

162-TD-001-008

# Science Software I&T Operational Procedures for the ECS Project (aka, The Green Book)

Technical Data

August 2001

Prepared Under Contract NAS5-60000

## RESPONSIBLE ENGINEER

John M. Corbett /s/ 8/8/01  
John M. Corbett Date  
EOSDIS Core System Project

## SUBMITTED BY

Karin Loya /s/ 8/9/01  
Karin Loya, SSI&T Manager Date  
EOSDIS Core System Project

Raytheon Company  
Upper Marlboro, Maryland

This page intentionally left blank.

# Abstract

---

The purpose of this Technical Data Paper (also referred to as the “Green Book”) is to delineate the operational procedures to accomplish the various steps that may be involved in the integration and test of Science Data Production Software (SDPS/W) with the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS). The SDPS/W integration and test (SSI&T) is performed at the Distributed Active Archive Center (DAAC) responsible for the generation of the standard products.

General information concerning preparing and delivering SDPS/W to the DAAC is found in the *Science User’s Guide and Operations Procedure Handbook for the ECS Project, Part 4: Software Developer’s Guide to Preparation, Delivery, Integration and Test with ECS* (205-CD-002-006). Each DAAC and Instrument Team (IT) combination has formulated specific agreements, understandings, or procedures that will guide their respective SSI&T activities. The procedures in this document provide detailed instructions on how to use the tools that are provided in the Release B of ECS to accomplish the steps outlined in the DAAC-IT procedures.

This page intentionally left blank.

# Table of Contents

---

## Abstract

### 1. Introduction

1.1	Purpose.....	1-1
1.2	Organization.....	1-1
1.3	Changes from the Previous Version.....	1-2
1.4	Review and Approval .....	1-3

### 2. Related Documentation

2.1	Parent Documents .....	2-1
2.2	Applicable Documents.....	2-1
2.3	Information Documents .....	2-1

### 3. General Information

3.1	SSI&T Road Map.....	3-1
3.2	How to Use the Procedures.....	3-3

### 4. Configuration Management

4.1	Creating a View in ClearCase.....	4-1
4.2	Setting a View in ClearCase .....	4-2
4.3	Entering a Single File into ClearCase .....	4-3
4.4	Entering a New Directory into ClearCase.....	4-5
4.5	Checking Out an Element from ClearCase.....	4-6
4.6	Checking a Modified Element into ClearCase.....	4-7

## **5. The ECS Assistant**

5.1	Using ECS Assistant to Start Up / Shut Down Servers .....	5-1
5.2	Using ECS Assistant to View ECS Science Data Server Database .....	5-4

## **6. The SSIT Manager**

6.1	Description of SSIT Manager Tools .....	6-1
6.2	Starting the SSIT Manager.....	6-3

## **7. DAP Insert**

7.1	Performing a DAP Insert.....	7-1
7.2	An Example of a DAP Metadata File .....	7-2
7.3	Insert Testing of Products .....	7-3

## **8. DAP Acquire**

8.1	Performing a DAP Acquire Using SSIT Manager .....	8-1
-----	---	-----

## **9. PGE Checkout**

9.1	Checking for ESDIS Standards Compliance in Fortran 77.....	9-1
9.2	Checking for ESDIS Standards Compliance in Fortran 90.....	9-2
9.3	Checking for ESDIS Standards Compliance in C and C++.....	9-3
9.4	Checking for Prohibited Functions .....	9-4
9.4.1	Checking for Prohibited Functions: GUI Version .....	9-4
9.4.2	Checking for Prohibited Functions: Command-Line Version .....	9-8
9.5	Checking Process Control Files .....	9-8
9.5.1	Checking Process Control Files: GUI Version .....	9-8
9.5.2	Checking Process Control Files: Command-Line Version .....	9-11
9.6	Extracting Prologs.....	9-11

## **10. Compiling and Linking Science Software**

10.1	Setting Up the SDP Toolkit Environment .....	10-1
10.2	Compiling Status Message Facility (SMF) Files .....	10-3
10.3	Building Science Software with the SDP Toolkit.....	10-4

## **11. Running a PGE in a Simulated SCF Environment**

11.1	Updating the Process Control File (PCF) .....	11-1
11.2	Setting Up the Environment for Running the PGE.....	11-3
11.3	Running and profiling the PGE.....	11-3

