, type **mv *IncludeFilename* \$PGSINC** and then press the **Enter** key.
- This step is only required if either the **-r** or the **-i flag** were not used in Step 7.
 - ***IncludeFilename*** is the name of the include file created in Step 7.
 - For example, type **mv PGS_MODIS_39123.h \$PGSINC** and then press the **Enter** key.
- 9 Type **mv *RuntimeFilename* \$PGSMSG** and then press the **Enter** key.
- ***RuntimeFilename*** is the name of the runtime message file created in Step 7.
 - For example, type **mv PGS_MODIS_39123 \$PGSMSG** and then press the **Enter** key.
-

This page intentionally left blank.

Building Science Software with the SCF Version of the SDP Toolkit

In order to be tested at the DAAC, science software must be compiled and linked to produce binary executables. These binary executables are then packaged into one or more shell scripts as defined by the science software developer (Instrument Team). These science software packages are the Product Generation Executives (PGEs) delivered to the DAACs during SSI&T. PGEs are the smallest schedulable unit of science software in the DAAC production system.

Building science software into PGEs should be done in accordance with supplied documentation. Such documentation should describe the process in detail. In general, science software deliveries will come with make files or other build scripts to automate the build process.

In general, science software will be built, run, and tested with the SCF version of the SDP Toolkit to ensure that the software has been successfully ported to the DAAC. Once this test has been completed successfully, the science software will be re-built, rerun, and re-tested with the DAAC version of the SDP Toolkit. Only with the DAAC Toolkit can the PGE be run within the DAAC production system.

This procedure describes some general principles that may or may not be applicable to a particular science software delivery for building a PGE with the SCF version of the SDP Toolkit. See Section for Building a PGE with the DAAC version of the SDP Toolkit.

Building Science Software with the SCF Version of the SDP Toolkit - Activity Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The C shell (or a derivative) is the current command shell.

Build Science Software with the SCF Version of the SDP Toolkit (“Typical” Procedure)

- 1 Read all instructional material supplied with the science software delivery. Such material should be the primary source of information on how to build the science software.
 - Read the *Systems Description* document and the *Operations Manual*. Both of these or their equivalent should be in the delivery.
 - Typically, there will be “**readme**” files accompanying each PGE in the directory structure, perhaps in a doc directory.
 - Text files (ASCII) may be viewed with the UNIX command, *more* or with the *vi* editor.
 - PostScript documents may be viewed with *ghostview*, which is accessible via the SSIT Manager.

- PDF formatted documents may be viewed with *acroread*, the Acrobat Reader, also accessible via the SSIT Manager.
 - Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows emulator. The MS Windows emulator may be accessed from the SSIT Manager.
- 2 From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
 - 3 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - *hostname* refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
 - It is recommended that this procedure begin within a new command shell on the SGI.
 - 4 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname*** and then press the **Enter** key.
 - *ToolkitPathname* is the home directory of the desired SDP Toolkit version, in this case, an SCF version.
 - 5 Type **source \$PGSHOME/bin/*sgiX*/pgs-dev-env.csh** and then press the **Enter** key.
 - *sgiX* refers to the appropriate processor. For example, type **source \$PGSHOME/bin/*sgi*/pgs-dev-env.csh** and then press the **Enter** key.
 - 6 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - The *ViewName* is the name of a view allowing the make files to be accessible.
 - 7 Type **cd *pathname*** and then press the **Enter** key.
 - The *pathname* is the full path name of the directory (in the VOB) where the make file has been checked in.
 - 8 Type **cleartool checkout -nc *makefile*** and then press the **Enter** key.
 - The *makefile* is the name of the make file to examine and possibly modify.
 - This step is only necessary if any of the make files (or build scripts) are in **ClearCase** (in the **VOB** under configuration management).
 - 9 Examine and alter (if necessary) any **make** files using any text editor (**vi**, **emacs**).
 - There may be several make files for a particular PGE.
 - Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.

- The Toolkit set up (from Step 6) will set many environment variables that can be used in the make files. To see the current environment variable settings, at the UNIX prompt on the SGI, type **env** and then press the **Enter** key.
- 10 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building (e.g., /usr/ecs/TS1/CUSTOM/ssit/PGE32/message).
 - 11 Verify that the directory structure for the PGE source files matches the directory structure expected by the make files or build scripts.
 - Deliveries may come with install scripts that place files into various directories according to some predefined structure.
 - 12 If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc filename** and then press the **Enter** key.
 - **filename** is the file name of the executable, object file, or make file to be checked out of ClearCase. The **-nc flag** means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
 - Note that checking in executables or object files is **not** recommended in the first place.
 - 13 Build the software in accordance with instructions delivered.
 - Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
 - Choose the most appropriate optimization/debugger flag. During testing, the **-g** is often used. This results in larger and slower executables, but assists in debugging. For production, the **-O** flag may be used to optimize execution time. Variants of the **-g** and **-O** flags may be incompatible.
 - 14 If necessary, at the UNIX prompt on the SGI, type **cleartool checkin filename -nc** and then press the **Enter** key.
 - **filename** is the file name of the executable, object file, or make file to be checked into ClearCase. The **-nc flag** means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
 - Note that checking in executable or object files is **not** recommended.
-

Building Science Software with the DAAC Version of the SDP Toolkit

In general, science software will be built, run, and tested with the SCF version of the SDP Toolkit to ensure that the software has been successfully ported to the DAAC. Once this test has been completed successfully, the science software will be re-built, rerun, and re-tested with the

DAAC version of the SDP Toolkit. Only with the DAAC Toolkit can the PGE be run within the DAAC production system.

This procedure describes some general principles that may or may not be applicable to a particular science software delivery for building a PGE with the DAAC version of the SDP Toolkit.

Building Science Software with the DAAC Version of the SDP Toolkit - Activity Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The C shell (or a derivative) is the current command shell.

To build science software with the DAAC version of the SDP Toolkit, be aware of the “typical” procedure steps that follow:

Build Science Software with the DAAC Version of the SDP Toolkit

- 1** Read all instructional material supplied with the science software delivery. Such material should be the primary source of information on how to build the science software.
 - Read the *Systems Description* document and the *Operations Manual*. Both of these or their equivalent should be in the delivery.
 - Typically, there will be “**readme**” files accompanying each PGE in the directory structure, perhaps in a doc directory.
 - Text files (ASCII) may be viewed with the UNIX command, *more* or with the *vi* editor.
 - PostScript documents may be viewed with *ghostview*, which is accessible via the SSIT Manager.
 - PDF formatted documents may be viewed with *acroread*, the Acrobat Reader, also accessible via the SSIT Manager.
 - Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows emulator. The MS Windows emulator may be accessed from the SSIT Manager.
- 2** From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 3** Perform a remote login by typing **ssh hostname** then press the **Enter** key.
 - *hostname* refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
 - It is recommended that this procedure begin within a new command shell on the SGI.

- 4 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname*** and then press the **Enter** key.
 - *ToolkitPathname* is the home directory of the desired SDP Toolkit version, in this case, a DAAC version.
- 5 Type **source \$PGSHOME/bin/*sgiX*/pgs-dev-env.csh** and then press the **Enter** key.
 - *sgiX* refers to the appropriate processor. For example, type **source \$PGSHOME/bin/*sgi*/pgs-dev-env.csh** and then press the **Enter** key.
- 6 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - *ViewName* is the name of a view allowing the make files to be accessible.
- 7 Type **cd *pathname*** and then press the **Enter** key.
 - *pathname* is the full path name of the directory (in the VOB) where the make file has been checked in.
- 8 Type **cleartool checkout -nc *makefile*** and then press the **Enter** key.
 - The *makefile* is the name of the make file to examine and possibly modify.
 - This step is only necessary if any of the make files (or build scripts) are in ClearCase (in the VOB under configuration management).
- 9 Examine and alter (if necessary) any make files using any text editor (*vi*, *emacs*). If the software had already been built and tested with the SCF version of the SDP Toolkit, this step may be unnecessary.
 - There may be several make files for a particular PGE.
 - Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.
 - The Toolkit set up (from Step 5) will set many environment variables that can be used in the make files. To see the current environment variable settings, at the UNIX prompt on the SGI, type **env** and then press the **Enter** key.
- 10 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building.
- 11 Verify that the directory structure for the PGE source files matches the directory structure expected by the make files or build scripts.
 - Deliveries may come with install scripts that place files into various directories according to some predefined structure.

- 12** If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc *filename*** and then press the **Enter** key.
- *filename* is the file name of the **executable, object file, or make file** to be checked out of ClearCase. The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
 - Note that checking in executable or object files is *not* recommended in the first place.
- 13** Build the software in accordance with instructions delivered.
- Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
 - Choose the most appropriate optimization/debugger flag. During testing, the **-g** is often used. This results in larger and slower executables, but assists in debugging. For production, the **-O** flag may be used to optimize execution time. Variants of the **-g** and **-O** flags may be incompatible.
- 14** If necessary, at the UNIX prompt on the SGI, type **cleartool checkin *filename* -nc** and then press the **Enter** key.
- *filename* is the file name of the **executable, object file, or make file** to be checked into ClearCase. The **-nc flag** means “**no comment**”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
 - Note that checking in executable or object files is *not* recommended.
-

Running a PGE in a Simulated SCF Environment

Science software delivered to the DAACs for SSI&T was developed and tested at individual SCFs using the SCF version of the SDP Toolkit. Before linking the software with the DAAC version of the Toolkit and integrating it with the production system, it is prudent to first link the software to the SCF version of the Toolkit and run it as it was run at the SCF. This type of testing can reveal problems associated with the process of porting the software to another platform whose architecture may be quite different from the one on which the software was developed.

A simulated SCF environment means that the software is built using the SCF version of the Toolkit and is run from the UNIX command line. The Planning and Data Processing System (PDPS) and the Data Server are not involved.

The procedures that follow describe how to run the science software in a simulated SCF environment.

Setting Up the Environment for Running the PGE

Running a PGE that has been built with the SCF version of the SDP Toolkit requires some environment set up as it does at the SCF. This procedure describes how to set up a simulated SCF environment.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The Process Control File (PCF) exists and has been tailored for the DAAC environment.
- The C shell or a derivative (*e.g.* T shell) is the current user shell.

Set Up an Environment for Running the PGE

- 1 From the **SSIT Manager** select **Tools** → **Xterm** from the pull-down menu.
- 2 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the **SPR SGI**.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the **SPR SGI** using **ssh**.
 - It is recommended that this procedure begin within a new command shell on the **SPR SGI**.

- 3 At the UNIX prompt on the SPR SGI, type **setenv PGSHOME *ToolkitPathname*** and then press the **Enter** key.
 - ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version, in this case, an SCF version. For example, **/usr/ecs/TS1/CUSTOM/scf_toolkit_f77**.
 - 4 Type **source \$PGSHOME/bin/*sgiX*/pgs-dev-env.csh** and then press the **Enter** key.
 - ***sgiX*** refers to the appropriate processor. For example, type **source \$PGSHOME/bin/*sgi*/pgs-dev-env.csh** and then press the **Enter** key.
 - 5 At the UNIX prompt on the SPR SGI, type **setenv PGS_PC_INFO_FILE *PCFpathname/PCFfilename*** and then press the **Enter** key.
 - ***PCFpathname*** is the full path name to the location of the Process Control File (PCF) to be associated with this PGE.
 - ***PCFfilename*** is the file name of the PCF.
 - For example, type **setenv PGS_PC_INFO_FILE /disk2/PGE32/PCF/PGE32.pcf** and then press the **Enter** key.
 - 6 Optionally, at the UNIX prompt on the SPR SGI, type **rm *LogPathname/LogFilename*** and then press the **Enter** key.
 - ***LogPathname*** is the full path name to the location of the PGE log files for this PGE.
 - ***LogFilename*** is the file name of the **PGE log file** to remove from a previous run of the same PGE. PGE log files can be **Status**, **User**, or **Report**.
 - ***LogFilename*** may use wildcard characters to remove all of the log files at the same time.
 - This step is optional. If log files from a previous run of the same PGE are not removed, they will be appended with the information from the current run.
 - The environment will then be set up. Continue on the next Section.
 - 7 If necessary, set any other shell environment variables needed by the PGE by sourcing the appropriate scripts or setting them on the command line.
 - For example, for a PGE requiring IMSL, at the UNIX prompt on the SPR SGI, type **source /usr/ecs/TS1/COTS/imsl/vni/ipt/bin/iptsetup.cs** then press the **Enter** key.
 - For some PGEs, the environment variables to be set will be specified in the documentation or the files to source will be supplied in the delivery. Refer to documentation included in the delivery.
-

Running and Profiling the PGE

Profiling a PGE refers to the process of gathering information about the runtime behavior of a PGE. The information includes the wall clock time, user time and system time devoted to the PGE; the amount of memory used; the number of page faults; and the number of input and output blocks.

The Planning and Data Processing System (PDPS) database must be populated with the above information when the PGE is registered with the PDPS during the integration phase of SSI&T. This information may be delivered with the PGE or it may need to be determined at the DAAC during SSI&T. This procedure addresses the latter need.

Note that profiling, as used here, does not involve altering the binary executable to produce instrumented code.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The PGE has been built successfully with the SCF version of the SDP Toolkit.
- The required SMF runtime message files have been produced and placed in the correct locations.
- The Process Control File (PCF) exists and has been tailored for the DAAC environment.
- The required environment for running the PGE has been set up.
- The required input files are available and accessible.
- The C shell or a derivative (*e.g. T shell*) is the current user shell.

Run and Profile the PGE

- 1 At the UNIX prompt on the SPR SGI in the window containing the set up environment, type **cd *PGEbinPathname*** then press the **Enter** key.
 - The *PGEbinPathname* is the full path name of the directory containing the built PGE binary executable. For example: `/usr/ecs/TS1/CUSTOM/ssit/PGE32/bin/`
- 2 At the UNIX prompt on the SPR SGI, type:
`/usr/ecs/MODE/CUSTOM/bin/DPS/EcDpPrRusage PGE.exe >& ResultsOut` then press the **Enter** key.
 - The PGE.exe is the name given to the PGE binary executable.
 - The ResultsOut is the file name in which to capture the profiling results as well as any messages from standard output (stdout) and standard error (stderr) that may be produced by the running PGE. Note that PGEs should not write to stdout or stderr.

- For example: `/usr/ecs/TS1/CUSTOM/bin/DPS/EcDpPrRusage run_PGE32.exe >& ../logs/PGE32.profile.out`
 - The EcDpPrRusage is the profiling program that outputs information about the runtime behavior of the PGE.
 - Depending upon the PGE, it may take some time before the UNIX prompt returns.
- 3** At the UNIX prompt on the SPR SGI, type `echo $status` then press the **Enter** key.
- The \$status is an environment variable that stores the exit status of the previous program run, in this case, the PGE.
 - A status of zero indicates success; a status of non-zero indicates an error of some kind.
 - The meaning of a non-zero exit status should be documented and included with the DAPs.
 - This command must be run immediately after the EcDpPrRusage command.
- 4** At the UNIX prompt on the SPR SGI, type `vi ResultsOut` then press the **Enter** key.
- The ResultsOut is the file name under which the profiling output was saved. Other output of the PGE may also be in this file.
 - The EcDpPrRusage results may then be recorded and used when the PGE is registered in the PDPS.
 - Any text editor/viewer may be used.
-

Checking the PGE for Memory Leaks

One important type of "code inspection" is checking the PGE for memory leaks. This session assumes that PGE build and command line run both been successfully run. The task requires building and running with a memory error checking utility (such as Purify).

The ECS Assistant Functionality Replaced in Part by Scripts and Monitor GUI Whazzup

Given that the system is complex, comprised of many subsystems and components running on multiple heterogeneous host machines, the coordination of all the subsystems for SSI&T is an arduous, time consuming, error prone task. In order to improve our effectiveness and efficiency, an easy-to-use GUI tool, “ECS Assistant,” has been developed to facilitate SSI&T activities.

Currently, the ECS Assistant has lost its capability to monitor the system. This capability now exists with GUI Whazzup. ECS Assistant continues to have the ability to monitor subsystem functions and is mainly used for doing ECS Assistant Subsystem Installs (E.A.S.I.) and staging ESDT/DLL's into the directories, CUSTOM/data/ESS and CUSTOM/lib/ESS respectively. Database Review capability still exists however. The ESDT Manager installation/deletion functions are no longer available nor is the ability to start up and shut down subsystem servers.

The use of scripts provides users with the means to perform functions such as subsystem server startup and shutdown. During the course of performing these tasks, SSI&T operators can use the following scripts to perform the specified functions:

- Start up or shut down multiple servers at the same time, a script is used that accesses a list of subsystem servers.
- Start up and shut down servers individually using a script established within each subsystem.
- Graphically monitor the server up/down status with the Whazzup GUI.
- View ESDTs for SSI&T.
- Review various databases used in the system by using an ISQL Browser established in each subsystem.

In the sections that follow we will address aspects of how to use scripts, the Whazzup GUI and portions of ECS Assistant in our SSI&T activities.

Using Scripts to Start Up/Shut Down Servers

The DAAC may have established their own scripts to start up/shut down subsystem servers. This procedure describes scripts that are used at Landover on the VATC and PVC systems to start up and shut down subsystem servers. The procedure described here will apply to all the servers from different subsystems as well as individual servers.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Start Up/Shut Down Multiple Servers

- 1 To start or kill multiple servers log in to a machine on the system in question.
 - 2 Type `cd /home/cmshared/bin/` then press the **Enter** key.
 - The file HOSTLST (i.e., /home/cmshared/bin/HOSTLST) is a list of the servers that will be affected.
 - 3 To start up the servers type `rcmd_start_mode MODE` then press the **Enter** key; to shut down a server type `rcmd_kill_mode MODE` then press the **Enter** key.
 - Note that script names may vary somewhat.
-

Start Up/Shut Down Individual Servers

- 1 To start or kill a server log in to the individual machine that supports the server in question.
 - 2 Type `cd /home/cmshared/bin/` then press the **Enter** key.
 - 3 To start up a server type `start_mode MODE` then press the **Enter** key; to shut down a server type `kill_mode MODE` then press the **Enter** key.
 - Note that script names may vary somewhat.
 - 4 Repeat Steps 1 through 3 to start up or shut down other servers.
-

Using ECS Assistant

Bringing Up ECS Assistant

Assumptions:

- The ECS Assistant has been properly installed.
- The required environment variables have been set properly.

NOTE: SGI machines are not a part of ECS Assistant functionality.

Run the ECS Assistant

- 1 At the UNIX Console or Terminal type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
- 2 Create an xterm by typing: **xterm -n *hostname* &**
 - The *hostname* is the name of the machine on which the ECS Assistant is to be displayed i.e., the machine that you are using.
- 3 Log into one of the host machines used for SSIT.
- 4 Type **setenv ECS_HOME/usr/ecs** then press the **Enter** key.
- 5 Type **setenv TK_LIBRARY/tools/lib/tk4.2** then press the **Enter** key.
- 6 If necessary, at the UNIX prompt on the host from which the ECS Assistant is to be run, type **cleartool setview *ViewName*** and then press the **Enter** key.
 - The *ViewName* is the ClearCase view to be used while the ECS Assistant is running in this session. For example, type: **cleartool jdoe** and then press the **Enter** key.
 - A ClearCase view is required only if the ECS Assistant needs to be able to “see” into a ClearCase VOB; a view is not necessary otherwise.
- 7 At the UNIX prompt, type **EA** and then press the **Enter** key.
 - File **/tools/common/ea** must exist in the path. (This can be set in the **.cshrc** or **.kshrc** file)
 - **EA** is an alias for: **/tools/common/ea** is the path where ECS Assistant is installed.
 - This will invoke the ECS Assistant GUI with three push buttons for selecting the proper activities, as indicated in Figure 8.
- 8 At the ECS Assistant GUI, click the **Subsystem Manager** pushbutton.
 - This will invoke the Subsystem Manager GUI, as indicated in Figure 9.
- 9 Select a mode by clicking a mode in the mode listing. The mode should be the one to be used for SSI&T.
 - Once the mode is selected, the color of the subsystem name list is changed.
- 10 Select a subsystem with the **Subsystem** radio button.
 - The component list for the selected subsystem will appear in the component window.