## **12. PGE Registration and Test Data Preparation**

12.1	Science Metadata .....	12-1
12.1.1	PGE ODL Preparation .....	12-1
12.1.2	ESDT ODL Preparation .....	12-2
12.1.3	Updating the PDPS Database with Science Metadata .....	12-3
12.2	Operational Metadata.....	12-4
12.3	Insertion of Data Granules .....	12-5
12.3.1	Generating a Source Metadata Configuration File (MCF) .....	12-6
12.3.2	Target MCF (.met) for a Dynamic/Static Granule .....	12-7
12.3.3	Inserting Dynamic Data Granules to the Science Data Server.....	12-7
12.3.4	Inserting Static Data Granules to the Science Data Server .....	12-8
12.4	Placing Science Software Executable onto Science Data Server.....	12-9
12.4.1	Assembling a Science Software Executable Package.....	12-9
12.4.2	Inserting Science Software Executable Package onto Science Data Server ....	12-10

## **13. PGE Planning, Processing, and Product Retrieval**

13.1	Using the Production Request Editor.....	13-1
13.1.1	Invoking the Production Request Editor.....	13-1
13.1.2	Defining a New Production Request.....	13-3
13.1.3	Listing Production Requests .....	13-4

13.1.4	Listing Data Processing Requests .....	13-6
13.2	Using the Production Planning Workbench.....	13-7
13.2.1	Using the Planning Workbench to Run One PGE .....	13-7
13.3	Monitoring Production.....	13-9
13.4	Using the Q/A Monitor .....	13-11

## **14. File Comparison**

14.1	Using the GUI HDF File Comparison Tool.....	14-1
14.2	Using the hdiff HDF File Comparison Tool .....	14-1
14.3	Using the ASCII File Comparison Tool.....	14-2
14.4	Using the Binary File Difference Assistant.....	14-2

## **15. Data Visualization**

15.1	Viewing Product Metadata with the EOSView Tool.....	15-1
15.2	Viewing Product Data with the EOSView Tool .....	15-3
15.2.1	Viewing HDF Image Objects.....	15-4
15.2.2	Viewing HDF-EOS Grid Objects .....	15-6
15.2.3	Viewing HDF-EOS Swath Objects.....	15-9
15.2.4	Viewing HDF SDS Objects .....	15-11
15.3	Viewing Product Data with the IDL Tool.....	15-13
15.3.1	Creating an Image Display Using IDL.....	15-13
15.3.2	Saving an Image Display Using IDL.....	15-17
15.3.3	Creating a Plot Display Using IDL .....	15-18
15.3.4	Saving a Plot Display Using IDL.....	15-19
15.3.5	Raster Math Fundamentals Using IDL .....	15-19
15.4	Raster Mapping Fundamentals.....	15-20

## **16. Inserting a Science Software Archive Package**

16.1	Inserting a SSAP into PDPS .....	16-2
------	----------------------------------	------

## 17. Updating a Science Software Archive Package

17.1	Updating a SSAP .....	17-1
------	-----------------------	------

## 18. ESDT Management

18.1	Inspecting ESDTs .....	18-1
18.2	Removing ESDTs .....	18-4
18.2.1	ESDT Removal Scenario 1 - Using ContributionDriverStart .....	18-5
18.2.2	ESDT Removal Scenario 2 - Using EcIoDbDeleteCollection .....	18-10
18.3	Adding ESDTs .....	18-14
18.3.1	Adding an ESDT Using the Science Data Server GUI .....	18-14
18.4	Updating ESDTs .....	18-16
18.4.1	Updating an ESDT Using the Science Data Server GUI .....	18-17
18.5	ESDT Volume Group Configuration .....	18-19
18.5.1	Modifying an ESDT's Volume Group Information .....	18-19
18.5.2	Adding an ESDT's Volume Group Information .....	18-23

## List of Figures

5.1-1.	ECS Assistant GUI .....	5-2
5.1-2.	Subsystem Manager GUI .....	5-3
5.1-3.	Task Execution Feedback GUI .....	5-4
5.2-1.	Database Login GUI .....	5-5
5.2-2.	DB Viewer GUI .....	5-6
8.1-1.	Selecting DAP Acquire from SSIT Manager .....	8-2
9.4.1-1.	Invoking the Prohibited Function Checker .....	9-6
9.4.1-2.	Starting Screen for Prohibited Function Checker .....	9-7
9.4.1-3.	File Selection Menu .....	9-7
9.4.1-4.	Selected Files .....	9-7
9.4.1-5.	Results Screen .....	9-7
9.5.1-1.	Invoking the Process Control File Checker .....	9-10