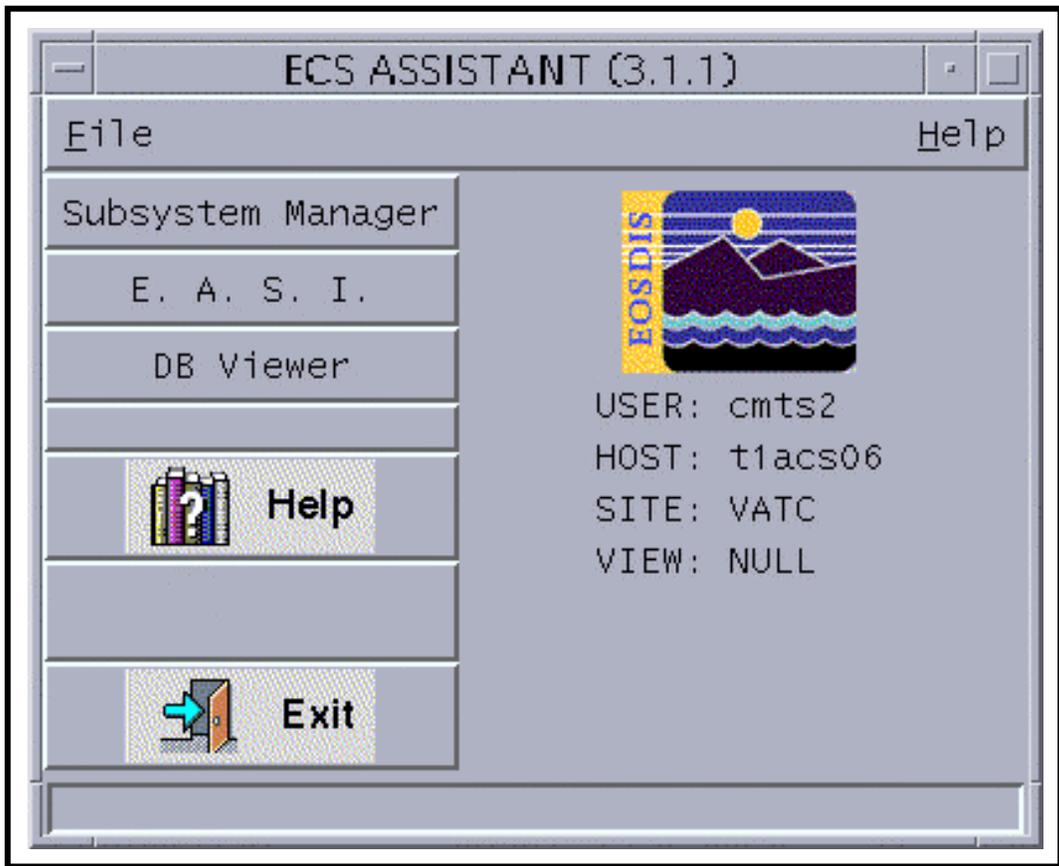


Figure 8. ECS Assistant Main GUI

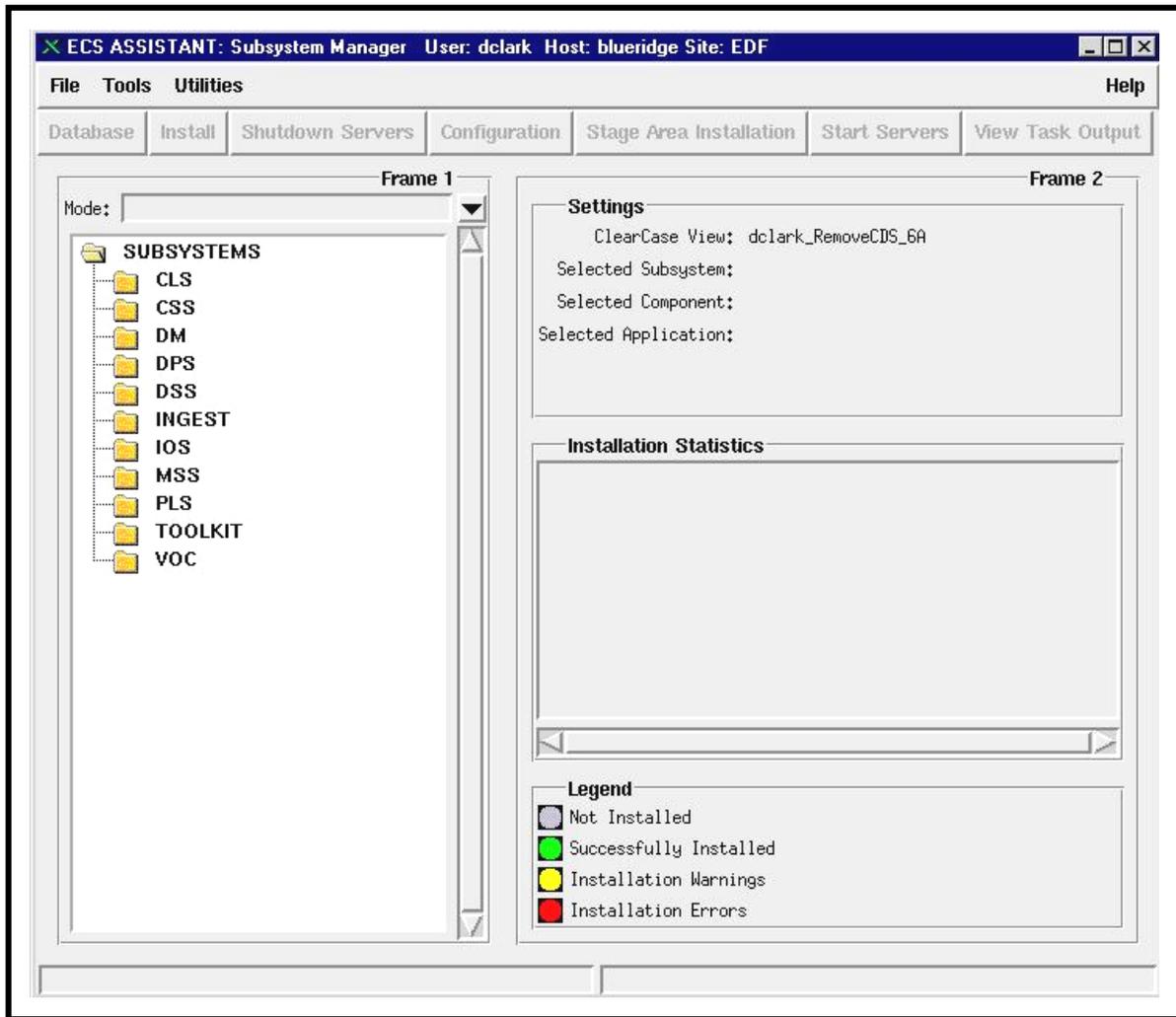


Figure 9. Subsystem Manager GUI

- 11 Select a component by clicking the component name under the component window.
 - The selected component will be highlighted.
 - The server list corresponding to that component will appear in the server window.
- 12 Select a server by clicking the server name from the server list under the servers window.
 - The server selected is highlighted.
- 13 To exit the Subsystem Manager GUI select **File** → **Exit** in the menu bar of the Subsystem Manager GUI.
 - This will terminate the Subsystem Manager GUI.

Using ECS Assistant to View the Science Data Server Database

ESDTs and their granules can be viewed in the Science Data Server database. ECS Assistant provides an easy way to review the records stored in this database by using the ECS Assistant DB Viewer. There are two main windows in the DB Viewer. The first is called Collections and is used to display ESDT information included in the Collection database table. Information listed in this table includes ESDT short names, times last updated, types, etc. If an ESDT is added to the Science Data Server, its record will be shown in this window. The other window is called Granules and is used to display information included in the Granule database table. If a granule is inserted for an ESDT, the granule information will be listed in this window if its ESDT is highlighted in the Collection window. In addition to these two main windows, this DB Viewer GUI can also show ESDT database validation rules, PSA information, and summary information about the database reviewed.

Detailed procedures for the SSI&T operator to perform are provided in the sections that follow.

Assumptions:

- The ECS Assistant has been properly installed.
- The Subsystem Manager is running.
- The environment variables for using the database have been set correctly.

Use ECS Assistant to View the Science Data Server Database

- 1** Invoke the ECS Assistant GUI as described in the procedure to **Run the ECS Assistant** (previous section of this lesson).
 - The ECS Assistant GUI is launched.
- 2** At the ECS Assistant GUI, select the ESDT Manager GUI by clicking on **DB Viewer**.
 - The Database Login GUI will appear as shown in Figure 10.
 - Fill in the fields to point to the specific database for the mode used.
 - Click on **Login** to open the DB Viewer.
 - The DB Viewer GUI will appear as shown in Figure 11.
 - ESDTs are listed in the Collections window.
- 3** To view the inserted granules for a selected ESDT, first select an ESDT by clicking its short name in the **Collections** window.
 - The selected ESDT is highlighted.
 - Granule information for that ESDT, if there is any, will be listed in the Granules window.

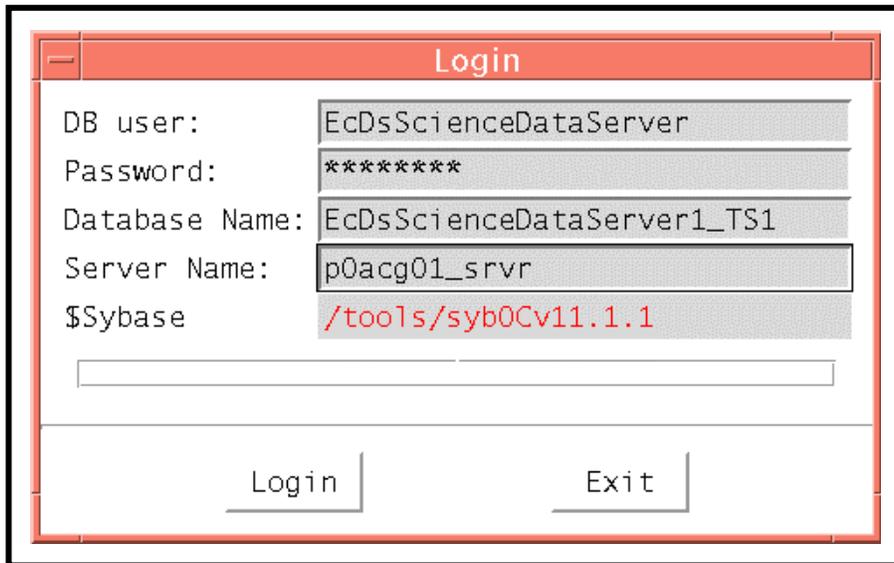


Figure 10. Database Login GUI

- 4 To exit click on the **EXIT** button. This will end the DB Viewer.
-

Monitoring the System Using WHAZZUP Web GUI

Monitor the System Using WHAZZUP Web GUI

- 1 Log in to a server that has Web access and links to the Whazzup GUI.
 - 2 Type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - 3 Type **/tools/bin/ssh *UserId@hostname*** then press the **Enter** key.
 - 4 At the **Password:** prompt type *password* then press the **Enter** key.
 - 5 Type **netscape &** then press the **Enter** key.
 - The Netscape browser is displayed.
 - 6 Select **whazzup** from the list of bookmarks.
 - A GUI such as the one depicted in Figure 12 should appear on the terminal screen.
 - Notice that the bookmarks include links to other EMD-supported sites.
-

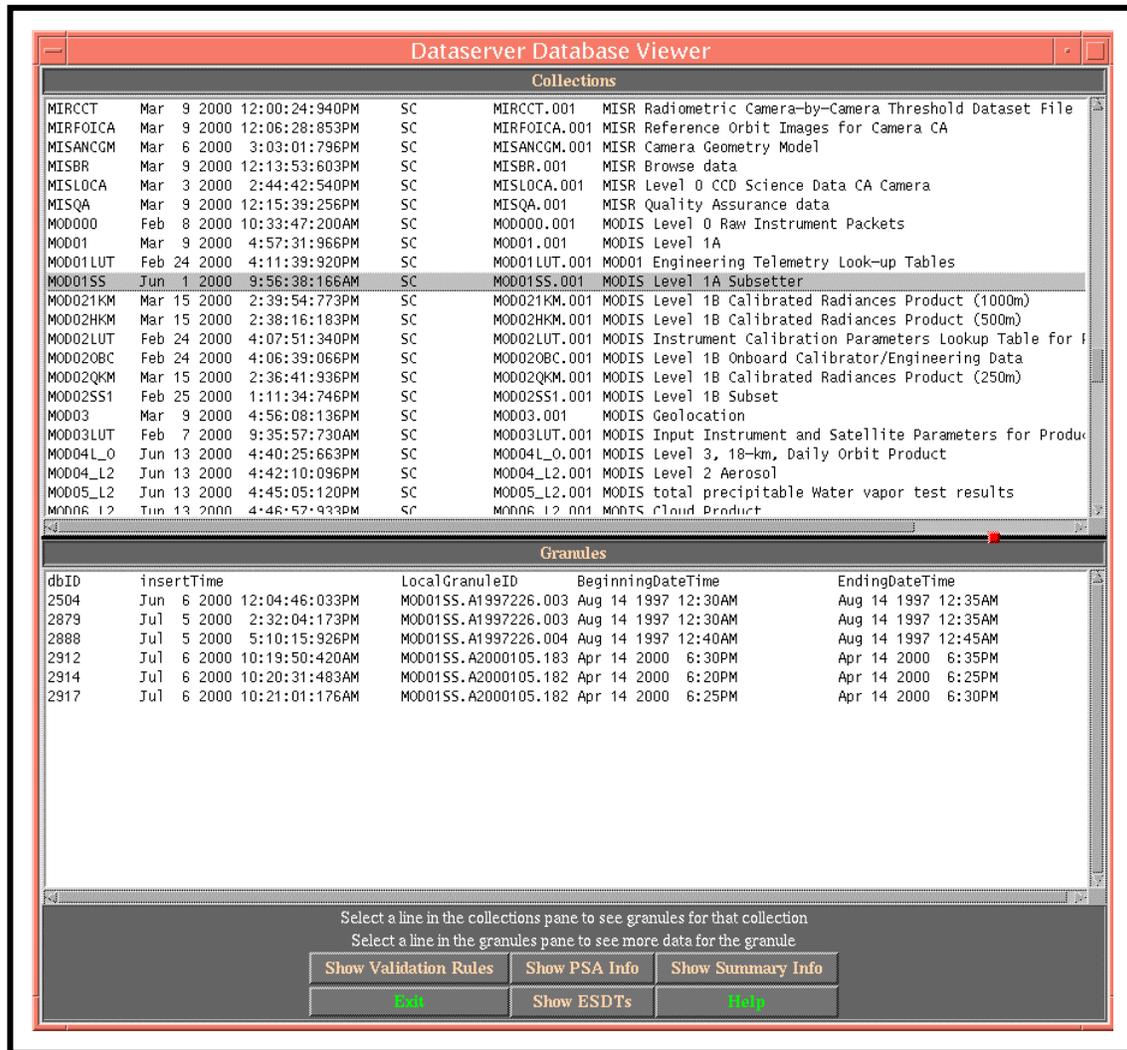


Figure 11. DB Viewer GUI

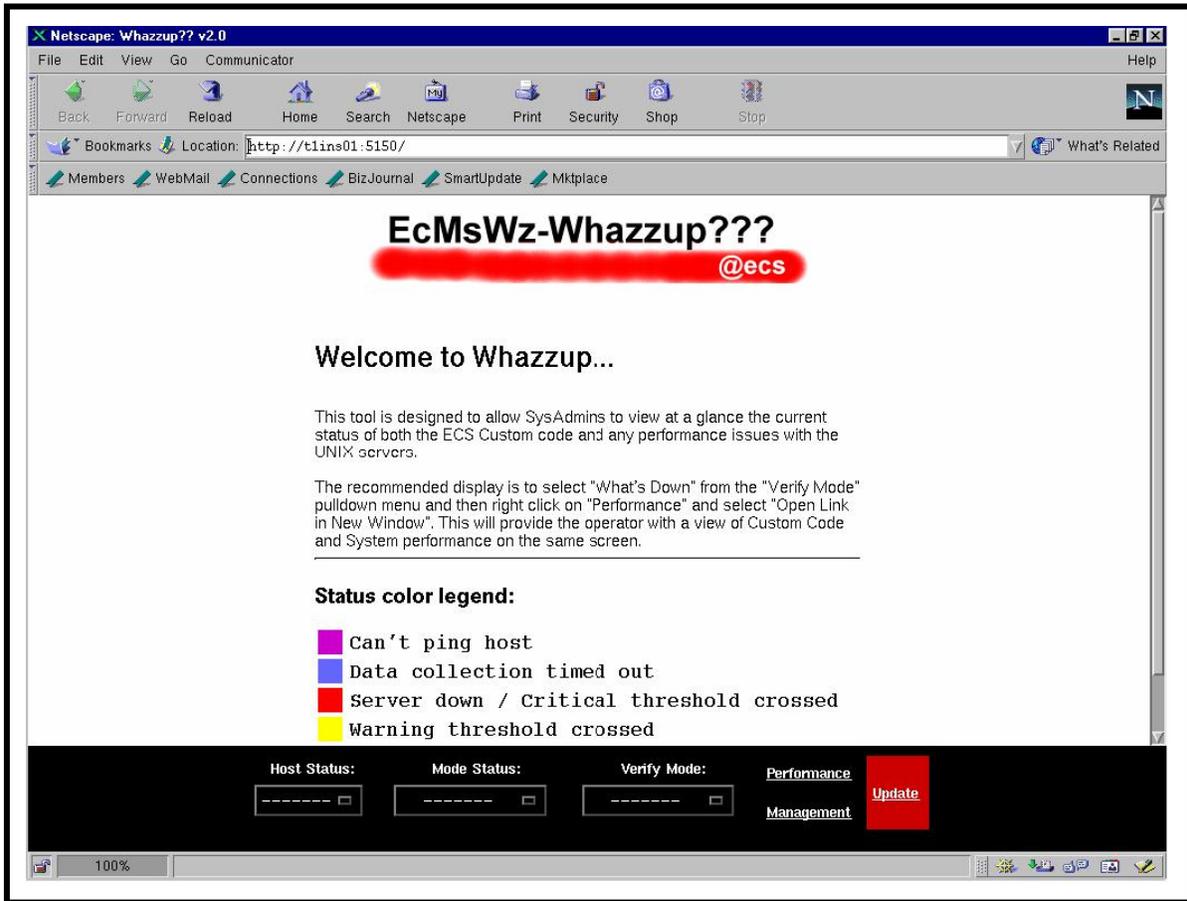


Figure 12. WHAZZUP Monitor GUI

This page intentionally left blank.

ESDT Management

In order for science data to be handled by the system, it must be formally described. At the collection level, that description is the Earth Science Data Type or ESDT. Basically, when an ESDT is defined/installed to the data server subsystem, the SDSRV parses the descriptor into various portions needed by its own CSCIs, and other subsystems.

An entry is made in the SDSRV database containing the meaning of the ESDT, each of its attributes, and each of its services (references to the executable DLLs). The (Sybase) database managed by the SDSRV has sufficient information to satisfy queries sent to the SDSRV.

The ESDT Descriptor file text contains the information mentioned above in an ODL format. The bulk of these files are placed in a given mode during the system install process for that mode. They generally reside in directory `/usr/ecs/MODE/CUSTOM/data/ESS`. In order for these descriptors to be of any use, their information needs to be extracted and parsed to various subsystem databases. This is called the ESDT Install Process.

Also, the ESDT Descriptor files may contain errors or the basic ESDT information is evolved such that the old descriptor information may have to be replaced or updated in the relevant databases.

This section will describe how to check, install, remove or update an ESDT.

Inspecting ESDTs

Before installing or updating an ESDT, one needs to check for its existence. Also, one may want to examine the contents of an ESDT, e.g., what does the header say about the latest changes and when they were made. These types of checks/inspections can be performed with general UNIX tools or with the Science Data Server GUI.

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Science Data Server (EcDsScienceDataServer) for the desired mode is running.
- The Sybase server for the Science Data Server database (e.g., for PVC: p0acg05_srvr) is running.

Launch the Science Data Server GUI

- 1 Log into the Science Data Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
- 2 At the UNIX prompt on the host from which the Science Data Server GUI is to be run, type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.

- 3 Next, type **EcDsSdSrvGuiStart** *MODE* then press the **Enter** key.
 - The Science Data Server GUI (Figure 13) is displayed.
 - 4 Scroll through the **Data Type Information** box.
 - If the shortname and version of the ESDT is displayed, then that ESDT was installed at least to the Science Data Server database.
 - For an ESDT to be fully installed and useful, there are three other databases that need to have been successfully affected by the installation process. These are the Data Dictionary and Subscription databases.
 - If the shortname/version of the desired ESDT does not appear in the box as mentioned, it can be assumed it is not installed in the Science Data Server database. However, it could be installed in one or more of the other three databases.
 - 5 To view the contents of an ESDT Descriptor File, first select the shortname/version in the box then click on **View**.
 - The descriptor file (Figure 14) is displayed.
 - Alternately, one can use a text editor and open up the corresponding ESDT Descriptor file. The installed version of the Descriptor file resides in `/usr/ecs/MODE/COMMON/cfg/DsESDTRDesc`. During the ESDT installation process, the descriptor file is copied from `/usr/ecs/MODE/COMMON/data/ESS` into `/usr/ecs/MODE/COMMON/cfg/DsESDTRDesc`.
 - The descriptor file contains version/date information in its header. The rest of the file is in ODL metadata format and describes the corresponding ESDT to the system.
-

Removing ESDTs

If one wants to install a modified version of an ESDT, retaining the shortname and version number, there are two ways to go. The first way is to completely replace the descriptor file and corresponding ESDT data in the relevant databases then perform an ESDT add. The second way, if the nature of the old and new ESDT permits, is to perform an update. The advantages/disadvantages are summarized below:

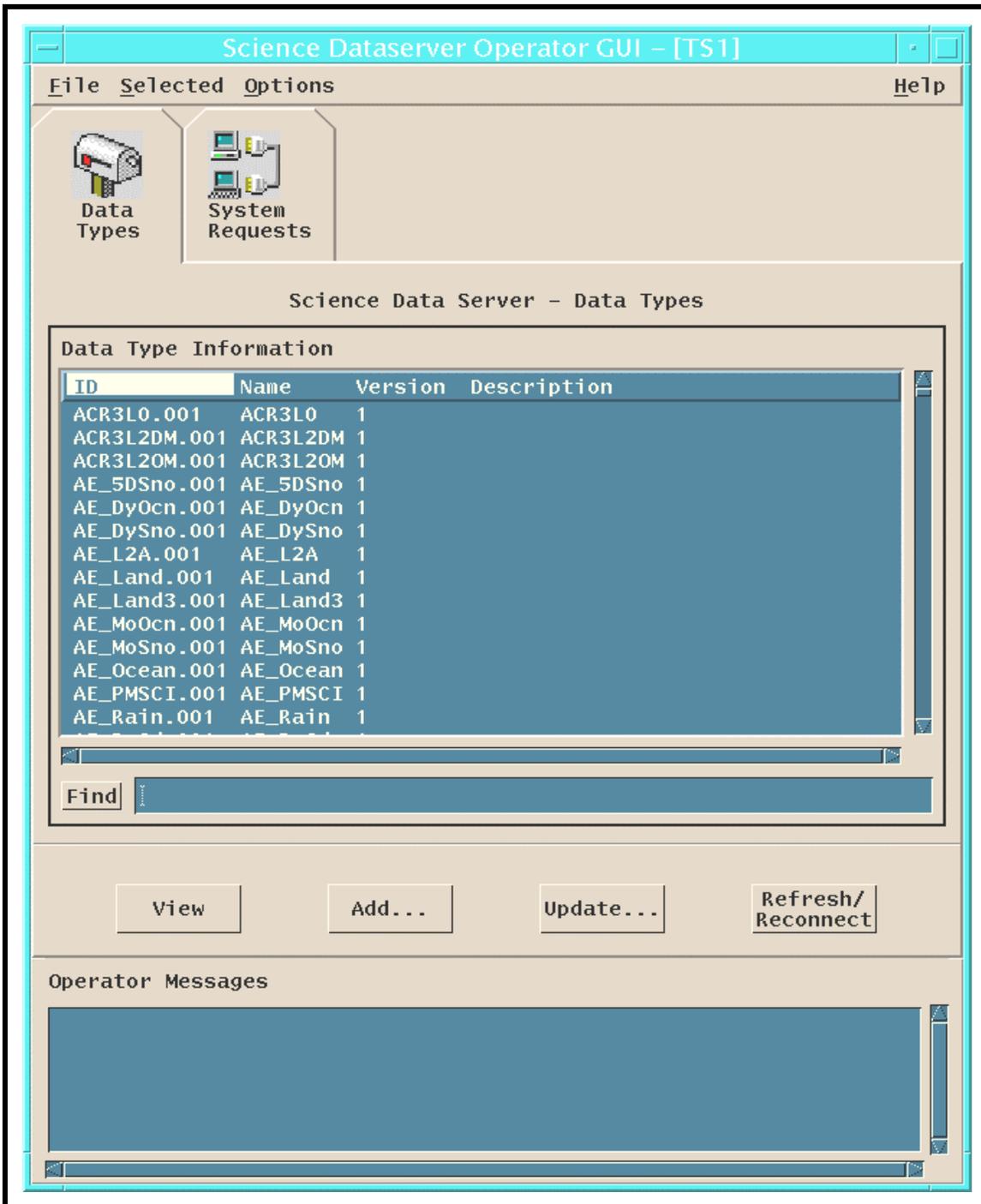


Figure 13. Science Data Server Operator GUI

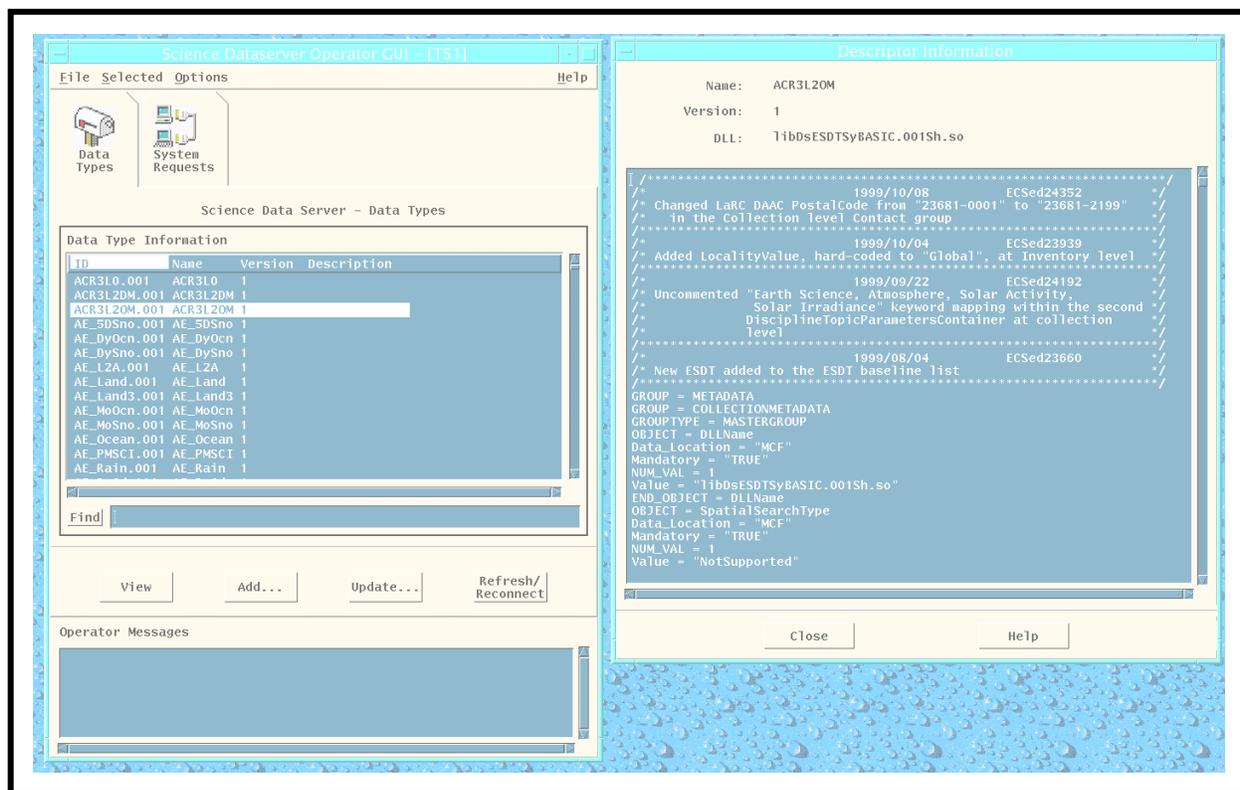


Figure 14. Viewing An ESDT Descriptor File

ESDT Add Advantages

Can be done with any ESDT.

Disadvantages

Prior to Add, ESDT removal scripts must be run for all affected databases.

All granule pointer information is lost for granules belonging to that ESDT.

ESDT Update Advantages

No removal scripts need to be run.

Granule references are preserved.

Disadvantages

Only certain ODL structures supported.

Science Data Server must be brought up with a StartTemperature=maintenance. This means it is unusable by anything else until it is recycled with an operational StartTemperature.

Many other conditions must be met.

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Sybase servers for the involved databases are working properly.

- The user knows the various IDs, passwords and parameters required by the scripts that will be used.

To remove an ESDT from the system for a given mode, execute the steps that follow:

Remove an ESDT From the Data Dictionary Database

- 1 Log into the (DMS) Data Dictionary Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
 - 2 At the UNIX prompt on said host type `cd /usr/ecs/MODE/CUSTOM/dbms/DMS` then press the **Enter** key.
 - This subdirectory contains the script to be run - **DmDbCleanCollection**.
 - 3 Prior to running the script, assign proper values to the following four environmental parameters:
 - **DSQUERY** points to the Sybase server that contains the Data Dictionary database.
 - **DBNAME** is the name of the Data Dictionary database.
 - **DBUSERNAME** is the Data Dictionary login ID.
 - **DBPASSWORD** is the Data Dictionary login password.
 - To set the environmental parameters in C shell, follow the sample steps that follow.
 - At the UNIX prompt type `setenv DSQUERY p0ins02_srvr` then press the **Enter** key.
 - At the UNIX prompt type `setenv DBNAME EcDmDictService_TS1` then press the **Enter** key.
 - At the UNIX prompt type `setenv DBUSERNAME *****` then press the **Enter** key.
 - At the UNIX prompt type `setenv DBPASSWORD *****` then press the **Enter** key.
 - 4 Type `DmDbCleanCollection ShortName VersionID` then press the **Enter** key.
 - *ShortName* is the ESDT shortname. and the version number for said shortname. The form is `DmDbCleanCollection ShortName Version`. A PVC example would be:
 - *VersionID* is the version number for the shortname.
 - For example: `DmDbCleanCollection AE_5Dsno 001`
-

Remove an ESDT from the Subscription Database

- 1 Log into the (CSS/IDG) Subscription Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
 - 2 At the UNIX prompt on said host type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.
 - This subdirectory contains the script to be run - `dbDeleteEvents.csh`.
 - 3 Type `dbDeleteEvents.csh MODE ShortName VersionID UserName Password` then press the **Enter** key.
 - *MODE* is the DAAC mode to be affected by the script.
 - *ShortName* is the ShortName of the ESDT to be removed.
 - *VersionID* is the version of ShortName to be removed.
 - *UserName* is the Subscription database login username.
 - *Password* is the Subscription database login password.
 - For example, to remove the ESDT with ShortName `AE_5Dsno` (version 1) from the Subscription database in mode TS1 on the PVC, at the UNIX prompt, type `dbDeleteEvents.csh TS1 AE_5Dsno 001 ***** *****` then press the **Enter** key.
-

Remove an ESDT from the Science Data Server Database

- 1 Log into the (DSS/SDSRV) Science Data Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
- 2 At the UNIX prompt on said host type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.
 - This subdirectory contains the script to be run - `EcDsSrRmesdt`.
- 3 Type `EcDsSrRmesdt MODE DescripFileName1 ... DescripFileNameN` then press the **Enter** key.
 - *MODE* is the DAAC mode to be affected by the script.
 - *DescripFileName1* is the name of an ESDT descriptor file that corresponds to an ESDT to be removed.

- **DescripFileNameN** is the name of an ESDT descriptor file that corresponds to an ESDT to be removed. More than one ESDT may be removed when running the script **EcDsSrRmesdt**. For each ESDT to be removed, a corresponding ESDT descriptor file name is required.
- For example, to remove the ESDT with ShortName **AE_5Dsno** (version 1) from the Science Data Server database in mode TS1, at the UNIX prompt type **EcDsSrRmesdt TS1 DsESDTAmAE_5Dsno.001.desc** then press the **Enter** key.

NOTE: The ESDT descriptor files are located in two subdirectories. The "holding area" is `/usr/ecs/MODE/CUSTOM/data/ESS`. They are first put here during mode drop installations or manually as new versions are delivered. When an ESDT is installed into the Science Data Server database, the descriptor file is copied from the "holding area" and placed into `/usr/ecs/MODE/CUSTOM/cfg/DsESDTEDesc`.

NOTE: The ESDT version ID number is incorporated in the descriptor file name as exemplified by the location of ".001" in **DsESDTAmAE_5Dsno.001.desc**.

Adding ESDTs

Generally, an ESDT or group of ESDTs are Added using the Science Data Server GUI. The single biggest disadvantage of doing an ESDT add over performing an ESDT update is that one must remove the ESDT (if it exists) before performing the add. Aside from the labor involved removing an ESDT, once the ESDT is removed, all granule pointers in the various databases for that ESDT are lost. If one wants to reference the granules after the revised ESDT is added, the granules will have to be reinserted. This would definitely impact ongoing operations.

The main advantage of adding an ESDT is that the add operation itself is fairly simple.

Adding an ESDT Using the Science Data Server GUI

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Sybase servers for the involved databases are working properly.
- The subsystem servers associated with the involved databases are running.
- If the ESDT to be added already exists, it will be removed from the involved databases before the add operation takes place.

Add an ESDT to the System for a Given Mode

- 1 Log into the (DSS) Science Data Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.

- 2 At the UNIX prompt on said host type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.
- 3 Type `EcDsSdSrvGuiStart MODE` then press the **Enter** key.
 - This brings up the Science Operator Data Server GUI as shown in Figure 15.

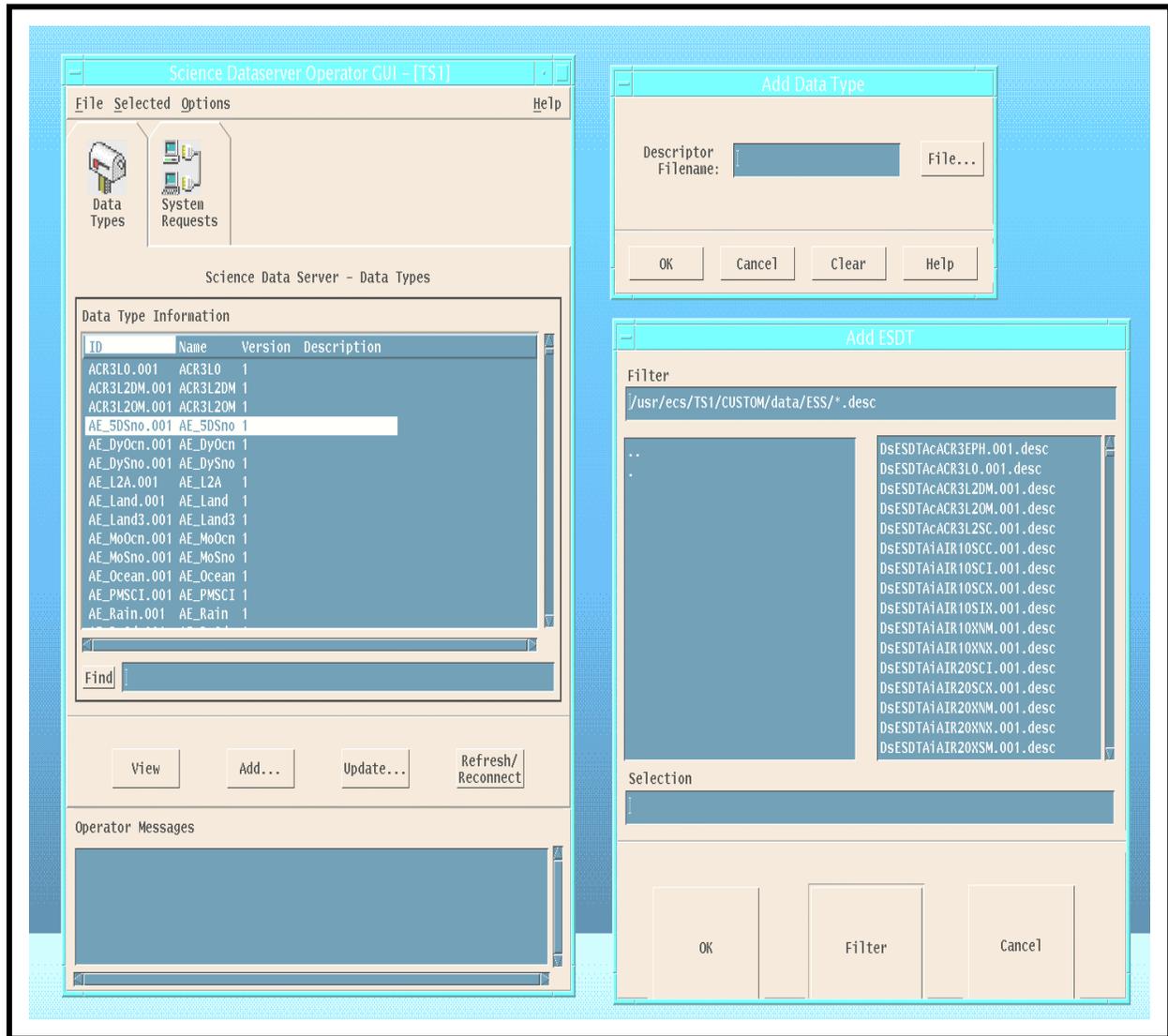


Figure 15. Adding An ESDT - The GUIs Involved

- 4 Click the **Add...** button on the Science Data Server Operator GUI.
 - This brings up the **Add Data Type** GUI as shown in Figure 15.

- 5 The user can type in the descriptor file name for the ESDT to be added. However, it is usually easier and less prone to error, to click the **File...** button.
 - This brings up the **Add ESDT** GUI as shown in Figure 15.
 - The **Add ESDT** GUI displays a list of the eligible descriptor files that can be used to add an ESDT.
 - 6 Tailor the list of the eligible descriptor files by making the appropriate entry in the **Filter** box.
 - Select one or more descriptor file names from the file list. To select more than one file name, the user needs to hold down the **Control** or **Shift** key while clicking on the desired file names.
 - For each descriptor file selected, the corresponding ESDT will be added when the process is complete.
 - 7 When the desired descriptor files have been selected, click in the **OK** box of the **Add ESDT** GUI.
 - The **Add Data Type** GUI will become populated with the descriptor file selections.
 - 8 Click the **OK** box in the **Add Data Type** GUI.
 - The Add ESDT process will start.
 - The **Operator Messages** will display the status of the install process.
 - If the install process seems to have errors, go to the `/usr/ecs/MODE/CUSTOM/logs` directories on the various server platforms and check the logs for the servers involved as well as for the "Science Data Server Operator GUI" log.
 - If everything went well, the selected ESDTs have been added.
-

Updating ESDTs

Generally, an ESDT or group of ESDTs are Updated using the Science Data Server GUI. There are certain conditions that a given set of descriptors must meet in order for an Update to be possible. Furthermore, the Science Data Server has to be running with a StartTemperature value of "**maintenance**" in order for the Update function to work. This means the mode involved is unusable by anyone else. The main advantage of performing an Update is that the old ESDT doesn't have to be removed first, thus preserving the system's knowledge of any granules that were inserted with the ESDT shortname that is being updated. This is definitely a plus for an operational mode.

Following is a listing of what the Update ESDT Capability can do. Implicit in this are the things it can't do.

- Add optional Collection level metadata
- Add optional Inventory level metadata (including Product Specific Attributes (PSAs))
- Add additional services
- Add additional events
- Add new parameters to existing services
- Add qualifiers to existing events
- Add additional valid values to Inventory level metadata attributes
- Change values of single and multi-value Collection level metadata attributes (exceptions: AdditionalAttributeName, AdditionalAttributeType, AnalysisShortName, CampaignShortName, InstrumentShortName, PlatformShortName, SensorCharacteristicName, SensorCharacteristicType, SensorShortName, ShortName, VersionID, and type)
- Change a mandatory attribute to optional
- Modify parameters in existing services

One thing an ESDT Update explicitly can't do:

- Add Mandatory attributes.

It is clear from the above, that before an ESDT Update is attempted, consultation with an ESDT specialist is advised.

Updating an ESDT Using the Science Data Server GUI

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Sybase servers for the involved databases are working properly.
- The subsystem servers associated with the involved databases are running.

Update an ESDT in the System for a Given Mode

- 1 Log into the (DSS) Science Data Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
- 2 At the UNIX prompt on said host type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.

- 3 Type **EcDsSdSrvGuiStart** *MODE* then press the **Enter** key.
 - This brings up the Science Operator Data Server GUI.
 - 4 Click the **Update...** button on the Science Data Server Operator GUI.
 - This brings up the small **Update ESDT** GUI.
 - 5 Either type in the descriptor file name for the ESDT to be updated or (usually easier and less prone to error) click on the **File...** button.
 - This brings up the large **Update ESDT** GUI.
 - The large **Update ESDT** GUI displays a list of the eligible descriptor files that can be used to update an ESDT.
 - 6 Tailor the list of the eligible descriptor files by making the appropriate entry in the **Filter** box.
 - Select one or more descriptor file names from the file list. To select more than one file name, the user needs to hold down the **Control** or **Shift** key while clicking on the desired file names.
 - For each descriptor file selected, the corresponding ESDT will be updated when the process is complete.
 - 7 When the desired descriptor files have been selected, click in the **OK** box of the large **Update ESDT** GUI.
 - The small **Update ESDT** GUI will become populated with the descriptor file selections.
 - 8 Click the **OK** box in the small **Update ESDT** GUI.
 - The Update ESDT process will start.
 - The **Operator Messages** will display the status of the install process.
 - If the install process seems to have errors, go to the `/usr/ecs/MODE/CUSTOM/logs` directories on the various server platforms and check the logs for the servers involved as well as for the "Science Data Server Operator GUI" log.
 - If everything went well, the selected ESDTs have been updated.
-

ESDT Volume Group Configuration

Once an ESDT is installed, the system knows how to deal with the collections and granules associated with that ESDT - up to a point. Storage Management in the Data Server Subsystem needs some additional information for its database so that it knows where to archive and retrieve the data associated with a given ESDT. This is the ESDT Volume Group information. When an

insert or acquire is performed, Storage Management needs to know which HWCI (Hardware CI) and directory are involved.

This Volume Group information can be created and modified using the Storage Management GUI. The GUI start script is `EcDsStmgtGuiStart` and resides in the standard utilities directory for each mode. This GUI normally resides on the (DSS) Data Distribution Server platform. In the PVC, for example, that platform is `p0dis02`.

Modifying an ESDT's Volume Group Information

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Sybase server for the Storage Management databases is working properly.

Modify a Given ESDT's Volume Group Information

- 1 Log into the (DSS) Data Distribution Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
- 2 At the UNIX prompt on said host, type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.
- 3 Type `EcDsStmgtGuiStart MODE` then press the **Enter** key.
 - This brings up the Storage Management Control GUI as shown in Figure 16.
- 4 Click the **Vol Grp Config** tab on the GUI.
 - This brings up the "Volume Group Information" pane, which is what Figure 16 actually illustrates.
- 5 Select an ESDT to be modified by scrolling through the Volume Group Information pane and clicking on the **Data Type Name** (ShortName.VersionID) desired.
- 6 Click on the **Modify** button.
 - The Modify Volume Groups GUI as shown in Figure 17.
- 7 Type in only the information that needs to be changed.
 - The HWCI information must be selected from a list that is brought up by clicking on the option button associated with the **New HWCI** field.
 - A view of the Modify Volume Groups GUI with user entries made is illustrated with Figure 18.

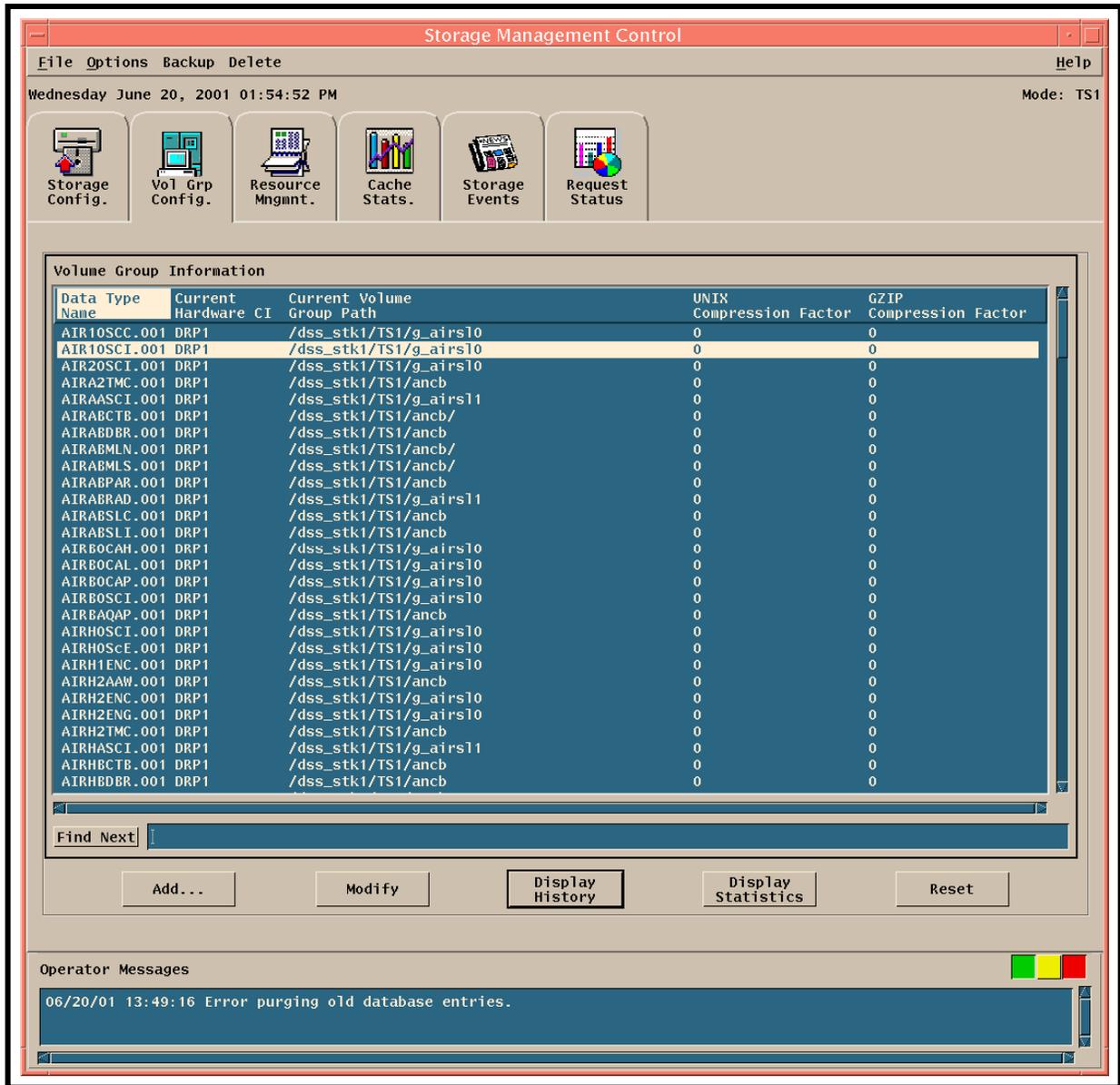


Figure 16. Storage Management GUI

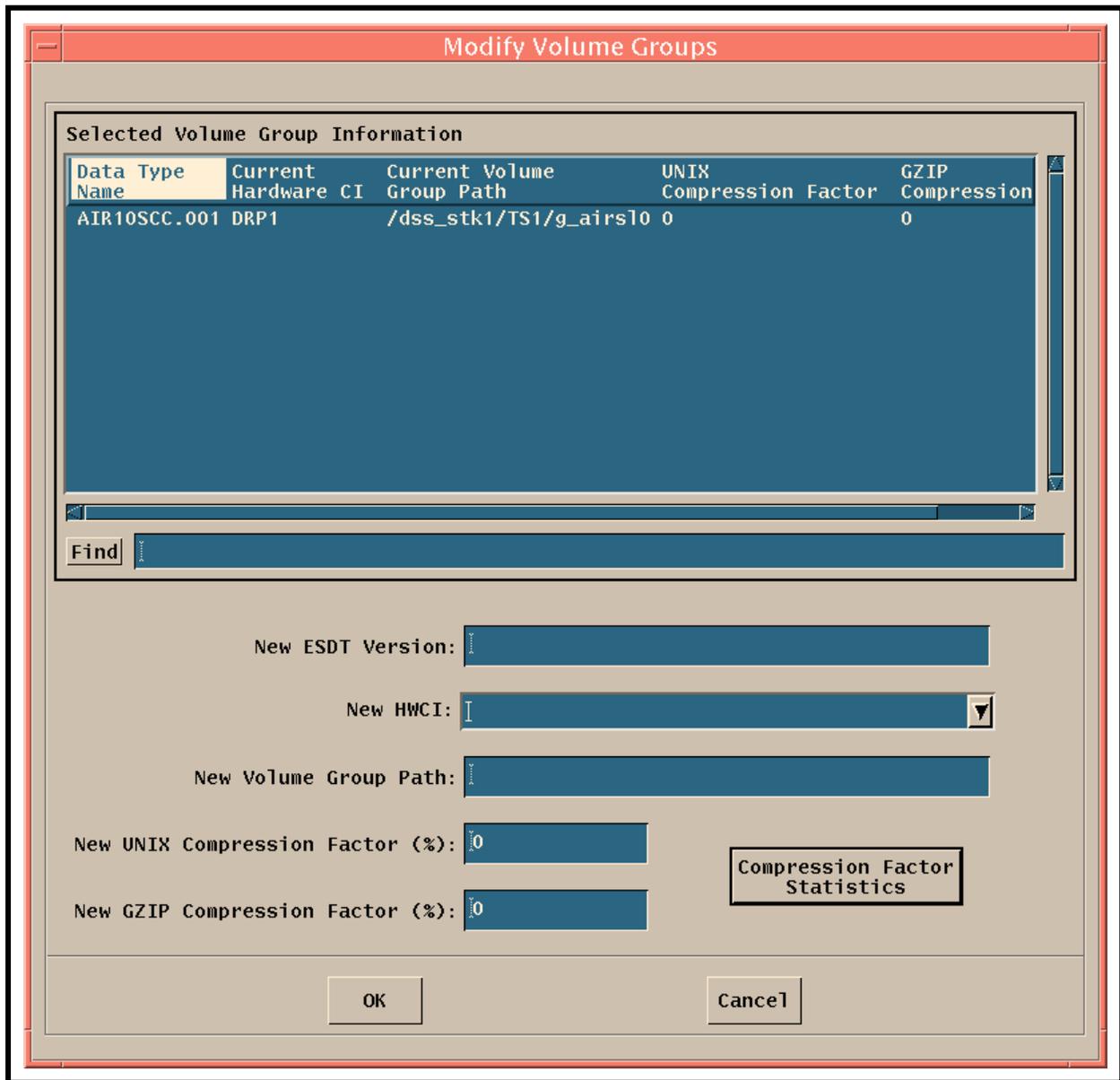


Figure 17. Modify Volume Groups GUI

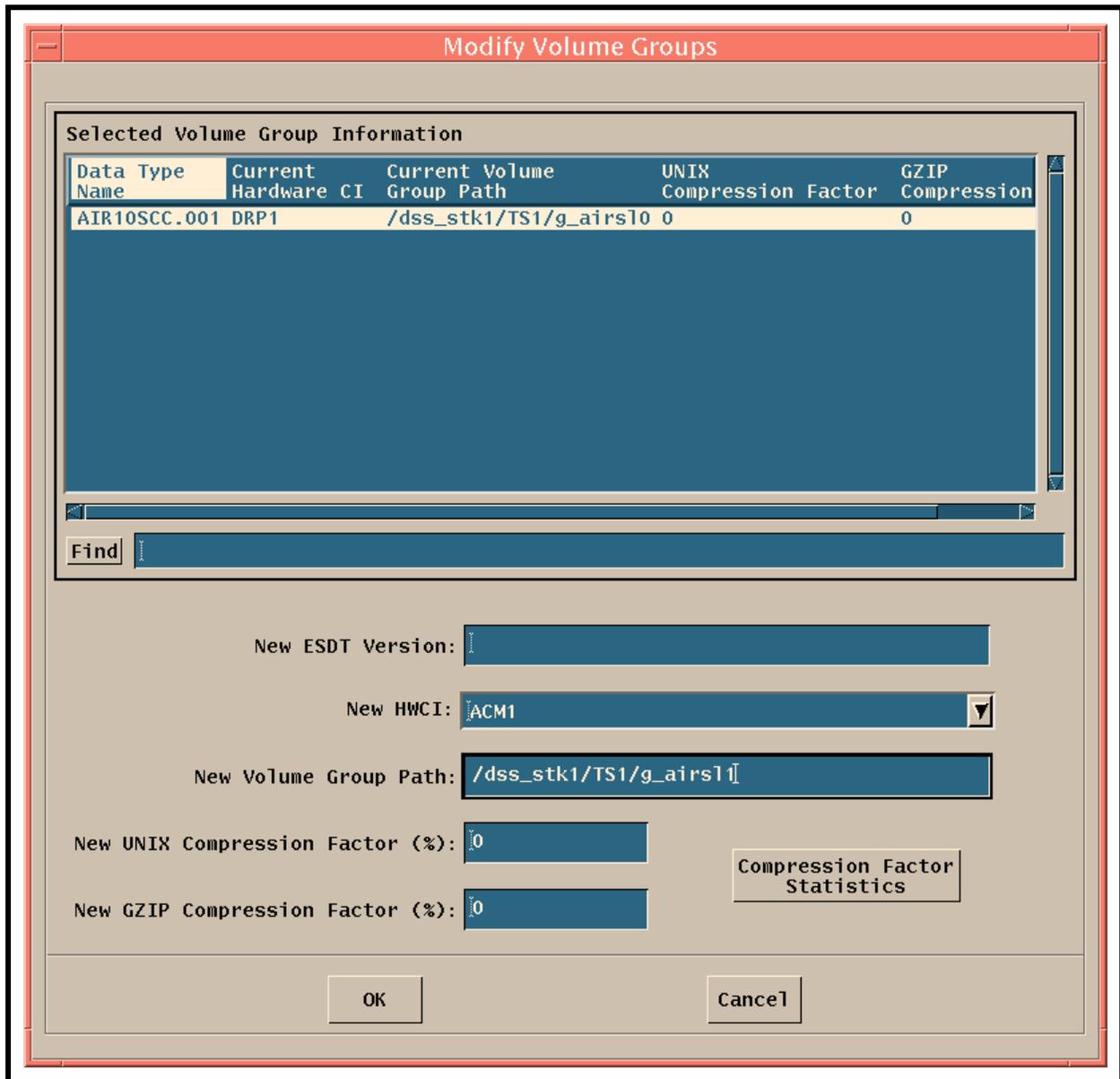


Figure 18. Modify Volume Groups GUI with Entries

- 8 Click on the **OK** button when satisfied with the modifications entered.
 - The Update ESDT process will start.
 - The **Operator Messages** field displays the status of the install process.

Adding an ESDT's Volume Group Information

Assumptions:

- The required environment variables (e.g., **DISPLAY**) have been set properly.
- The Sybase server for the Storage Management databases is working properly.

To enter information for an ESDT not already entered into the Storage Management's database (i.e., the ESDT.VersionID doesn't appear in the Volume Group Information pane of the Storage Management Control GUI) for a given mode, execute the steps that follow:

Add an ESDT Volume Group

- 1** Log into the (DSS) Data Distribution Server host.
 - The file `.sitemap` under `/usr/ecs/MODE/CUSTOM` contains this information.
 - 2** At the UNIX prompt on said host, type `cd /usr/ecs/MODE/CUSTOM/utilities` then press the **Enter** key.
 - 3** Type `EcDsStmgtGuiStart MODE` then press the **Enter** key.
 - This brings up the Storage Management Control GUI (Figure 16).
 - 4** Click the **Vol Grp Config** tab on the GUI.
 - This brings up the "Volume Group Information" pane, which is what Figure 16 actually illustrates.
 - 5** Click on the **Add...** button.
 - The **Add Volume Group** GUI (Figure 19) is displayed.
 - 6** Type in all the required information.
 - The HWCI information must be selected from a list that is brought up by clicking on the option button associated with the HWCI field.
 - 7** Click on the **OK** button when satisfied with the information entered.
 - The Add Volume Group process will start.
 - The **Operator Messages** will display the status of the install process.
-

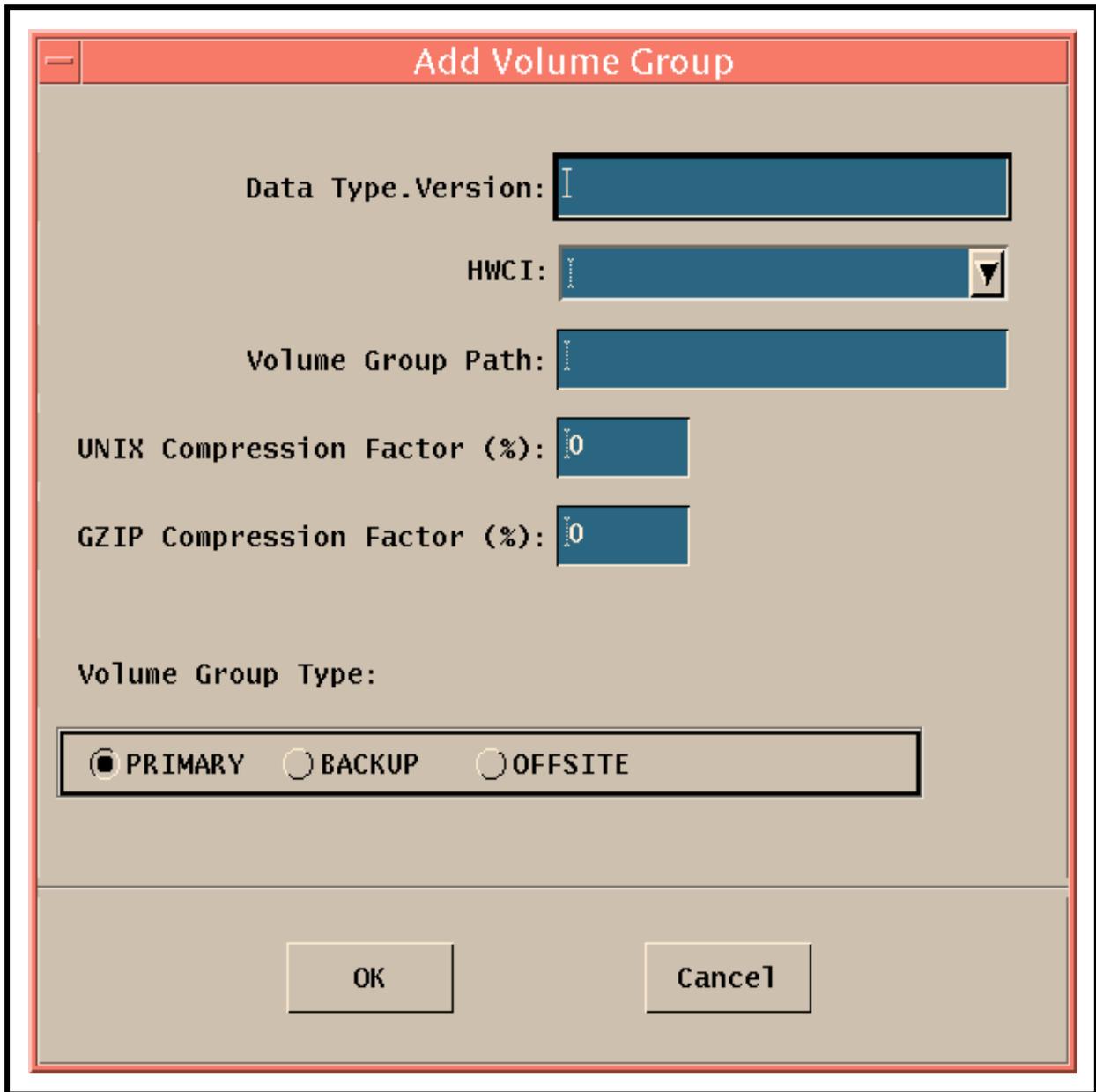


Figure 19. Add Volume Group GUI

Required Operating Environment

For information on the operating environment, tunable parameters, and environment variables refer to 910-TDA-022, *Custom Code Configuration Parameters for ECS*. The document is available at <http://cmdm-ldo.raytheon.com/baseline/> under "Technical Documents."

This page intentionally left blank.

Production Rules

Production Rules

Production rules are the instructions about how a particular PGE is to be run. The instructions specify a wide range of information such as the input and output data types, the frequency of execution, activation conditions, and error handling instructions.

Production rules are described in detail in 625-EMD-006, *Training Material for the EMD Project Volume 6: Production Planning and Processing*. Refer to that lesson for further information.

This page intentionally left blank.

Data Preprocessing (DPREP)

Data Preprocessing (DPREP)

Data Preprocessing (DPREP) is a vital part of SSI&T. A general description of DPREP for Terra, Aqua, and Aura is provided in 625-EMD-006, *Training Material for the EMD Project Volume 6: Production Planning and Processing*, and has not been duplicated in this lesson. Refer to the Production Planning and Processing lesson for further information.

This page intentionally left blank.

Updating the Orbit Model

Updating the Orbit Model

To determine real-time the latest Orbit Start times, Orbit Period, Path Number and Orbit Number, PDPS takes in specific information about the orbit of the satellite during initial SSI&T. This information then becomes the basis for predictions of future orbit start times and numbers. Because this value is accurate within a fraction of a second of time, the satellite may “drift” or a correction to orbit, known as a “burn” may have been applied. Therefore, the satellite Orbit Start Time can get out of sync either +/- with reality. The consequences are an elapse in time that will affect the Production Request Editor’s ability to find a granule that should match with a DPR, or an incorrect Orbit Time could be passed to the PGE. The update of Orbit parameters will be done weekly at a specific time with scripts specifically written to extract the new Orbit Parameters from the most recent DPREP output file. These parameters in turn will be inserted manually to the ORBIT.ODL file. Then SSI&T personnel will re-register the Orbit.ODL file in PDPS. The DAAC Support Help Desk team is responsible for knowing when changes to Orbit location have taken place from the Flight Dynamics System (FDS). A KnowledgeBase with backup procedures will be maintained for contingencies concerning Orbit Model updates. DPREP processing will be the most likely place to experience a failure due to Orbit time sync error encounters. The restoration of Orbit parameters with new values from FDS will most likely be necessary. The following procedures are provided to bring about an updated Orbit Model upon receipt of updated orbit parameters:

- ORBIT_NUMBER.
- ORBIT_PERIOD.
- ORBIT Path Number.
- ORBIT_START Time.

Update the Orbit Model

- 1 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - ***hostname*** refers to the host on the (ais) system where ODL files are stored.
- 2 Type **cd /usr/ecs/MODE/CUSTOM/data/DPS/ODL** then press the **Enter** key.
- 3 Select the ORBIT.odl file that is currently being used.
- 4 Using **vi** update the following files with the new parameter values received:
 - **ORBIT_AM1.odl** and/or **ORBIT_EOSAM1.odl** if they both are in use.

- 5 Have someone double-check your entries for accuracy before proceeding to the SSIT Manager for registering the new ODL file in the PDPS system.
- 6 For **Test Data** only - determine the Instrument PGE ODL that will be updated (MISR, MODIS, etc.).
 - Using **vi** update the corresponding PATHMAP_Instrument_.odl file with the new parameter values received.
 - Ensure that the ABSOLUTE_PATH and MAPPED_PATH parameters agree with those in the new ORBIT_XXXX.odl.
- 7 SSI&T personnel will execute an Orbit Model Update by running a Dummy PGE established for this purpose at each of the DAACs.

NOTE: A dummy PGE is run because a normal PGE cannot be re-registered if any DPRs exist in the system.

- 8 Notify the DAAC Operations Supervisor that the Orbit Model has been updated.
 - The DAAC Operations Supervisor will make a log entry of such action taken and may request the old computed values and the new replacement values be provided. The Supervisor will ensure that the orbital change is within several seconds of the expected change and not minutes!
-

PGE Registration and Test Data Preparation

PGE Registration

The integration of science software with the DAAC production system requires that information about the Product Generation Executives (PGEs) be made known to the PDPS in its database. In addition, the PGEs themselves and the test files that they use (both input and output) need to be placed on the Data Server. These steps must be accomplished before the science software can be run and tested within the DAAC production system.

The following procedures describe how to register a new PGE with the DAAC production system. This involves updating the PDPS database with information needed to plan, schedule, and run the PGE.

The first step in the PGE registration process is to determine which ESDTs are needed for the PGE. You must verify that an ESDT metadata ODL file exists for each ESDT or generate an ODL file.

The next step in the process is to create a PGE metadata ODL file using the delivered PCF.

Finally, additional operational information (resource requirements and runtime statistics) must be input into the PDPS database. This is the last step in the PGE registration process. The order in which these procedures are done is important and should be done as indicated.

PGE ODL Preparation

This section describes how to prepare PGE ODL files. It is assumed that the SSIT Manager is running.

Prepare the PGE ODL File

- 1 From the **SSIT Manager** select **Tools** → **PDPS Database** → **PCF ODL Template** from the pull-down menu.
 - An xterm with title **SSIT: Science Metadata ODL Template Creation** is displayed.
- 2 At the program prompt **Configuration Filename (enter for default: ../../cfg/EcDpAtCreateODLTemplate.CFG)?** then press the **Enter** key for the default configuration file.
- 3 At the program prompt **ECS mode of operations?** type **MODE** then press the **Enter** key, or just press the **Enter** key if the default shown is correct.
 - **MODE** refers to the database used and will typically be OPS or TS1.

- 4 At the program prompt **Process Control file name (PCF to generate template from)?** type *PCFpathname/PCFfilename* and then press the **Enter** key.
 - The *PCFpathname* is the full path name to the location of the PCF. If not specified, the directory from which the SSIT Manager was run will be assumed.
 - The *PCFfilename* is the file name of the PCF.
- 5 At the program prompt **PGE name (max 10 characters)?** type *PGEname* and then press the **Enter** key.
 - The *PGEname* is the name of the PGE that will be registered.
- 6 At the program prompt **PGE version (max 10 characters)?** type *PGEversion* then press the **Enter** key or just press the **Enter** key if the default shown is correct.
 - The *PGEversion* is the version of the PGE that will be registered.
- 7 At the prompt **PGE Profile ID (0 for Null, max 999)?** type **1** or any valid profile ID.
 - After a brief time, the message “Successfully created ODL template file” should be displayed if the task was successful.
 - The program will output a file with the filename *PGE_PGEname#PGEversion#ProfileID.tpl*.
 - For example, if the PGE name was PGE35, and the version and profile ID were both 1, this output file will be named *PGE_PGE35#001#01.tpl*.
 - GUI will ask for a full path to place template file.
- 8 At the program prompt **Directory for output template file (including full path)?** type **ODLtplPathname** then press the **Enter** key.
 - The **ODLtplPathname** is the full path to the directory for the output template file, e.g., */usr/ecs/TS1/CUSTOM/data/DPS*.
- 9 At the program prompt **Hit Enter to run again, ‘q <Enter>’ to quit:** press the **Enter** key to repeat process with another PCF or type **q** and press **Enter** to quit.
 - The xterm will disappear.
- 10 At a UNIX prompt on an AIT Sun, type **cd ODLtplPathname** and then press the **Enter** key.
 - **ODLtplPathname** is the full path to the directory from which the SSIT Manager was run, for example */usr/ecs/TS1/CUSTOM/bin/DPS*. This will be the directory where the file *PGE_PGEname#PGEversion#ProfileID.tpl* will reside.

- 11 At a UNIX prompt on the AIT Sun, type **cp PGE_PGEname#PGEversion#ProfileID.tpl PGE_PGEname#PGEversion#ProfileID.odl** and then press the **Enter** key.
- The PGE_PGEname#PGEversion#ProfileID.tpl is the file name of the ODL template file created in Step 7.
 - The PGE_PGEname#PGEversion#ProfileID.odl is the file name of a copy which can be safely edited. This file-name convention must be used.
- 12 At a UNIX prompt on the AIT Sun, type **mv PGE_PGEname#PGEversion#ProfileID.odl /usr/ecs/MODE/CUSTOM/data/DPS/ODL**
- This will place the ODL file in the directory from which the executable that populates the PDPS database will read. **PGE_PGEname#PGEversion#ProfileID.odl** is the file name of the copy created in Step 11.
- 13 At a UNIX prompt on the AIT Sun, change the directory to the one in step above and type **vi PGE_PGEname#PGEversion#ProfileID.odl** and then press the **Enter** key.
- The **PGE_PGEname#PGEversion#ProfileID.odl** is the file name of the copy created in Step 11.
 - Any text editor may be used such as *emacs*. For example, type **emacs PGE_PGE35#001#01.odl** and then press the **Enter** key.
- 14 In the file, add required metadata to the ODL template.
- For an explanation of what metadata is required, see file */usr/ecs/MODE/CUSTOM/data/DPS/PGE_ODL.template*.
 - Note that the ShortNames typed into this file must each have a corresponding PDPS ESDT metadata ODL file.
 - All objects corresponding to output ESDTs will automatically have the SCIENCE_GROUP and YIELD set during the generation of PGE ODL.
 - All objects corresponding to output ESDTs will have an attribute "ASSOCIATED_MCF_ID. Place here the Logical Unit Number (LUN) listed in the PCF for the associated MCF listing.
 - All objects corresponding to static input ESDTs must have the SCIENCE_GROUP set. Objects corresponding to *dynamic* input ESDTs should NOT have the SCIENCE_GROUP set.
- 15 Save the changes made to the ODL template file and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq** and then press the **Enter** key.

- For other editors, refer to that editor's documentation.
-

ESDT ODL Preparation

Assumption:

- The PGE ODL file has been created and edited for the required PGE.

Follow the steps below to prepare ESDT ODL files for each ESDT required by the PGE.

Prepare the ESDT ODL

- 1 Determine ShortName for required ESDTs corresponding to a Logical Unit Number (LUN) in the PGE ODL file.
- 2 At a UNIX prompt on an AIT Sun, type **ls /usr/ecs/*MODE*/CUSTOM/data/DPS/ODL/ESDT_*ShortName*#*Version*.odl** and then press the **Enter** key.
 - The **ESDT_*ShortName*#*Version*.odl** is the file name of the ESDT ODL file you are looking for where *ShortName* is the ESDT's ShortName and *Version* is the ESDT version.
 - If a file for the desired ESDT is listed, then it has already been prepared and this procedure can be exited now.
 - For example, if the desired ESDT has the ShortName MOD03 and version 001, type **ls /usr/ecs/TS1/CUSTOM/data/DPS/ODL/ESDT_MOD03#001.odl** then press the **Enter** key.
 - If the desired file is *not* listed, continue on to Step 3.
- 3 At a UNIX prompt on the AIT Sun, type **cd *WorkingPathname*** and then press the **Enter** key.
 - The *WorkingPathname* is the full path name to a working directory for which the user has write permissions.
 - For example, **cd /home/jdoe/working/** and then press the **Enter** key.
- 4 At a UNIX prompt on the AIT Sun, type **cp /usr/ecs/*MODE*/CUSTOM/data/DPS/ESDT_ODL.template ESDT_*ShortName*#*Version*.odl** and then press the **Enter** key.
 - For *MODE* enter the mode you are working in, for example **OPS** or **TS1**.
 - The **ESDT_*ShortName*#*Version*.odl** is the file name of the ESDT ODL file to be created.

- This command copies a template ESDT ODL file to the ESDT ODL file to be created. The template is well commented.
 - For example, type **cp /usr/ecs/MODE/CUSTOM/data/DPS/ESDT_ODL.template ESDT_MOD03#001.odl** and then press the **Enter** key.
 - The **ESDT_ShortName#Version.odl** file-naming convention *must* be observed.
- 5** At a UNIX prompt on the AIT Sun, type **vi ESDT_ShortName#Version.odl** and then press the **Enter** key.
- The **ESDT_ShortName#Version.odl** represents the file name of the ESDT ODL template file created in Step 4.
 - Any text editor may be used such as *emacs*. For example, type **emacs ESDT_MOD03#001.odl** and then press the **Enter** key.
- 6** In the file, add required metadata to the ODL template.
- Use the internal documentation contained in the ODL file (from the original template) to aid in populating with metadata.
 - Note that the ShortName specified within the file must match the ShortName of the file name itself.
 - In addition, the ShortNames used in the PDPS PGE metadata ODL file must match the ShortNames in these files.
- 7** Save the changes made to the ESDT metadata ODL file and exit the editor.
- The specifics depend upon which editor is being used. If using **vi**, the command sequence to enter is **:wq** and then press the **Enter** key.
 - For other editors, refer to that editor's documentation.
- 8** Next type **mv ESDT_ShortName#Version.odl /usr/ecs/MODE/CUSTOM/data/DPS/ODL** then press the **Enter** key.
- This will place the just created ESDT ODL file in the directory from which PDPS will read it.
- 9** Repeat Steps 1 through 8 for each ESDT required by a particular PGE.
- 10** When all ESDT metadata ODL files have been completed, continue on to next section.
-

Update the PDPS Database with Science Metadata

- 1 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
 - 2 Type **setenv DISPLAY *ipaddress*:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - 3 Perform a remote login by typing **ssh *hostname*** then press the **Enter** key.
 - *hostname* refers to the (AITTL/DPS) x0ais01 or a machine that matches the SSIT Manager host.
 - 4 Type **cd /usr/ecs/*MODE*/CUSTOM/data/DPS/ODL** then press the **Enter** key.
 - Change to the directory that contains the PDPS ESDT metadata ODL files.
-

Update the PDPS Database with PGE/ESDT Metadata

- 1 From the **SSIT Manager** select **Tools → PDPS Database → SSIT Science Metadata Update** from the pull-down menu.
 - An xterm with title **SSIT: Science Metadata Database Update** is displayed.
- 2 At the program prompt **Configuration Filename (enter for default: *.././cfg/EcDpAtRegisterPGE.CFG*)?** then press the **Enter** key.
- 3 At the program prompt **ECS mode of operation?** type ***MODE*** and press **Enter** or just press **Enter** if the default shown is correct.
 - ***MODE*** refers to the database used and will typically be OPS or TS1.
- 4 At the program prompt **PGE name (max 10 characters)?** type ***PGEname*** and then press the **Enter** key.
 - The ***PGEname*** is the name of the PGE that will be registered. This name must match the PGE name specified.

- 5 At the program prompt **PGE version (max 10 characters1)?** type *PGEversion* and press **Enter** or just press **Enter** if the default shown is correct.
 - The *PGEversion* is the version of the PGE that will be registered. This version must match the PGE version specified.
- 6 At the program prompt **PGE Profile ID (0-999, 0 means null)?** type in a valid profile ID and press **Enter** or if already listed just press **Enter**.
 - The PDPS database will then be updated with the information contained in the file **PGE_PGEname#PGEversion#ProfileID.odl**
- 7 At the program prompt **Hit Enter to run again, q <Enter> to quit:** press the **Enter** key to update the PDPS database with another PGE ODL metadata file or type **q** and press **Enter** to quit.
- 8 If you make a mistake entering any values, press **Enter** here; your previous entries are restored as defaults and you won't have to retype them.

NOTE: If you make mistakes while editing the PGE and ESDT ODL files, you can run the ODL checker (**Tools → PDPS Database → Check ODL**) via the SSIT manager to locate any errors.

ODL files must have been created to define the PGE to PDPS. Examples of the ODL files are under the data directory: PGE_ODL.template, ESDT_ODL.template, ORBIT_ODL.template, TILE_ODL.template and PATHMAP_ODL. A tool can be run to generate a template ODL file for the PGE from the SSIT Manager via **Tools → PDPS Database → PCF Odl Template** script. This then has to be populated with all information that cannot be garnered from the PCF. The CheckOdl tool from the SSIT Manager via **Tools → PDPS Database → Check ODL** can be used to flag any errors in ODL before trying to put it in the database.

Perform SSIT Metadata Update (Alternate Procedure)

- 1 Source the .buildrc file for the mode in which you are working (e.g., **source .buildrc**).
 - Note that this only has to be done once per login.
- 2 Type **cd /usr/ecs/MODE/CUSTOM/bin/DPS** then press the **Enter** key.
- 3 Execute **EcDpAtDefinePGE**.
 - Shell script prompts user for information.
- 4 Enter in the location of the configuration file (**../../cfg/EcDpAtRegisterPGE.CFG**).
- 5 Enter the **MODE** of operation.
- 6 Enter name of PGE (it must match what is in the PGE ODL file).

- 7 Enter the version of the PGE (it must match what is in the PGE ODL file).
 - 8 Enter the Profile ID (it must match what is in the PGE ODL file).
 - The ODL file for the PGE must have the format: PGE_<PGE NAME>#<PGE VERSION>#<PROFILE ID>.
 - Each ODL file is displayed as it is processed.
 - A good status message should be displayed as a result.
 - Information about the PGE (inputs and outputs, Production Rules, etc) should be entered in the database.
-

SSIT Operational Metadata Update GUI

The SSIT version of the PDPS database is initialized and updated with SSIT Operational Metadata so that the Planning and Processing Subsystem can schedule and run PGEs. Here, PDPS Operational Metadata refers to PGE information that is supplied to the DAAC/SSIT Operator and may change frequently.

The operator enters this data directly into the SSIT Operational Metadata Update GUI (Figure 20). The program then writes the data directly to the SSIT version of the PDPS database. The SSIT Operational Metadata Update GUI is used to view or update the following operational parameters for a particular PGE:

- Performance parameters for the PGEs.
- Resource parameters for the PGEs.
- PGE user-defined static parameter.
- View the PGE science metadata file.

Operational Metadata

The SSIT version of the PDPS database is initialized and updated with SSIT Operational Metadata so that the Planning and Processing Subsystem can schedule and run PGEs. Here, PDPS Operational Metadata refers to PGE information that is supplied to the DAAC/SSIT Operator and may change frequently.

The operator enters this data directly into the SSIT Operational Metadata Update GUI. The program then writes the data directly to the SSIT version of the PDPS database.

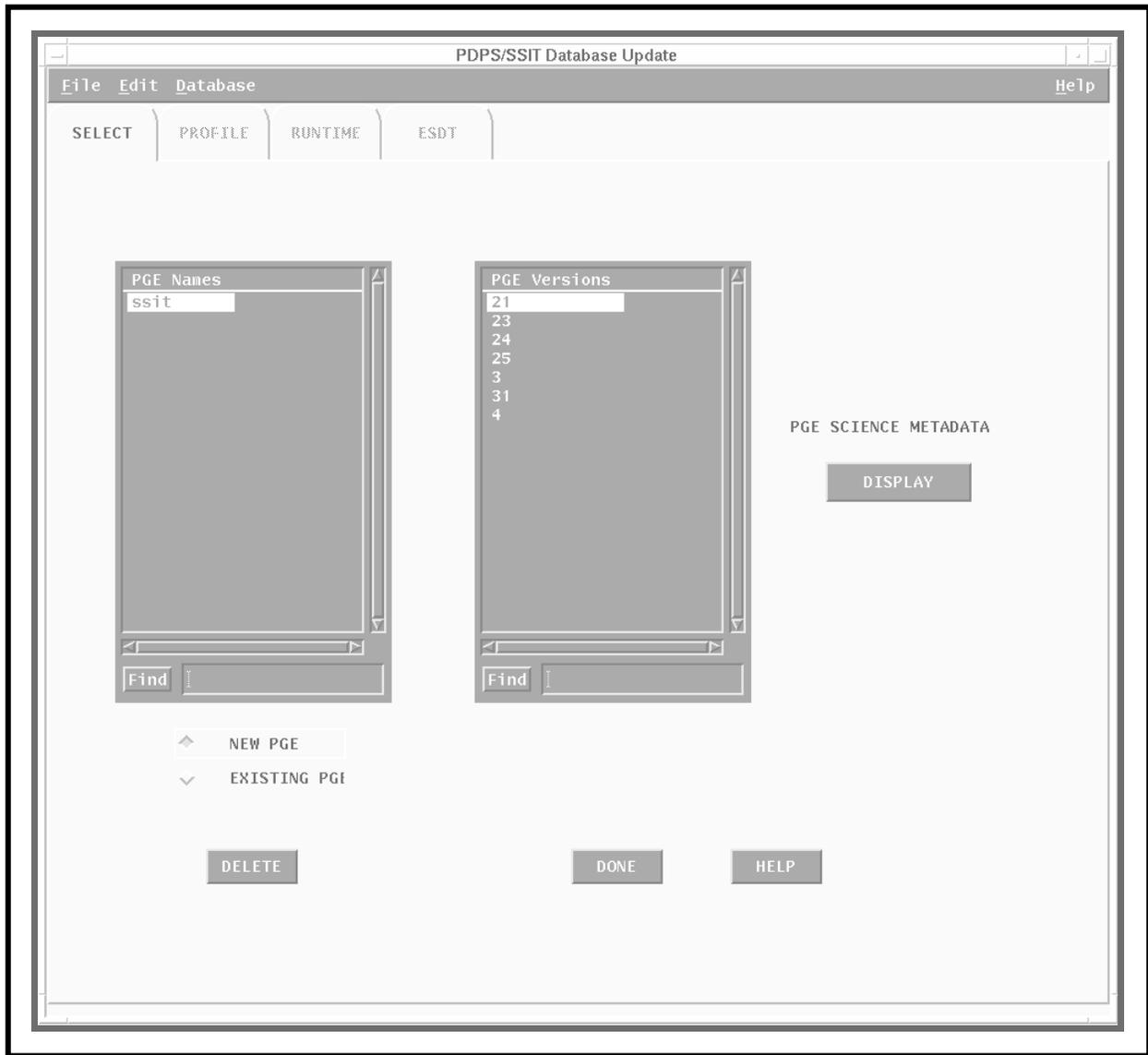


Figure 20. SSIT Database Operational Metadata Update GUI – SELECT View

Before running the SSIT Operational Metadata Update from the SSIT Manager, you must first update the PDPS with SSIT Science Metadata. In addition, to get initial PGE Performance data that will be entered into the GUI, you need to run the profiling utility, EcDpPrRusage on the PGE or have the information on profiling provided.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The required UNIX environment variables have been set.
- The Science metadata has been updated to the PDPS database for this PGE.

Update the SSIT Version of the PDPS database with Operational Metadata

- 1** From the **SSIT Manager** select **Tools** → **PDPS Database** → **SSIT Opnl Metadata Update** from the pull-down menu.
 - The PDPS/SSIT Database Update GUI "**DpAtOpDbGui**" will be displayed.
- 2** Click on the radio button labeled **NEW PGE** in the lower left quadrant.
 - The PGE that you are working on should appear in the subwindow labeled **PGE Names** along with its version number in the subwindow labeled **PGE Versions**.
- 3** In the subwindow labeled **PGE Names**, click on a PGE name. Then in the subwindow labeled **PGE Versions**, click on the PGE version for that PGE. Then click on the button labeled **EDIT**.
 - The PGE name and version will be highlighted when you click on them.
 - The page tabs **PROFILE**, **RUNTIME**, and **ESDT** will change from gray (indicating disabled) to black (indicating enabled).
 - To see the contents of PGE Metadata, click on the button labeled **DISPLAY** and then click on the button labeled **DONE**.
 - If the PGE name and/or version do (does) not appear in the lists, it means that updating of the PDPS database with PGE metadata was not successful.
- 4** Click on the **PROFILE** page tab.
 - The Profile page will be displayed.
- 5** In the fields under the label **Performance Statistics**, enter the information specified.
 - In the field labeled **Wall clock time** enter the amount of wall clock time it takes for one execution of the PGE, in seconds. The tab **PROFILE** will change from black (indicating enabled) to red (indicating database needs to be updated by **APPLY** button).
 - In the field labeled **CPU time (user)**, enter the so-called *user* time of the PGE, in seconds. This value should come from profiling the PGE.
 - In the field labeled **Max memory used**, enter the maximum amount of memory used by the PGE, in megabytes (MB). This value should come from profiling the PGE.

- In the fields labeled **Block input ops** and **Block output ops** enter the integer number of block inputs and block outputs, respectively. These values should come from profiling the PGE.
 - In the field labeled **Swaps**, enter the integer number of page swaps from the PGE. This value should come from profiling the PGE.
 - In the field labeled **Page faults**, enter the integer number of page faults from the PGE. This value should come from profiling the PGE.
- 6** In the fields under the label **Resource Requirements**, enter the information specified.
- In the field labeled **Runtime Directory Disk Space**, enter the maximum amount of disk used by the PGE during execution, in megabytes (MB), for staging MCF files, system PCF file, Profile file and generating log files. Typically this is 20 MB.
 - Click on one of the two radio buttons labeled **Proc. String** and **Computer Name** (if not already clicked on).
 - A list of processing strings should appear in the scrollable window to the left of the two radio buttons Proc. String and Computer Name. Nominally, only one item should be listed and should be highlighted.
 - In the field labeled Number of CPUs, the number 1 should be typed.
- 7** In the field labeled **Local filename of top level shell**, type in the appropriate top-level executable file name within a science software executable package.
- 8** In the field labeled **SGI Application Binary Interface (ABI)**, click on the selection appropriate for your PGE.
- 9** Once the fields on the **PROFILE** page have been completed, click on the **APPLY** button.
- This will update the PDPS database with the information just entered. The tab **PROFILE** will change from red (indicating database needs to be updated) to black (indicating enabled).
 - An information box will be displayed, click on **Ok**.
 - To start over, click on the **RESET** button. This will clear all fields.
-

Test Data Preparation and Insertion of Data Granules

This section describes how to prepare test data for use by registered PGEs. When PGEs are first delivered to the DAAC and registered within the PDPS, they will typically be run in isolation. That is, they will be run without any PGE dependencies. For this testing to be possible, test input data granules required by the PGE need to be pre-Inserted to the Data Server.

Data granules can be *dynamic* or *static*. Dynamic data granules are those whose temporal locality differs for each instance of the granule. Examples of dynamic granules are Level 0, Level 1, and Level 2 data sets. Static data granules are those whose temporal locality is static over long periods of time. Examples of static granules are calibration files, which may only change with a new version of a PGE. For any granule to be inserted to the Data Server, a Target MCF is needed (also known as an ASCII metadata ODL file or a .met file).

In the actual production environment, a Target MCF is produced by the PGE during execution. Thus, the data granule can be inserted automatically. In isolation testing of a PGE, however, the inputs needed by this PGE will not have been inserted by a previous PGE in the chain. This Insertion must be done manually. The next two sections describe how to use the Source MCF for a dynamic data granule to create a Target MCF. and then describes how to do the insert. In this way, a dynamic data granule can be inserted to the Data Server as if a PGE had produced it.

Generating a Metadata Configuration File (Source MCF)

Detailed procedures for tasks performed by the SSI&T Operator are provided in the sections that follow.

Assumptions:

- The SSIT Manager is running.
- ESDTs are installed onto the Science Data Server.

The MCFs for the output files are to be generated only for the purpose of comparison with the delivered MCFs. This way an SSIT operator can notice any metadata mismatch between the two files and notify the instrument team and DAAC ESDT group. In actual run of the PGE, the system will create the MCFs for the output files.

Generate the Metadata Configuration File (Source MCF) for the Input and Output ESDTs

- 1** From the **SSIT Manager** select **Tools** → **Data Server** → **Get MCF** from the pull-down menu.
 - An xterm in which EcDpAtGetMCF is running will be displayed as "**SSIT: Acquire MCF**".
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **EcDpAtGetMCF.sh** and then pressing the **Enter** key.
- 2** At the program prompt **Configuration Filename (default *defaultConfigFile*)?** type **../..../cfig/ *defaultConfigFile*** then press the **Enter** key.
 - ***defaultConfigFile*** will be replaced by the full path name and file name of the default configuration file.

- 3 At the program prompt **ECS mode of operation (enter for default: defaultMode)?** type **MODE** then press **Enter** or just press **Enter** if the default shown is correct.
 - **MODE** refers to the database used and will typically be **TS1**.
 - 4 At the program prompt **ESDT Short Name?** type **ESDT ShortName** and then press the **Enter** key.
 - **ESDTShortName** is the name of the ESDT that the EcDpAtGetMCF tool will use to generate the MCF.
 - 5 At the program prompt **ESDT Version?** type **ESDTversion** then press **Enter** or just press **Enter** if the default shown is correct.
 - The **ESDTversion** is the version of the **ESDT**.
 - 6 At the program prompt **Directory to receive MCF (must be full path)?** type **MCFpathname** and then press the **Enter** key.
 - **MCFpathname** is the full path name to the location where the source MCF will be placed. For example, **/home/jdoe/ssit**.
 - 7 To the final prompt **Hit Enter to run again, 'q <Enter> to quit:** then press **Enter** to generate another Source MCF or type **q** and press **Enter** to quit.
 - If you make a mistake entering any values, press **Enter** here; your previous entries are restored as defaults and you won't have to retype them.
-

Creating a Target MCF (.met) for a Dynamic/Static Granule

A Target MCF file for a corresponding data granule can be created based on the information provided in the Source MCF file and the involved science software package (PGE).

In standalone or isolation testing of a PGE, the inputs it needs will not have been inserted by a previous PGE in the chain. This insertion must be done manually. A Target MCF file for a corresponding data granule is required to run a standalone PGE. This way a dynamic data granule can be inserted to the Science Data Server as if a PGE had produced it.

The following steps can be used to obtain .met files for standalone PGE runs.

For all dynamic data granules, try to locate a .met file from an output of a previous PGE. Usually the input to a PGE is the output of some previous PGE. Hence, one can use the relevant documents from the instrument team to obtain such information on the required PGE. Once a .met file is available, all you need to do is edit the timestamp for your run of the PGE.

Use HDF_EOS-view from the SSIT manager to take a look at the header of the HDF data granules. The header contains information on the .met file.

Create a Metadata File for a Static Granule

- 1 At the UNIX prompt on the AIT Sun, type **cd *WorkingPathname*** then press the **Enter** key.
 - Example: **cd /usr/ecs/MODE/CUSTOM/data/DPS/ODL/**
 - The *WorkingPathname* is the full path name of the working directory containing the template metadata ODL file.
- 2 At the UNIX prompt on the AIT Sun, type **cp *StaticODLmet.tpl filename.met*** then press the **Enter** key.
 - The *StaticODLmet.tpl* is the file name of the template Target MCF.
 - The *filename.met* is the file name of the Target MCF for this static file. The file name extension must be **.met**.
 - This command will copy the template **Target MCF to *filename.met***. For example, type **cp StaticODLmet.tpl CER11T.mcf.met** then press the **Enter** key.
- 3 At a UNIX prompt on the AIT Sun, type **vi *filename.met*** then press the **Enter** key.
 - This command invokes the **vi editor** and reads in the Target MCF created above.
- 4 Edit the Target MCF with the specific information for the static data granule to be inserted. The following guidelines should be followed when editing on the template MCF:
 - The value for the ShortName object should be filled out with proper instrument name.
 - The value for the Version ID object should be filled out with the proper version number.
 - In the **INFORMATIONCONTENTCONTAINER** object enter the following:
 - The value for the **PARAMETERNAME** object of the class “1” should be filled out with the name of static data file.
 - The value for the **PARAMETERVALUE** object of the class “2” should be filled out based on the following guideline:
 - If the data granule is a coefficient file, a “C” followed by a numerical number **n** (**n=1,2,...**) will be used. Here n stands for the number of the coefficient file.
 - If the data granule is a MCF file, an “M” followed by a numerical number n (**n=1,2,...**) will be used. Here n stands for the number of the MCF file.

- 5 Save the changes made to the Target MCF (filename.met) and exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, then press the **Enter** key.
-

Inserting Static Data Granules into the Science Data Server

In order for static data files to be used both during the SSI&T and in production, this file must exist in the Data Server and be accessible by the local machine. A program called the Insert Static File can be used for inserting a static data granule into the Data Server.

The following Servers/Services must be up and operational:

- Science Data Server.
- Storage Management.

The following must have occurred between those Servers/Services:

- The ESDT of the static file must have been installed on the Science Data Server.
- The Target MCF (.met) for this data granule has been created for the Insert,

Prepare to Use SSIT functionality When Using a Script to Insert the Static Granule

NOTE: If using the SSI&T Manager, go to the next section.

- 1 Source the buildrc file for the mode in which you are working (*source .buildrc*).
 - Note that this only has to be done once per login.
 - 2 To change to the utilities directory type **cd /usr/ecs/MODE/CUSTOM/utilities** then press the **Enter** key.
 - 3 Type **cd /usr/ecs/MODE/CUSTOM/bin/DPS** then press the **Enter** key.
 - The Shell script will prompt user for information at each step of the process. See the next section, Step 4 to pick up on this process.
 - 4 Execute **EcDpAtInsertStatic**.
 - The Shell script will provide prompts for information at each step of the process. See the next section, Step 4 to pick up on this process.
-

What Must Be Done Using the SSIT Manager Tools:

Assumptions:

- The SSI&T Manager is running.

Start the Insert Static File Tool

- 1 From the **SSIT Manager** select **Tools** → **Data Server** → **Insert Static File**.
 - A GUI with title **SSIT: PGE Static Input File Insertion** is displayed.
- 2 At the program prompt **Configuration Filename? (enter for default: ../../cfg/EcDpAtInsertStaticFile.CFG)** type the *path/filename* then press the **Enter** key.
- 3 At the program prompt **mode (default ops)?** either press **Enter** to accept the default or type the *MODE* then press the **Enter** key.
- 4 Enter the **short name** of the ESDT (for the static file).
 - This value is in the PDPS database under the PIDataTypeMaster table and must be in the PGE ODL file.
- 5 At the program prompt **ESDT name?** type *ESDTShortName* then press the **Enter** key.
 - For example, type **MOD02LUT** then press the **Enter** key.
- 6 Enter the version of the ESDT for the static file.
 - The value is also in the PDPS database under the PIDataTypeMaster table and must be in the PGE ODL file.
- 7 At the program prompt **PGE version (default 1)?** type *PGEVersion* then press the **Enter** key.
 - The *PGEVersion* must match exactly the PGE version entered into the PDPS for this PGE.
- 8 Enter the science group for this static (this will be from the ODL created during Populating the PGE information in the Database).
- 9 At the program prompt **Science group for Static file (one of{C,L,D,O} followed by a 3 digit number)?** type *ScienceGroupID* then press the **Enter** key.
 - *ScienceGroupID* is an identifier used to define the file type as a coefficient file, a lookup table file, or a MCF. It distinguishes static granules of different types that share the same ESDT. For instance, for a coefficient file, use **Cn**, where number *n* could be 0, 1, 2...; this number *n* needs to be matched with the number *n* in the PGE_PGENAME#Version.odl file for an MCF. For example, type **C001** and then press the **Enter** key.
 - The **Science Group ID** must match what was edited into the PGE metadata ODL file for that PCF entry.
- 10 At the program prompt **Is there more than one data file for this Static (Y = Yes, N = No)? (enter for default: N)** if there is only one data file, press the **Enter** key and go to next step. If there is more than one data file, type **Y** then press the **Enter** key and go to Step 13.

- 11 At the program prompt **Single Static Filename to Insert (including FULL path)?** type *pathname/GranuleFileName* then press the **Enter** key.
 - *pathname/GranuleFileName* is the full path name and file name of the static data granule to be inserted. For example, type */home/MODIS/PGE10/MOD_PR28/coeff/emissivity.dat* and then press the **Enter** key.
 - 12 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)?** type *pathname/GranuleFileName.met* and then press the **Enter** key.
 - *pathname/GranuleFileName.met* is the full path name and file name of the **.met** file for the associated static data granule to be inserted. For example, type */home/MODIS/PGE10/MOD_PR28/MOD28LUT.met* then press the **Enter** key.
 - 13 At the program prompt **Directory where all data files and .met file exist (FULL path)?** type *pathname* then press the **Enter** key.
 - *pathname* is the full path of the directory where all data files and **.met** file exist.
 - Note that for a multi-file granule, the data files and **.met** file should be placed in the same working directory.
 - 14 At the program prompt **Name of MFG file (enter to end list)?** type each *GranuleFileName* one at a time and press the **Enter** key then press **Enter** again to end the list.
 - *GranuleFileName* is the name of each multi-file granule.
 - 15 At the program prompt **Associated ASCII Metadata Filename to Insert?** type *GranuleFileName.met* and then press the **Enter** key.
 - *GranuleFileName.met* is the name of one **.met** file that is used with all data granules in the event of a multi-file granule.
 - The static data granule will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.
 - 16 At the program prompt **Hit Enter to run again, 'q <Enter>' to quit:** type **q** and press **Enter** to quit or just press **Enter** to insert additional dynamic granules.
 - If continuing, repeat Steps 2 through 12.
-

Inserting Dynamic Data Granules to the Science Data Server

In order for dynamic data files to be used both during the SSI&T and in production, this file must exist in the Data Server and be accessible by the local machine. A program called the Insert Test Dynamic File can be used for inserting a dynamic data granule into the Data Server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The ESDTs have been installed on the Data Server.
- The Target MCF for this data granule has been created for the Insert.

Insert a Dynamic Granule to the Data Server

- 1** From the **SSIT Manager**, click on **Tools** → **Data Server** → **Insert Test Dynamic**.
 - An xterm with title **SSIT: PGE Test Dynamic Input File Insertion** is displayed.
- 2** At the program prompt **Configuration Filename? (enter for default: .././cfg/EcDpAtInsertTestFile.CFG)** type the *pathname/filename* then press the **Enter** key.
- 3** At the program prompt **ECS Mode of operations?** type the *MODE* then press the **Enter** key.
- 4** At the program prompt **ESDT short name for the file(s) to insert?** type *ESDTShortName* then press the **Enter** key.
 - *ESDTShortName* is the ShortName of the ESDT descriptor file corresponding to this granule to be inserted. For example, type **MOD021KM** then press the **Enter** key.
- 5** At the program prompt **ESDT Version for the file(s) to insert?** type *number* then press the **Enter** key.
 - *number* is the ESDT version number.
- 6** At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?** if there are no multi-files for the ESDT, press the **Enter** key and go to Step 7; if there is more than one file for this granule, go to Step 9.
- 7** At the program prompt **Single Filename to Insert? (including FULL path)** type *pathname/GranuleFilename* then press the **Enter** key.
 - *pathname/GranuleFilename* is the full path name and file name of the data granule to be inserted. For example, type **/home/MODIS/PGE10/MOD021KM.A1996217.0014.002.hdf** and then press the **Enter** key.
- 8** At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** type *pathname/GranuleFileName.met* then press the **Enter** key.
 - *pathname* is full name of the path and *GranuleFileName.met* is the name of the associated .met file. For example, **/home/MODIS/PGE10/MOD021KM.met**
 - The dynamic data **granule** will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.

- 9 At the program prompt **Directory where all data files and .met file exist (FULL path)?** type *pathname* then press the **Enter** key.
 - *pathname* is the full path of the directory where all data files and .met files exist.
 - Note that for a multi-file granule, the data files and .met file should be placed in the same working directory.
 - 10 At the program prompt **Name of MFG file (enter to end list)?** type each *GranuleFileName* one at a time and press the **Enter** key then press **Enter** again to end the list.
 - *GranuleFileName* is the name of each multi-file granule.
 - 11 At the program prompt **Associated ASCII Metadata Filename to Insert?** type *GranuleFileName.met* then press the **Enter** key.
 - *GranuleFileName.met* is the name of one .met file that is used with all data granules in the event of a multi-file granule.
 - The **dynamic** data granule will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.
 - 12 At the program prompt **Hit Enter to run again, 'q <Enter>' to quit:** type **q** and press **Enter** to quit or just press **Enter** to insert additional dynamic granules.
 - If continuing, repeat Steps 2 through 8.
-

Alternate Method of Inserting a Granule into the Science Data Server

The Science Data Server subsystem includes a utility that allows users to manually insert a data granule. The tool prompts the user for key inputs for inserting the granule. The following procedures describe this process.

Insert a Granule into the Science Data Server (Alternate Procedure)

- 1 Ensure that the **Science Data Server** subsystem is currently executing on the appropriate ACMHW HWCI server machine.
- 2 Start the Science Data Server test utility by entering the following at the UNIX prompt on the SDSRV workstation:

```
setenv MODE TS1  
cd /usr/ecs/MODE/CUSTOM/bin/DSS/  
source ../ ../utilities/EcCoEnvCsh
```

/usr/ecs/MODE/CUSTOM/bin/DSS/dttest6ConfigFile
/usr/ecs/MODE/CUSTOM/cfg/EcDsScienceDataServerClient.CFG *ecs_mode*
MODE

- The following selection menu is displayed:

1 INSERT granule
2 INSERT granule with browse file
3 ACQUIRE granule with date
4 ACQUIRE granule with a UR
5 DELETE granule
6 Exit

Please make selection=>

3 Choose option **1** to perform the insert.

- The program indicates that the search process will take place first by displaying the message “**Executing insert** “. The user is prompted to enter the datatype of the data that will be inserted:

Enter data type=>

4 Type *datatype* (e.g., **AST_L1BT**) then press the **Enter** key.

- The program then prompts for the full path name of the data file:

Enter datafile name (full path)=>

5 Type *pathname* (e.g., **/tmp/:SC:AST_L1BT:1391:1.EOSHDF**) then press the **Enter** key.

- The program then prompts for the full path name of the metadata file:

Enter data metafile name(full path)=>

6 Type *pathname* (e.g., **/tmp/AST_L1BT.MCF**) then press the **Enter** key.

- The program will then give status on the success of the insert. The following messages should appear on the successful insert:

Insert science data only...
Trying to make a request to {DSSDSRV} Success.

7 Press the **Enter** key.

- The program redisplay the first menu that was given in Step 2.

8 Type **6** then press the **Enter** key.

- The program exits.
-

Alternate Method of Acquiring a Granule from the Science Data Server

The Science Data Server subsystem includes a utility that will allow users to manually search and retrieve (acquire) a data granule. The tool will prompt the user for key inputs for acquiring the granule. The following procedures describe this process.

Acquire a Granule from the Science Data Server (Alternate Procedure)

1 Ensure that the **Science Data Server** subsystem is currently executing on the appropriate ACMHW HWCI server machine.

2 Start the Science Data Server test utility by entering the following at the UNIX prompt on the SDSRV workstation:

```
setenv MODE TS1
```

```
cd /usr/ecs/MODE/CUSTOM/bin/DSS/
```

```
source ../ ../utilities/EcCoEnvCsh
```

```
/usr/ecs/MODE/CUSTOM/bin/DSS/dttest6
```

```
ConfigFile /usr/ecs/MODE/CUSTOM/cfg/EcDsScienceDataServerClient.CFG  
ecs_mode MODE
```

- The following selection menu is displayed:

```
1 INSERT granule  
2 INSERT granule with browse file  
3 ACQUIRE granule with date  
4 ACQUIRE granule with a UR  
5 DELETE granule  
6 Exit
```

```
Please make selection=>
```

3 Choose option **3** to perform the search and acquire.

- The program indicates that the search process will take place first by displaying the message “**Executing search** “. The user is prompted to enter a hostname:

```
Enter hostname=>
```

4 Type *hostname* (e.g., **x0acs11**) then press the **Enter** key.

- Enter the hostname of the machine on which the Science Data Server process is executing.
- The user is prompted to enter a datatype:

```
Enter data type=>
```

5 Type *datatype* (e.g., **AST_L1BT**) then press the **Enter** key.

- Enter the ESDT short name of the type of data that is to be acquired.
- The program then prompts for a start and end date:

Enter starting date(mm/dd/yy)=>

6 Type *startdate* (e.g., **07/02/04**) then press the **Enter** key.

- The start date and end date indicate a range over which the user would like to search the database for data of the given type. The start and end dates will narrow the search to those data granules that were collected within that time range. Times are given in the format **mm/dd/yy**.
- The program then prompts for an end date:

Enter end date(mm/dd/yy)=>

7 Type *enddate* (e.g., **07/05/04**) then press the **Enter** key.

- The program makes a request of the Science Data Server and the following message is displayed:

Trying to make a request to [:DSSDSRV]

- A table of data granules is displayed if any were found within the given time range. For example, if the request were to display all of the data granules for the **ESDT AST_L1BT** within a certain time range, the following type of table would be displayed:

UR Type Create Date Size

```
-- -----  
AST_L1BT SC:AST_L1BT:1390  
AST_L1BT SC:AST_L1BT:1391  
AST_L1BT SC:AST_L1BT:1322  
AST_L1BT SC:AST_L1BT:1289  
AST_L1BT SC:AST_L1BT:1299
```

- The table displays the data type (AST_L1BT) and UR (i.e., SC:AST_L1BT:1390) for each granule found.
- In addition to the table, a list of the granules with a corresponding numerical index will be displayed with a prompt to the user to enter the index of the granule that they wish to acquire. An example of the indexed list follows:

NOW ENTERING ACQUIRE

There is(are)5 in the collection

```
Index = 0 AST_L1BT SC:AST_L1BT:1390 Index = 1 AST_L1BT  
SC:AST_L1BT:1391 Index = 2 AST_L1BT SC:AST_L1BT:1322 Index = 3  
AST_L1BT SC:AST_L1BT:1289 Index = 4 AST_L1BT SC:AST_L1BT:1299  
Please enter the index of the associated UR
```

8 Type *number* (the numerical index of the granule to acquire) and press the **Enter** key.

- The program then prompts for an media type:

Valid Media Types:

FtpPull

FtpPush

scp

8MM

CDROM

DVD

DLT

Enter media type (case sensitive)=>

9 Type *mediatype* (e.g., **FtpPush**) then press the **Enter** key.

- The media type indicates the type of medium on which the data will be retrieved.
- The program then prompts for the media format:

Enter mediaformat=>

- The media format should be entered as “**FILEFORMAT**”.

10 Type **FILEFORMAT** then press the **Enter** key.

- The program then prompts for the user profile ID:

Enter userProfID =>

- The user profile ID entry can be any alphanumeric character.

11 Type *userProfID* (e.g., **a**) then press the **Enter** key.

- The program then prompts for the user ID:

Enter username=>

12 Type *username* then press the **Enter** key.

- The program then prompts for the password:

Enter password=>

13 Type *password* then press the **Enter** key.

- The program then prompts for the name of the host to which the data ftp are to be sent by ftp:

Enter host=>

- 14 Type *hostname* (e.g., **x0acs11**) then press the **Enter** key.
- The program then prompts for the fully qualified destination directory to which the file will be sent by ftp:
Enter destination=>
- 15 Type *destination* (e.g., **/tmp**) then press the **Enter** key.
- The program then prompts for the fully qualified destination directory to which the file will be sent by ftp:
Enter destination=>
 - The program displays the following message:
Trying to make a request to [:DSSDSRV]
 - If the acquire operation is successful, the utility displays the following message:
**Acquire successful.
Please <CR> to continue.**
- 16 Press the **Enter** key.
- The program redisplay the first menu that was given in Step 2.
- 17 Type **6** then press the **Enter** key.
- The program exits.
-

Placing the Science Software Executable (SSEP) onto Science Data Server

In order to be able to run a PGE within the DAAC production system, the EXE TAR file has to be inserted to the Science Data Server. This tar file consists of all files needed to run a PGE, except for input data files. This includes the executables, any scripts, and the SDP Toolkit message files.

Assembling a Science Software Executable Package

This section describes how to assemble a Science Software executables Package (SSEP) and create a corresponding Target MCF. A SSEP is a UNIX tar file that contains PGE executables and SDP Toolkit message files.

In order to insert a PGE tar file into the Science Data Server, a corresponding Target MCF (.met) must be generated before insertion. Such an ASCII metadata ODL file can be obtained by editing an existing template ODL file with the information of the specific PGE. The following procedures describe how to assemble a PGE tar file and create an ASCII metadata ODL file.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- PGE executables and message files required by this PGE are available to make a SSEP.

Create an SSEP

- 1** At the UNIX prompt on an AIT Sun, type **mkdir *SSEPpathname*** and then press the **Enter** key.
 - The *SSEPpathname* is the full path name of a *new* directory that will contain all the files to be placed into the SSEP as well as the SSEP itself.
 - It is recommended that *SSEPpathame* be named with a convention that indicates the PGE for which a SSEP will be created. For example, type **mkdir PGE35.ssep** and then press the **Enter** key.
- 2** At the UNIX prompt on the AIT Sun, type **cd *SSEPpathname*** and then press the **Enter** key.
 - The *SSEPpathname* is the directory name of the new directory created in Step 1.
- 3** At the UNIX prompt on the AIT Sun, type **cp *pathname/file1 pathname/file2 ... pathname/filen .*** then press the **Enter** key (note the space and then “dot” at the end of the command).
 - The *pathname/file1, pathname/file2,...pathname/filen* represents a list of path names and file names (delimited by spaces) to copy into the current directory, *SSEPpathame* (the “dot” represents the current directory and must be last in the command).
 - For example, type **cp /data/MODIS/pge/PGE35.exe /data/MODIS/MOD_13453 .** then press the **Enter** key (note the space and then “dot” at the end of the command).
 - The files copied into this directory should be the PGE executable, any shell scripts or other executables that are part of the PGE and SDP Toolkit message files.
 - Files can be individually copied into the SSEPpathname directory. For example, type **cp /data/MODIS/pge/PGE35.exe .** then press the **Enter** key (note the space and then “dot” at the end of the command). Repeat for each file needed in the SSEP for this PGE.

- 4 At the UNIX prompt on the AIT Sun, type **tar cvf *SSEPfilename.tar* *** and then press the **Enter** key.
- To view the SSEP tar file type **tar tvf *SSEPfilename.tar*** and then press the **Enter** key. The ***SSEPfilename.tar*** is the file name for the SSEP tar file. The file name extension **.tar** is recommended but not required.
 - The asterisk (*) is a file name wildcard that represents all files in the current directory.
 - This will place all files in the SSEP tar file.
 - Do not apply compression (e.g. UNIX compress or gzip) to the tar file.
- 5 At the UNIX prompt on the AIT Sun, type **cp *filename.met.tpl filename.met*** and then press the **Enter** key.
- The ***filename.met.tpl*** is the file name of the template Target MCF for this SSEP.
 - The ***filename.met*** is the file name of the Target MCF to be tailored for this SSEP.
- 6 At the UNIX prompt on the AIT Sun, type **vi *filename.met*** and then press the **Enter** key.
- The ***filename.met*** is the Target MCF for this SSEP.
 - This command invokes the vi editor. Edit the filename.met with the specific information for the SSEP to be inserted.
 - The following guidelines should be followed when editing on the Target MCF (***filename.met***):
 - The value for the VERSIONID object should be filled out with the proper PGE version. For example: “1”.
 - In the INFORMATIONCONTENTCONTAINER object:
 - The value for the PARAMETERNAME object of the class “1” should be filled out with the PGE name. For example: “BTS”.
 - The value for the PARAMETERNAME object of the class “2” should be filled out with the PGE Science Software Version. For example: “1”.
 - The value for the PARAMETERNAME object of the class “3” should be filled out with the Platform Name. For example: “IRIX”.
 - The value for the PARAMETERNAME object of the class “4” should be filled out with the Platform Version. For example: “6.2”.
 - The value for the PARAMETERNAME object of the class “5” should be filled out with the date to perform the Insertion. For example: “970319”.
 - The value for the PARAMETERNAME object of the class “6” should be filled out with the time to perform the Insertion. For example: “14:45:00”.

- 7 Save the changes made to the SSEP's Target MCF (filename.met) and exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to exit is to type **:wq** and then press the **Enter** key.
 - For other editors, refer to that editor's documentation
-

Inserting the Science Software Executable Package onto Science Data Server

Science software, like any other data that are managed in the system, must be placed on the Science Data Server. A program called the Insert EXE TAR Tool can be used for inserting a Science Software Executable Package into the Data Server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The ESDT called PGEEEXE has been installed on the Science Data Server.
- A Target MCF (.met) for this PGEEEXE tar file has been created for the Insert.
- The PGEEEXE tar file has been created.
- The following servers/services must be up and operational:
 - Science Data Server.
 - Storage Management.

Insert the SSEP to the Science Data Server

- 1 From the **SSIT Manager**, click on **Tools** → **Data Server** → **Insert EXE TAR**.
 - An xterm with title **SSIT: PGE Executable Tar File Insertion** is displayed.
- 2 At the program prompt **Configuration Filename? (enter for default: ../../cfg/EcDpAtInsertExeTarFile.CFG)** type the *path/filename* then press the **Enter** key.
- 3 At the program prompt **ECS mode of operations?** type *MODE* then press the **Enter** key.
 - *MODE* may either be **OPS** or **TS1**.
- 4 At the program prompt **Name of PGE?** type *PGENAME* and then press the **Enter** key.
 - *PGENAME* is the name of the PGE for which this static granule is being inserted. For example, type **PGE01** and then press the **Enter** key.
 - The *PGENAME* must match exactly the PGE name entered into the PDPS for this PGE.

- 5 At the program prompt **Science software version of PGE?** type *SSWversion* and then press the **Enter** key.
 - *SSWversion* is the version of the science software that is being inserted in this SSEP.
 - 6 At the program prompt **Staged filename to insert (including Full path)?** type *pathname/SSEPFileName* then press the **Enter** key.
 - *pathname/SSEPFileName* is the full path name and file name of the SSEP tar file to be inserted. For example, type */data/MOD35/ssep/PGE35_1.tar* and then press the **Enter** key.
 - The SSEP tar file must not be compressed (e.g. with UNIX compress or gzip).
 - 7 At the program prompt **Associated ASCII metadata filename to insert (including Full Path)?** type *pathname/SSEPFileName.met* and then press the **Enter** key.
 - The default is the file name of the granule to insert with the .met file name extension. If the default is not correct, then the file name of this file must be entered.
 - 8 At the program prompt **Top level shell filename within tar file?** type *ExecFileName* and then press the **Enter** key.
 - The ExecFileName is the file name of the top-level executable or script within the SSEP tar file. It should be the same as was entered into the **PDPS/SSIT Database Update GUI**.
 - The SSEP will be inserted to the Science Data Server.
 - 9 At the program prompt **Hit Enter to run again, 'q <Enter>' to quit:** type **q** and press **Enter** to quit or just press **Enter** to insert additional dynamic granules.
 - If continuing, repeat Steps 3 through 8.
-

PGE Planning, Processing, and Product Retrieval

Production Planning and Processing

Production planning and processing activities are essential elements of SSI&T. Procedures for managing production requests, production plans, and production processing are described in detail in 625-EMD-006, *Training Material for the EMD Project Volume 6: Production Planning and Processing*, and have not been duplicated in this lesson. Refer to the Production Planning and Processing lesson for further information.

This page intentionally left blank.

Postprocessing and General Investigation

An important part of SSI&T is verifying that the output files produced at the DAAC are identical (within particular tolerances) to the test output files delivered with the DAPs. A successful comparison is a strong indication that the porting of the science software from the development facility at the SCF to the operational facility at the DAAC has not introduced any errors.

A number of file comparison tools are available during SSI&T via the SSIT Manager GUI or they can be invoked from the UNIX command line. Two tools are available for comparing HDF or HDF-EOS files, one tool for comparing ASCII files, and another tool for assisting in comparing binary files.

It is assumed that the Instrument Team has delivered test output files (produced at their SCF) with which to perform the comparison.

Examining PGE Log Files

The following three log files are produced by PGEs during runtime:

- Status log.
- User Log.
- Report log.

The log files are written by the SDP Toolkit and by the science software using the Toolkit's Status Message Facility (SMF). The location of these log files is specified in the Process Control File (PCF). When the PGE is built and run with the SCF version of the SDP Toolkit, the location and file names of the log files can be set as desired. When the PGE is built with the DAAC version of the SDP Toolkit and run within the PDPS, the location and file names of the log files is set by the system in the instantiated PCF.

The Status log file captures all error and status information. The User log file captures a subset of messages that are more informational. The Report log file captures arbitrary message strings sent by the PGE.

The aforementioned section describes how to examine log files produced by PGEs that have been built with the SCF version of the SDP Toolkit and run from the command line.

The aforementioned section describes how to examine log files (within the Production History) produced by PGEs that have been built with the DAAC version of the SDP Toolkit and run within the PDPS.

Examining PDPS-Related Scripts and Message Files

This section describes how users may access files, in addition to the PGE-produced log files, which are created during the execution of a DPR job and which may hold information useful in tracing processing problems.

Some of these files are written by default to directory paths that can only be accessed on either the SGI processor machine or one of the Sun workstations. More detailed descriptions of these files and the conditions under which they are generated will be supplied in future Green Book versions.

Mechanisms for Capturing Information about a PGE's Execution

This section contains a description of two tar files that are generated by the Planning and Data Processing Subsystems (PDPS):

- Production History (PH).
- Failed PGE Tar File.

One of the file types is generated for each execution of a PGE that is run within the DAAC production environment. The particular file type that is generated depends on the success or failure of the PGE execution. The DataGranule metadata contains a reference to the Production History.

Production History (PH)

For each successful execution of an instance of a PGE (represented by a data processing request - dpr), the processing history is captured in the production history tar file. The production history tar file is generated and archived (inserted into the Data Server) upon PGE completion. The PH is a UNIX tar file. The PH contains multiple files. Each PH can be uniquely retrieved from the Data Server. A science user has the option to acquire the PH when ordering a science granule. The name of the tar file will include a UR for the PH.

The tar file can be untarred and contains numerous component files. After un-tarring the PH, the file names of the components provide the traceability to a particular dpr (data processing request or job) that ran. Please note that the dprid is a concatenation of an abbreviated form of the PGE name (e.g., for MODIS PGE01, it would be MoPGE01#version) with a date/time stamp that is the start of the processing time for the dpr. The dprid string may have some trailing 0's filled in.

A PHcomponentFile can be examined to identify the components within the PH.

Other component files are:

- **PH log file or processing log:** Named PGEname#versionMMDDhhmm.Log – contains the DPR ID, the actual command used to run the PGE (.in), resource usage information, and the PGE exit status. It also contains a listing of output products generated, their file paths and file sizes.

- **PCF file:** Named PGName#versionMMDDhhmm.Pcf – contains the actual instantiated PCF used when running the instance of the PGE (this dpr). Note that the PCF contains URs for all inputs to this execution of the PGE. There is one UR for each input granule. If a granule is a multi-file granule, the same UR will appear (repeat itself) for each inputs file of the granule.
- **Production Log file:** Named PGName#versionMMDDhhmm.ProdLog – Contains the DPR ID, the PGEID, and resource usage information (same as in the .Log file). Resource usage information includes:
 - CPU time in application.
 - CPU time in system.
 - Physical memory.
 - Max. resident set size.
 - Avg. shared text size.
 - Avg. shared data size.
 - Avg. shared stack size.
 - Page reclaims.
 - Page faults.
 - Swaps.
 - Block input ops.
 - Block output ops.
 - Messages sent.
 - Messages received.
 - Signals received.
 - Voluntary context switches.
 - Involuntary context switches.
- **Profile file:** Named PGName#versionMMDDhhmm.Profile – Contains the environment variables defined during the execution of the PGE including the contents of the PATH environment variable.
- **The SDP Toolkit log files:**
 - **Report log file:** Named PGName#versionMMDDhhmm.TkReport – This is the same log file that the SDP Toolkit generates when the PGE runs outside of the DAAC production environment. (See SDP Toolkit documentation.)

- **Status log file:** Named PGEname#versionMMDDhhmm.TkStatus – This is the same status log file that the SDP Toolkit generates when the PGE runs outside of the DAAC production environment (See SDP Toolkit documentation.)
- **User log file:** Named PGEname#versionMMDDhhmm.TkUser – This is the same user log file that the SDP Toolkit generates when the PGE runs outside of the DAAC production environment (See SDP Toolkit documentation.)

Failed PGE Tar File

For each unsuccessful execution of an instance of a PGE (represented by a data processing request or dpr), a failed PGE tar file (may also be called the History Log or HL) is created as a diagnostic tool for the science software provider/analyst. The Failed PGE tar file is generated and archived (inserted into the Data Server) upon the abnormal completion of the PGE execution. Similar to the PH, the Failed PGE tar file is a UNIX tar file. An Instrument Team user would acquire a FailedPGE tar file by interacting with the DAAC operations staff. The name of the tar file will include a UR for the FailedPGE tar file.

The Failed PGE tar file contains numerous component files that can be examined after being untarred. After the FailedPGE tar file has been untarred, the PHcomponentFile can be examined to identify the components within. The component files are the same as those in the PH. In addition, if the PGE's execution ended with a core dump, the core file is included in the tar file. Since the FailedPGE tar file results from a failed execution of a PGE, the contents will reflect the point of the failure.

Operationally, the SDSRV error logs are examined to view the PH or Failed PGE tar files that were generated.

The error log entries referencing a PH tar file have the following format:

:PH.version:dbid:1.BINARY

The **version** is the ESDT version number for the Production History ESDT, and **dbid** is the database ID, a unique identifier within SDSRV.

The error log entries referencing a FailedPGE tar file have the following format:

:LM: FAILEDPGE.version:dbid:1.BINARY

The **version** is the ESDT version number for the FailedPGE ESDT, and **dbid** is the database ID, a unique identifier within SDSRV.

Operations staff may use the dbid to access the corresponding tar file itself from the data server. The file may then be untarred.

Examining AutoSys JIL Scripts

JILxxxxxxxx is the Job Information Language (JIL) script that defines the DPR job to **AutoSys** and which must be submitted to the **AutoSys** Database before a DPR job can be run. The name

of the file created is system-generated and begins with the characters 'JIL' followed by nine characters (e.g. JILAAa0066c).

Sample file content:

```
insert_job: 5251_823122483_1
job_type: command
command: /usr/ecs/MODE/CUSTOM/data/bin/sgi/EcDpAtExecutionMain 5251_823122483_1
machine: spr1sgigsfc
std_out_file: /home/cboettch/mockpge_msfc/out/dpat_std.out
std_err_file: /home/cboettch/mockpge_msfc/out/dpat_std.err
profile: /usr/ecs/MODE/CUSTOM/data/bin/sgi/EcDpAtRunProfile.sh
```

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Examine JIL Scripts

- 1 At the UNIX prompt on an AIT Sun, type `cd JILscriptPathname` and then press the **Enter** key.
 - The *JILscriptPathname* is the full path name to the location of the **JILxxxxxxx** scripts to be examined.
 - 2 At the UNIX prompt on the AIT Sun, type `vi JILscriptFilename` and then press the **Enter** key.
 - The *JILscriptFilename* is the file name of the **JILxxxxxxx script** to be examined.
 - This brings up the file named *JILscriptFilename* in the *vi* editor.
 - Any text editor may be used such as *emacs*. For example, type `emacs JILscriptFilename` and then press the **Enter** key.
-

Examining Application Log Files (ALOG)

Most of the custom code used during SSI&T routinely produces log files. For example, the SSIT Manager produces a log file named **EcDpAtMgr.log** and the tool used to Insert SSEPs to the Data Server (`EcDpAtInsertExeTarFile.sh`) produces a log file named **EcDpAtInsertExeTarFile.log**. These files are placed in the directory in which the tool was executed. If the **SSIT Manager** is run from the user's home directory, then the log files for each of the associated tools will be found in the user's home directory. Log files are produced at the first invocation of the tools, even if no messages are written to them. During subsequent use of the tools, the associated log files will be appended.

Log files are generally named according to the following convention:

ApplicationName.log

The *ApplicationName* is replaced with the name of the tool's executable binary. For tools that are shell scripts (e.g., **.sh files**), the shell name is left out of the log file name. For example, the tool `EcDpAtInsertStaticFile.sh` produces a log file named **EcDpAtInsertStaticFile.log** and not `EcDpAtInsertStaticFile.sh.log`.

Where an **SSIT Manager** application has been run, the path will be found using:

/usr/ecs/MODE/CUSTOM/logs/

Server connectivity failures have been encountered when installing **ESDTs, MCFs** and **.met files**. The expression “bounce the servers” has been widely used in conjunction with the effort to re-install or delete files. “Bounce” means to shut down a server and then bring it back up to rid the servers of unwanted or old bindings.

For **PGE.....odl, MCFs** and **.met files**, bouncing the **SDSRV** server needs to be done after installation and reinstallation. This can be done using ECS Assistant for each server.

File Comparison and Data Visualization

File Comparison Overview

An important part of SSI&T is verifying that the output files produced at the DAAC are identical (within particular tolerances) to the test output files delivered with the DAPs. A successful comparison is a strong indication that the porting of the science software from the development facility at the SCF to the operational facility at the DAAC has not introduced any errors.

A number of file comparison tools are available during SSI&T via the SSIT Manager GUI or they can be invoked from the UNIX command line. Two tools are available for comparing HDF or HDF-EOS files, one tool for comparing ASCII files, and another tool for assisting in comparing binary files.

Using the HDF File Comparison GUI

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- Two HDF or HDF-EOS files exist with similar structures.
- The Instrument Team has delivered test output files.
- If either of the two HDF/HDF-EOS files is in the ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

Compare Two HDF or HDF-EOS Files Using the HDF File Comparison GUI

- 1** From the **SSIT Manager**, select **Tools** → **Product Examination** → **HDF** from the pull-down menu.
 - The **HDF File Comparison GUI** window is displayed.
 - 2** In the **HDF File Comparison Tool GUI** click on the **File 1** button.
 - Read the Systems Description document and the Operations Manual. Both of these or their equivalent should be in the delivery.
-

Using the hdiff HDF File Comparison Tool

The hdiff File Comparison Tool is a text-oriented tool run from the command line. It allows comparison of two HDF or HDF-EOS files

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- Two HDF or HDF-EOS files exist with similar structures.
- The instrument Team has delivered test output files.
- If either of the two HDF/HDF-EOS files is in the ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

Compare Two HDF or HDF-EOS Files Using the hdiff File Comparison Tool

- 1** From the **SSIT Manager**, select **Tools → Product Examination → File Comparison → HDF** from the pull-down menu.
 - The **HDF File Comparison Tool** window is displayed.
 - An xterm window running hdiff is displayed.
 - 2** In the xterm window at the prompt **Options? (-h for help)**, type in any desired options then press the **Enter** key.
 - To see the list of available options, type **-h** then press the **Enter** key.
 - 3** In xterm window at the prompt **1st file to compare?** type *filename1*, then press the **Enter** key.
 - *filename1* is the file name of the first of two HDF or HDF-EOS files to be compared.
 - If *filename1* is not in the current directory (the directory from which the **SSIT Manager** was run), include the full path name with the file name.
 - 4** In xterm window at the prompt **2nd file to compare?** type *filename2*, then press the **Enter** key.
 - *filename2* is the file name of the second of two HDF or HDF-EOS files to be compared.
 - If *filename2* is not in the current directory (the directory from which the **SSIT Manager** was run), include the full path name with the file name. The two files will be compared and the output will be displayed in the xterm window.
-

Using the ASCII File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. A small minority may be in ASCII (text) format. The ASCII File Comparison Tool is a front-end to *xdiff* UNIX X Window tool for comparing two ASCII files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- Two ASCII files exist and have read permissions.
- The instrument Team has delivered test output files.

If either of the two ASCII files is in the ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

Compare Two ASCII Files

- 1** From the **SSIT Manager**, select **Tools** → **Product Examination** → **File Comparison** → **ASCII** from the pull-down menu.
 - An xterm window running ***xdiff*** will be displayed.
- 2** In xterm window at the prompt **1st file to compare?** type ***filename1*** then press the **Enter** key. Select a descriptor or mcf file in the directory with the PGE.
 - ***filename1*** is the file name of the first of two ASCII files to be compared.
 - If ***filename1*** is not in the current directory (the directory from which the **SSIT Manager** was run), include the full path name with the file name.
- 3** In xterm window at the prompt **2nd file to compare?** type ***filename2*** then press the **Enter** key.
 - ***filename2*** is the file name of the second of two ASCII files to be compared.
 - If ***filename2*** is not in the current directory (the directory from which the **SSIT Manager** was run), include the full path name with the file name.
 - A window labeled ***xdiff*** is displayed.
- 4** In the window labeled ***xdiff***, view the differences between the two files displayed.
 - File ***filename1*** is displayed on the left side of the window. File ***filename2*** is displayed on the right.
 - Only sections of file in which there are differences are displayed. A “bang” character (!) at the beginning of a line indicates that a difference was found.
 - For further help on ***xdiff***, type **man *xdiff*** in an xterm window then press the **Enter** key.
 - To close the display window select **Close** from the pull-down menu at the upper left corner of the window.

- 5 In the xterm window at the prompt **Hit Enter for another diff, 'q <Enter>' to quit:** type **q** and press **Enter** to quit or just press **Enter** to perform another comparison.
-

Using the Binary File Difference Assistant

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. A small minority may be in some binary format. The Binary File Difference Assistant aids the user in constructing code that allows comparison of binary output files. Since there is an unwieldy number of possibilities for binary file formats, this tool cannot compare two binary files without some custom code written at the DAAC, hence, the “Assistant” in the name. The Binary File Difference Assistant aids the user by generating a makefile, a driver module, and a template comparison module in C, FORTRAN 77 or IDL (Interactive Data Language). The user then edits these templates to read the particular binary format in question according to an SCF-supplied format specification.

The binary file comparison will not be performed during the SSIT training lesson.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- Two Binary files exist and have read permissions.
- The instrument Team has delivered test output files.
- If either of the two Binary files is in the ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

Compare Two Binary Files

- 1 From the **SSIT Manager**, select **Tools**→ **Product Examination** → **File Comparison** → **Binary** from the pull-down menu.
 - The **Binary File Difference Assistant** tool GUI is displayed.
- 2 In the **Binary File Difference Assistant** tool GUI, click on one of the languages listed under the **Select Language** label.
 - The choices are **C**, **FORTRAN**, or **IDL**.
 - The choice of language depends largely on preference. It does not necessarily have to be the language that was used to create the files being compared.
- 3 Optionally, click on either the **Image** button or the **Structure** button located under the label **Compare Function**.
 - Clicking on the **Image** button displays a code example for comparing binary files containing images.

- Clicking on the **Structure** button displays a code example for comparing binary files containing structures or records.
 - The displayed listing is well documented and should be read.
 - The language of the code will depend on the language selection made in Step 2.
- 4** Optionally, click on either the **Image** button or the **Structure** button located under the label **Driver**.
- Clicking on the **Image** button displays a code example for a driver invoking the compare function for binary files containing images.
 - Clicking on the **Structure** button displays a code example for a driver invoking the compare function for binary files containing structures or records.
 - The displayed listing is well documented and should be read.
 - The language of the code will depend on the language selection made in Step 2.
- 5** Optionally, click on the **Help** button.
- A **Help** window is displayed.
 - To end help click on the **Dismiss** button.
 - The **Help** window may remain displayed while using the **Binary File Difference Assistant**.
- 6** Once familiar with the code examples (Steps 3 and 4), click on the **Copy** button.
- A window labeled **Enter Unique ID** will be displayed.
 - In the field labeled **Enter unique file identifier:** type *fileID* and click on the **OK** button.
 - The *fileID* will be used in the file names of the files copied over. These files will be:
- C:**
- | | |
|--|------------------|
| DaacBinDiff_ <i>fileID</i> .c | Compare function |
| DaacBinDiff_ <i>fileID</i> _driver.c | Driver |
| DaacBinDiff_ <i>fileID</i> .mak | Makefile |
- FORTRAN:**
- | | |
|--|------------------|
| DaacBinDiff_ <i>fileID</i> .f | Compare function |
| DaacBinDiff_ <i>fileID</i> _driver.f | Driver |
| DaacBinDiff_ <i>fileID</i> .mak | Makefile |

IDL:

DaacBinDiff_fileID.pro	Compare function
DaacBinDiff_fileID_driver.pro	Driver
DaacBinDiff_fileID.sh	Shell script with help document

- The files will be copied into the directory from which the SSIT Manager is being run.
- 7 Using any desired text editor, customize the files for the job at hand.
 - 8 Build the executable using the customized makefile provided (for C and FORTRAN).
 - 9 Run the program to perform the binary file comparison.
-

Data Visualization

In order to view the success of science software in producing scientifically valid data sets, the data needs to be displayed in forms that convey the most information. Data visualization enables this to be done.

There are two visualization tools provided to the DAAC: EOSView and Interactive Data Language (IDL). These tools are both accessible via the SSIT Manager. EOSView is user friendly GUI for creating two-dimensional displays from HDF-EOS objects (Grid, Swath) as well as the standard HDF objects (SDS, Vdata, Image, Text). It has additional features such as thumbnail-panning, colorization, zooming, plotting, and animation. Only some aspects of data visualization will be addressed in this training material. For further information, see the related references.

IDL is a COTS display and analysis tool widely applied in the scientific community. It is used to create two-dimensional, three dimensional (volumetric), and surface/terrain displays from binary, ASCII, and many other formats in addition to HDF.

Only a limited number of file types will be available during SSIT training.

Viewing Product Metadata with the EOSView Tool

This procedure describes how to use the EOSView tool to inspect the metadata in the HDF-EOS output file from a PGE. To view product metadata with the EOSView tool, execute the procedure steps that follow:

View Product Metadata Using EOSView

NOTE: In the event that the SSIT Manager does not support data visualization, a backup method that uses EOSView is available.

- 1 On workstation **x0ais##**, at the UNIX prompt in a terminal window, log in using your *user id* and *password*.
 - The **x** in the workstation name will be a letter designating your site: g = GSFC, m = SMC, l = LaRC, e = EDC (LP DAAC), n = NSIDC. The **##** will be an identifying two-digit number (e.g., e0ais02 indicates an AIT Workstation at the LP DAAC).
 - 2 Type **setenv DISPLAY ipaddress:0.0** then press the **Enter** key.
 - *ipaddress* is the IP address of **x0ais##** and is required when entering this command on a Sun terminal.
 - 3 Log into an Algorithm and Test Tools (AITTL) environment.
 - 4 Perform a remote login by typing **ssh hostname** then press the **Enter** key.
 - *hostname* refers to the AIT Workstation (x0ais01).
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &** and then press the **Enter** key. Then log in to the AIT Workstation using **ssh**.
 - 5 Type **cd /usr/ecs/TS1/CUSTOM/eosview** then press the **Enter** key.
 - 6 Type **EOSView** then press the **Enter** key.
 - The EOSView GUI is displayed.
 - 7 Use the **select** buttons as necessary to guide you toward the view desired.
-

Viewing Product Metadata Using the SSIT Manager

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.

View Product Metadata Using the SSIT Manager

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **EOSView** from the pull-down menu.
 - The **EOSView GUI** is displayed.

- 2 In the GUI labeled **EOSView - EOSView Main Window**, click on **File → Open** from the pull-down menu.
 - The **Filter** GUI is displayed.
 - In the subwindow labeled **Filter**, select the appropriate directory and file to open.
 - A GUI labeled **EOSView - MyOutputFile.hdf** is displayed where *MyOutputFile.hdf* is the file name of the selected file. Once displayed, a list of **HDF** objects appears in the main window. If nothing is listed, it means that no **HDF** objects were found within the file.
 - 3 In the GUI labeled **EOSView - MyOutputFile.hdf** click on the object for which metadata is to be inspected.
 - The selected object is highlighted.
 - Do not double-click on an object since this will cause a **Dimension GUI** to be displayed instead.
 - 4 In the GUI labeled **EOSView - MyOutputFile.hdf** select **Attributes → Global** from the pull-down menu.
 - The global metadata associated with the object selected is displayed in a scrollable field.
 - If instead the message “Contains no Global Attributes” appears, then the selected object contains no global metadata.
 - 5 Repeat Steps 3 and 4 for each HDF object within the selected HDF-EOS file for which metadata is to be examined.
 - 6 To close the GUI labeled **EOSView - MyOutputFile.hdf** select **File → Close** from the pull-down menu.
 - 7 To exit from the GUI labeled **EOSView - EOSView Main Window** select **File → Exit**.
-

Viewing HDF Image Objects

This procedure describes how to use the **EOSView tool** to view science Images in the HDF-EOS output file from a PGE.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
- At least one object is an HDF image (RIS8, RIS24, i.e. Browse data).

View HDF Image Objects

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **EOSView** from the pull-down menu.
 - The **EOSView** GUI is displayed.
- 2 In the GUI labeled **EOSView - MyOutputFile.hdf**, double-click on an image object for which data is to be inspected.
 - A GUI labeled **EOSView - Image Display Window - MyImageObject** is displayed where *MyImageObject* is the name of the object selected.
- 3 To apply optional colorization, in the GUI labeled **EOSView - Image Display Window - MyImageObject** click on the **Palette** menu then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
 - This selection may be repeated until the desired palette is chosen.
- 4 To apply optional zooming, in the GUI labeled **EOSView - Image Display Window - MyImageObject** click on the **Zooming** menu then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**.
- 5 To continue applying optional zooming, in the GUI labeled **EOSView - Image Display Window - MyImageObject** click on the **Zoom In** or **Zoom Out** buttons to apply the method.
- 6 To apply optional panning while zooming, in the GUI labeled **EOSView - Image Display Window - MyImageObject** select **Options** → **Pan Window** from the pull-down menu.
 - A thumbnail representation of the entire image is displayed in the subwindow labeled **Pan Window**. The portion of the zoomed image shown in the main window is the portion indicated by the hollow rectangle on the thumbnail image.
- 7 To continue applying optional panning while zooming, use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
 - The panning option (Steps 6 and 7) may be repeated as desired.
- 8 To end the session with colorization, zooming, or panning, in the GUI labeled **EOSView - Image Display Window - MyImageObject** select **File** → **Close** from the pull-down menu.
- 9 To apply optional animation, in the GUI labeled **EOSView - MyOutputFile.hdf** select **Options** → **Animated images** from the pull-down menu.
 - A GUI labeled **EOSView - Image Animation Window - MyOutputFile.hdf** is displayed.

- 10 To continue applying optional animation, in the GUI labeled **EOSView - Image Animation Window - *MyOutputFile.hdf*** select **Options** → **Mode** from the pull-down menu then select one of the modes listed: **Stop at end**, **Continuous run**, or **Bounce**.
 - 11 To end the animation session, in the GUI labeled **EOSView - Image Animation Window - *MyImageObject*** select **File** → **Close** from the pull-down menu.
-

Viewing HDF-EOS Grid Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF-EOS Grid format. These are generally the science data and not browse images.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
- At least one object is an HDF-EOS Grid.

View HDF-EOS Grid Objects

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **EOSView** from the pull-down menu.
 - The **EOSView** GUI is displayed.
- 2 In the GUI labeled **EOSView - *MyOutputFile.hdf*** double click on a grid object listed for which data is to be inspected.
 - A GUI labeled **EOSView - Grid Select** is displayed.
 - To display information on Grid Information, Projection Information, Dimensions, Attributes for the selected object click on the appropriate checkboxes.
- 3 In the GUI labeled **EOSView - Grid Select** click on the **Data Fields** checkbox.
- 4 In the GUI labeled **EOSView - Grid Select** click on the **OK** button.
 - A GUI labeled **EOSView - Grid - *GridObjectName* - Start/Stride/Edge** will be displayed where ***GridObjectName*** is replaced by the name of the grid object selected in Step 1.
- 5 Double-click on one of the data fields listed.

- 6 To start the process of displaying the data in the form of a table of values, in the GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge** click on the checkboxes for both **YDim** and **XDim**.
 - 7 To complete the process of displaying the data in the form of a table of values, in the GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge** click on the **OK** button.
 - A GUI labeled *MyDataField* is displayed where *MyDataField* is replaced by the name of the data field selected in Step 5.
 - 8 To display the data field in image form, in the GUI labeled *MyDataField* select **File → Make Image** from the pull-down menu.
 - A GUI labeled **EOSView - Swath/Grid Image** appears.
 - To apply optional colorization, zooming, or panning while zooming refer to the **View HDF Image Objects** procedure (preceding section of this lesson).
 - 9 To end the display Grid object session, in the GUI labeled **EOSView - Swath/Grid** select **File → Close** from the pull-down menu.
 - The **EOSView - Swath/Grid** GUI disappears.
-

Viewing HDF-EOS Swath Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF-EOS Swath format. These are generally the science data and not browse images.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
- At least one object is an HDF-EOS Swath.

View HDF-EOS Swath Objects

- 1 From the **SSIT Manager**, select **Tools → Product Examination → EOSView** from the pull-down menu.
 - The **EOSView** GUI is displayed.

- 2 In the GUI labeled **EOSView - MyOutputFile.hdf**, double-click on a Swath object for which data is to be inspected.
 - A GUI labeled **EOSView - Swath Select** is displayed.
 - To display information on Dimensions, Geolocation Mappings, Indexed Mappings, Geolocation Fields, Attributes for the selected Swath Object click on the corresponding checkboxes.
 - 3 In the GUI labeled **EOSView - Swath Select** click on the **Data Fields** checkbox.
 - 4 In the GUI labeled **EOSView - Swath Select** click on the **OK** button.
 - A GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge** is displayed where *SwathObjectName* will be replaced by the name of the Swath object selected in Step 2.
 - 5 In the GUI labeled **EOSView - Swath Select** double click on one of the data fields listed.
 - 6 To start the process of displaying the data field in the form of a table of values, in the GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge** click on the checkboxes for both **ScanLineTra** and **PixelsXtrac**.
 - 7 To complete the process of displaying the data in the form of a table of values, in the GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge** click on the **OK** button.
 - 8 To display the data field in image form, in the GUI labeled *MyDataField* select **File → Make Image** from the pull-down menu.
 - A GUI labeled **EOSView - Swath/Grid Image** appears.
 - To apply optional colorization, zooming, or panning while zooming refer to the **View HDF Image Objects** procedure (preceding section of this lesson).
 - 9 To end the display swath object session, in the GUI labeled **EOSView - Swath/Grid** select **File → Close** from the pull-down menu.
 - The **EOSView - Swath/Grid** GUI disappears.
-

Viewing HDF SDS Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF SDS (standard HDF science data set) format. To view an HDF SDS object with the EOSView tool, execute the procedure steps that follow:

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
- At least one object is an HDF-SDS.

View HDF SDS Objects

- 1** From the **SSIT Manager**, select **Tools** → **Product Examination** → **EOSView** from the pull-down menu.
 - The **EOSView** GUI is displayed.
 - 2** In the GUI labeled **EOSView - MyOutputFile.hdf** double-click on an SDS object for which data is to be inspected.
 - A GUI labeled **EOSView - Multi-Dimension SDS** is displayed.
 - A number of checkboxes are displayed, one for each of the dimensions in the selected SDS (there are at least two, an **X** and a **Y**).
 - 3** In the GUI labeled **EOSView - Multi-Dimension SDS** click on two of the dimension checkboxes.
 - 4** In the GUI labeled **EOSView - Multi-Dimension SDS** click on the **Table** button.
 - 5** In the GUI labeled **EOSView - Multi-Dimension SDS** double-click on one of the data fields listed.
 - A GUI labeled with the name of the SDS object selected in Step 2 is displayed.
 - 4** To display the data field in image form, in the GUI labeled **MySDS** select **File** → **Make Image** from the pull-down menu.
 - A GUI labeled **EOSView - Image Display Window - MySDS** appears.
 - To apply optional colorization, zooming, or panning while zooming refer to the **View HDF Image Objects** procedure (preceding section of this lesson).
 - 5** To end the display SDS object session, in the GUI labeled **EOSView - Image Display Window - MySDS** select **File** → **Close** from the pull-down menu.
 - The **EOSView - Image Display Window - MySDS** GUI disappears.
-

Viewing Product Data with the IDL Tool

The following procedures describe how to use the IDL (Interactive Data Language) COTS tool to inspect the data in the output file from a PGE. These procedures are geared toward binary and ASCII formats, but can be extended to other formats supported by IDL including HDF, NetCDF, and PGE. Consult the IDL references for details on these other formats.

The major activities addresses here include creating an image display, saving an image display, creating a plot display, and saving a plot display.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The output file is binary, ASCII, or one of the other IDL supported data formats.
- IDL has been properly installed and is accessible to the user.

View Product Data with the IDL Tool

- 1 From the **SSIT Manager**, select **Tools → Product Examination → IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
 - 2 Select the procedure depending upon the activity to be performed.
 - 3 To start the process of ending the IDL session, close any display windows remaining.
 - 4 To complete the process of ending the IDL session, at the IDL prompt type **quit**, then press the **Enter** key.
 - The IDL session is closed.
-

Creating an Image Display Using IDL

The following procedure describes how to use the IDL Tool to create an image display.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- The PGE output file to be examined is of an IDL-supported type/format (if in doubt, consult the IDL Reference Guide)
- IDL has been properly installed and is accessible to the user.
- For binary files, data is assumed to be 8-bit characters

Create an Image Display Using the IDL Tool - Binary Data

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
 - 2 At the IDL prompt, type **OPENR,1,'MyBinaryFilename'** then press the **Enter** key.
 - **MyBinaryFilename** is the full path name and file name of the binary data file of known dimensions to read in.
 - The single quotes (‘) must be included around the path/file name.
 - The **1** is the logical unit number.
 - 3 At the IDL prompt, type **MyImage=BYTARR(dim1, dim2)** then press the **Enter** key.
 - **MyImage** is the name to be given to the image once created.
 - **dim1** and **dim2** are the dimensions of the input data.
 - 4 At the IDL prompt, type **READU,1,MyImage** then press the **Enter** key.
 - 5 At the IDL prompt, type **TV,MyImage** then press the **Enter** key.
 - The image (**MyImage**) should be displayed.
 - 6 At the IDL prompt, type **LOADCT,3** then press the **Enter** key.
 - This command loads color table number 3. Other color tables are available
 - 7 At the IDL prompt, type **CLOSE,1** then press the **Enter** key.
 - This closes logical unit **1**.
 - Always close logical units or an error will result the next time an access is attempted.
-

Create an Image Display Using the IDL Tool - ASCII Data

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.

- 2 At the IDL prompt, type **OPENR,*I*,('MyASCIIfilename')** then press the **Enter** key.
 - *MyASCIIfilename* is the full path name and file name of the ASCII data file of known dimensions to read in.
 - The single quotes (‘) must be included around the **path/file name**.
 - The *I* is the logical unit number.
 - 3 At the IDL prompt, type **MyImage=BYTARR(dim1,dim2)**, then press the **Enter** key.
 - *MyImage* is the name to be given to the image once created.
 - *dim1* and *dim2* are the dimensions of the input data.
 - 4 At the IDL prompt, type **READF,*I*,MyImage** then press the **Enter** key.
 - 5 At the IDL prompt, type **TV,MyImage** then press the **Enter** key.
 - The image (*MyImage*) is displayed.
 - 6 At the IDL prompt, type **LOADCT,3** then press the **Enter** key.
 - This command loads color table number 3. Other color tables are available; refer to the IDL Reference Guide for more details.
 - 7 At the IDL prompt, type **CLOSE,*I***, then press the **Enter** key.
 - This closes logical unit *I*.
 - Always close logical units or an error will result the next time an access is attempted.
-

Create an Image Display Using the IDL Tool - PGM Data

- 1 From the **SSIT Manager**, select **Tools → Product Examination → IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
- 2 At the IDL prompt, type **READ_PPM,"MyPGMfilename",MyImage,r,g,b** then press the **Enter** key.
 - *MyPGMfilename* is the full path name and file name of the PGM-formatted data file.
 - The double quotes (“) must be included around the path/file name.
 - *MyImage* is the name to be given to the image created.
- 3 At the IDL prompt, type **TVLCT,r,g,b** then press the **Enter** key.
 - Note that **r,g,b** color table syntax is used for most formatted file types in IDL.

- 4 At the IDL prompt, type **TV,MyImage** then press the **Enter** key.
 - The image (*MyImage*) should be displayed.
-

Saving an Image Display Using IDL

The next procedure describes how to save an image display (once created) to either a data file or a graphic file.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- IDL is running
- The PGE output file to be examined is of an IDL-supported type/format (if in doubt, consult the IDL Reference Guide)
- For binary files, data is assumed to be 8-bit characters
- The image display is to be saved in a binary (8-bit) or ASCII (comma-delimited characters) format.

Save an Image Display Using IDL - Binary Data

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
- 2 At the IDL prompt, type **OPENW,1,('MyBinaryFilename.bin')** then press the **Enter** key.
 - *MyBinaryFilename.bin* is the full path name and file name of the binary data file to write out.
 - The single quotes (‘’) must be included around the path/file name.
 - The *1* is the logical unit number.
- 3 At the IDL prompt, type **WRITEU,1,MyImage** then press the **Enter** key.
 - *MyImage* is the name of the image to save.
- 4 At the IDL prompt, type **CLOSE,1** then press the **Enter** key.
 - This closes logical unit *1*.

- Always close logical units or an error will result the next time an access is attempted.
-

Save an Image Display Using IDL - ASCII Data

- 1 From the **SSIT Manager**, select **Tools** → **Product Examination** → **IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
 - 2 At the IDL prompt, type **OPENW,1,('MyASCIIfilename.asc')** then press the **Enter** key.
 - *MyASCIIfilename.asc* is the full path name and file name of the ASCII data file to write out.
 - The single quotes (') must be included around the path/file name.
 - The *I* is the logical unit number.
 - 3 At the IDL prompt, type **PRINTF,1,MyImage** then press the **Enter** key.
 - *MyImage* is the name of the image to save.
 - 4 At the IDL prompt, type **CLOSE,I** then press the **Enter** key.
 - This closes logical unit *I*.
 - Always close logical units or an error will result the next time an access is attempted.
-

Save an Image Display Using JPEG Data

- 1 At the IDL prompt, type **WRITE_JPEG,"MyJPEGfilename.jpg",MyImage** and then press the **Enter** key.
 - The *MyJPEGfilename.jpg* is the full path name and file name of the JPEG data file to write out.
-

Creating a Plot Display Using IDL

The procedures for creating a plot display are clearly described in the IDL manuals; some exceptions are clarified below.

Set Axis Limits for a Plot

- 1 At the IDL prompt, type **SURFACE,MyPlot,AX=70,AZ=70,xrange=[0,20],yrange=[0,20]zrange=[0,30]** then press the **Enter** key.
 - **MyPlot** is the IDL session variable (to which you have assigned some math function, program output, image, etc.).
 - **AX** sets the displayed rotation about the X-axis.
 - **AZ** sets the displayed rotation about the Z-axis.
 - The values of **xrange** set the displayed portion of the X-axis.
 - The values of **yrange** set the displayed portion of the Y-axis.
 - The values of **zrange** set the displayed portion of the Z-axis.
 - The plot is displayed on the screen.
-

Set Axis Titles for a Plot

- 1 At the IDL prompt, type **SURFACE,MyPlot,AX=70,AZ=70,xtitle='this is X',ytitle='this is Y',ztitle='this is Z'** and press the **Enter** key.
 - **MyPlot** is the IDL session variable (to which you have assigned some math function, program output, image, etc.).
 - The value of **xtitle** sets the displayed title of the X-axis.
 - The value of **ytitle** sets the displayed title of the Y-axis.
 - The value of **ztitle** sets the displayed title of the Z-axis.
 - The plot is displayed on the screen.
-

Save a Displayed Plot to a Permanent File

- 1 At the IDL prompt, type **MyPlotDisplay=SURFACE,MyPlot,AX=80,AZ=20** then press the **Enter** key.
 - **MyPlotDisplay** is the session name for the displayed plot of **MyPlot**.
 - **MyPlot** is the IDL session variable (to which you have assigned some math function, program output, image, etc.).

- 2 At the IDL prompt, type **SAVE,MyPlotDisplay,4,'MyPlotOutput.ps'** and then press the **Enter** key.
 - *MyPlotDisplay* is the session name of the plot display.
 - *MyPlotOutput.ps* is the desired name for the saved file.
 - The **SAVE** option number **4** sets the output file type to **PostScript** (ps). There are other options, of course (consult the IDL manuals).
-

Raster Mapping Fundamentals

The procedures in this section describe how to use the IDL Tool to perform raster mapping. The first procedure describes how to use the IDL Tool to perform basic raster mapping functions. These are spatial functions involving map projections, but do not include surface modeling (also called “2.5D”) or two-dimensional spectral functions.

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- IDL is running

Perform Raster Mapping (Global Data Set Image)

- 1 From the **SSIT Manager**, select **Tools → Product Examination → IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
- 2 At the IDL prompt, type **TV,MyImage** then press the **Enter** key.
 - *MyImage* is the image name of the global image data set.
 - The image (*MyImage*) should be displayed.
- 3 At the IDL prompt, type **MAP_SET,/ORTHOGRAPHIC** then press the **Enter** key.
 - IDL also supports other map projections. Refer to IDL Reference Guide.
- 4 At the IDL prompt, type **MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN)** then press the **Enter** key.
 - *MyNewImage* is the name to assign to the resulting image.
 - *MyImage* is the name of the original global image data set.

- 5 At the IDL prompt, type **TV,MyNewImage,startx,starty** then press the **Enter** key.
 - The image (*MyNewImage*) should then be displayed.
 - 6 To apply the optional overlay “Lat/Long,” at the IDL prompt type **MAP_GRID** then press the **Enter** key.
 - This overlays Lat/Long graticule onto *MyNewImage*.
 - 7 To apply the optional overlay “world coastlines,” at the IDL prompt type **MAP_CONTINENTS** then press the **Enter** key.
 - This overlays world coastlines onto *MyNewImage*.
-

The next procedure describes how to use the IDL Tool to perform raster mapping for a sub-global data set image, one having geocentric-LLR coordinates defined for subintervals of longitude and latitude (e.g., from -88 to -77 degrees East Longitude and 23 to 32 degrees North Latitude).

The following is a list of tools and/or assumptions:

- The SSIT Manager is running.
- IDL is running

Perform Raster Mapping (Sub-Global Data Set Image)

- 1 From the **SSIT Manager**, select **Tools → Product Examination → IDL** from the pull-down menu.
 - An xterm (on the AIT Sun) is displayed within which the IDL command interpreter will run.
- 2 At the IDL prompt, type **TV,MyImage** then press the **Enter** key.
 - *MyImage* is the image name of the sub-global image data set.
 - The image (*MyImage*) should be displayed.
- 3 At the IDL prompt, type **MAP_SET,/MERCATOR,LIMIT=[lat1,lon1,lat2,lon2]** then press the **Enter** key.
 - The *lat1*, *lon1*, *lat2*, and *lon2* specify the latitude and longitude intervals of the sub-global image data set.
- 4 At the IDL prompt, type **MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN.LATMIN=lat1,LATMAX=lat2,LONMIN=lon1,LONMAX=lon2)** then press the **Enter** key.
 - *MyNewImage* is the name to assign to the resulting image.

- *MyImage* is the name of the original global image data set.
 - The *lat1*, *lon1*, *lat2*, and *lon2* specify the latitude and longitude intervals of the sub-global image data set.
- 5** At the IDL prompt, type **TV,MyNewImage,startx,starty** then press the **Enter** key.
- The image (*MyNewImage*) should then be displayed.
- 6** To apply the optional overlay “Lat/Long,” at the IDL prompt type **MAP_GRID** then press the **Enter** key.
- This overlays Lat/Long graticule onto *MyNewImage*.
- 7** To apply the optional overlay “world coastlines,” at the IDL prompt type **MAP_CONTINENTS** then press the **Enter** key.
- This overlays world coastlines onto *MyNewImage*.
-

Miscellaneous

Setting Up an Environment for Direct PDPS Database Access

The PDPS database contains many tables containing information about science software running in the production system. The population of some of this information is part of standard SSI&T procedures and no special environment set-up is required for these procedures. It may be necessary, however, to gain direct access to the PDPS database from time to time and this procedure describes how to set up the required environment. Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

- The user has been given read access (at a minimum) for the PDPS database.
- The C shell or a derivative (e.g., T shell) is the current user shell.

Set Up an Environment for PDPS Database Access

1 At a UNIX prompt on an AIT Sun or on the SPR SGI, type **vi \$HOME/.cshrc** then press the **Enter** key.

- This brings up the file named `.cshrc` in the vi editor.
- Any text editor may be used such as emacs. For example, emacs `$HOME/.cshrc` and then press the Enter key.

2 In the editor add the following lines if not already there:

```
setenv SYBASE /vendor/Sybase
```

```
setenv SYBASE /vendor/Sybase
```

```
setenv SYBROOT $SYBASE/sybooks
```

```
setenv EBTRC $SYBROOT/sun5m/.ebtrc
```

```
setenv DSQUERY computer-server
```

```
set path = ($path $SYBASE $SYBASE/bin $SYBROOT/sun5m/bin)
```

- The `computer-server` is the name of the server at the local DAAC and should be known. If not known, ask your SA or DBA.
- The above lines should be entered into the `.cshrc` file, one per line as shown.

- 3 Save the changes made to the **.cshrc file** and **exit** the editor.
 - The specifics depend upon which editor is being used. If using **vi**, the command sequence to enter is **:wq** and then press the **Enter** key.
 - For other editors, refer to that editor's documentation.
 - 4 At the UNIX prompt on the AIT Sun, type **source \$HOME/.cshrc** and then press the **Enter** key.
 - This will reinitialize the setting in the **.cshrc** file.
 - The environment set up for access to the PDPS database should now be complete.
-

Examining Application Log Files

Assumptions:

- The particular application makes use of and produces log files.
- The location of the application log file is known.
- The name of the application producing the log file is known.

Examine Application Log Files

- 1 At the UNIX prompt on an AIT Sun or on the SPR SGI, type **cd *LogFilesPathname*** and then press the **Enter** key.
 - The ***LogFilesPathname*** is the full path name to the location of a particular application log file. This is typically the directory from which the SSIT Manager or other tool is run.
 - For example, type **cd \$HOME/ceres** and then press the **Enter** key.
- 2 At a UNIX prompt on an AIT Sun or on the SPR SGI, type **vi *ApplicationName.log*** then press the **Enter** key.
 - This ***ApplicationName.log*** is the file name of the log file to examine.
 - Any text editor may be used such as **emacs**. For example, type **emacs *QAMonitor.log*** and then press the **Enter** key.
- 3 When finished, exit the editor.
 - The specifics depend upon which editor is being used. If using **vi**, the command sequence to enter is **:wq** and then press the **Enter** key.

- For other editors, refer to that editor's documentation.
-

This page intentionally left blank.

Practical Exercise

Introduction

This exercise is designed to give the students practice in science software integration and test activities.

Equipment and Materials

One workstation per student.

Statement of the requirements for the exercise.

Release 7.10 Operations Tools Manual for the EMD Project, 609-EMD-001, one copy per student.

Release 7.10 Mission Operation Procedures for the EMD Project, 611-EMD-001, one copy per student.

Training Material for the EMD Project, Volume 6: Production Planning and Processing, 625-EMD-006, one copy per student.

Science Software Integration and Test

Situation: A new PGE and associated files have been developed by the Instrument Team. This exercise will require the CM Administrator and the Science Data Coordinator to prepare the PGE for installation and operation on the Production Server.

- 1 Acquire the science software as a Delivered Algorithm Package (DAP) tar file and unpack. For this training exercise, copy the tar file from the instructor-provided directory.
- 2 Import the source and code make file into ClearCase. This will require creation of a view and a VOB to place these files.
- 3 Edit the ESDT descriptor files, MCF, makefile, and all “.met” files.
- 4 Perform the initial setup of the SSIT Manager and verify that the SSIT Manager is prepared to use the SDP Toolkit, PCF files, and that all required environmental variables point to locations in the SDP Toolkit directory.
- 5 Verify that the code is in compliance with the ESDIS Standards. For this training exercise, compile the PGE code by using the compiler’s default of ANSI Standard.
- 6 Invoke the Prohibited Function Checker on the source files.

- 7 Use the PCF checker to verify that the PCFs are syntactically correct, and contain all necessary information for the PGEs to run within the DAAC production environment (if not, update the PCF with correct data and rerun the PCF Checker).
 - 8 Compile the PGE and Link with the SDP Toolkit.
 - 9 Run the PGE in an SCF type environment and generate performance statistics.
 - 10 Examine the metadata in output HDF files using EOSView.
 - 11 Examine the PGE LogStatus, Log User, and LogReport.
 - 12 Update the PDPS Database by performing the following:
 - 13 Copy the edited ESDT ODL files into the configured PDPS area.
 - 14 Run the PCF ODL Update program from the SSIT Manager and modify the output file.
 - 15 Copy the PCF ODL files into the configured PDPS area.
 - 16 Run the Science Metadata Update Program from the SSIT Manager.
 - 17 Run the PGE Operational Metadata Program from the SSIT Manager.
 - 18 Update the Data Server by performing the following:
 - 19 Run the Insert Static Program from the SSIT Manager to insert the MCF file.
 - 20 Run the Insert Test Dynamic Program from the SSIT Manager to insert the binary data granule for input into the PGE.
 - 21 Make the tar file for the Science Software Executable Package.
 - 22 Run the EXE TAR program to inform the SSIT Manager to insert the executable package in the data server.
 - 23 Submit a Production Request for the PGE.
 - 24 Invoke the Planning Workbench to plan, schedule and activate the Production Requests.
 - 25 Monitor production by using AutoSys.
 - 26 Invoke the QA Monitor to view the products under EOSView and Production History File.
 - 27 Use DX Browser by invoking command db to look at the PDPS databases.
 - 28 If there is a problem, activate DDTs and record the problem.
-

Slide Presentation

Slide Presentation Description

The following slide presentation represents the slides used by the instructor during the conduct of this lesson.

This page intentionally left blank.