9.5.1-2. Selecting a File to Check .....	9-10
13.1.1-1. Production Request Editor (Planning) GUI.....	13-2
13.1.2-1. PR Editor GUI.....	13-4
13.1.3-1. PR List GUI.....	13-6
13.2.1-1. Planning Workbench GUI .....	13-9
16.1-1. SSAP Main Gui.....	16-3
16.1-2. Window to define new SSAP .....	16-3
16.1-3. File list tab for file manipulation in defining new SSAP .....	16-4
16.1-4. Metadata tab to enter metadata for new SSAP .....	16-5
16.1-5. The Edit Associated Window .....	16-6
17.1-1. Metadata window for updating metadata .....	17-2
18.1-1. Science Data Server Operator GUI.....	18-2
18.1-2. Viewing An ESDT Descriptor File.....	18-3
18.3.1-1. Adding an ESDT - The GUIs Involved .....	18-15
18.4.1-1. Updating An ESDT - The GUIs Involved .....	18-18
18.5.1-1. Storage Management GUI .....	18-20
18.5.1-2. Modify Volume Groups GUI.....	18-21
18.5.1-3. Modify Volume Groups GUI with entries .....	18-22
18.5.2-1. Add Volume Group GUI .....	18-24

## List of Tables

3.1-1. Road Map of SSI&T for a PGE.....	3-1
9.4.1-1. File Name Extensions Recognized.....	9-4
9.6-1. Extracting Prologs - Prolog Delimiters.....	9-12
9.6-2. Extracting Prologs - File Name Extensions.....	9-12
10.1-1. Object Types on the SGI.....	10-1
10.1-2. SDP Toolkit Versions at the DAAC.....	10-2
13.3-1. AutoSys Jobs for a DPR .....	13-10
15.3.1-1. Creating an Image Display Using IDL - Activity Checklist .....	15-14

15.3.2-1. Saving an Image Display Using IDL - Activity Checklist.....	15-17
15.4-1. Raster Mapping Fundamentals - Activity Checklist.....	15-20
18.1-1. ESDT Inspection Using Science Data Server GUI - Quick-Step Procedures.....	18-4
18.2.1-1. ESDT Removal Using ContributionDriverStart - Quick-Step Procedures.....	18-10
18.2.2-1. ESDT Removal Using EcIoDbDeleteCollection- Quick Step Procedures .....	18-13
18.3.1-1. Adding ESDTs With Science Data Server GUI .....	18-16
18.4.1-1. Updating ESDTs With Science Data Server GUI .....	18-19
18.5.1-1. Changing ESDT Volume Group Information.....	18-23
18.5.2-1. Adding ESDT Volume Group Information .....	18-25

## **Appendix A. Troubleshooting and General Investigation**

## **Appendix B. Historical Background of ECS Release B.0 (Architectural Differences from Pre-Release B Testbed)**

## **Appendix C. Examples of the Various ODL Files Used by Each Instrument Team**

## **Appendix D. List of Responsible Engineers for Each Section**

This page intentionally left blank.

# 1. Introduction

---

## 1.1 Purpose

The purpose of this Technical Data Paper (also referred to as the “Green Book”) is to delineate the operational procedures to accomplish the various steps that may be involved in the integration and test of Science Data Production Software (SDPS/W) with the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS). The SDPS/W integration and test (SSI&T) is performed at the Distributed Active Archive Centers (DAACs) responsible for the generation of the standard products.

General information concerning preparing and delivering SDPS/W to the DAAC is found in the *Science User’s Guide and Operations Procedure Handbook for the ECS Project, Part 4: Software Developer’s Guide to Preparation, Delivery, Integration and Test with ECS* (205-CD-002-006). Each DAAC and Instrument Team (IT) combination has formulated specific agreements, understandings, or procedures that will guide their respective SSI&T activities. The procedures in this document provide detailed instructions on how to use the tools that are provided in the Release B of ECS to accomplish the steps outlined in the DAAC-IT procedures.

## 1.2 Organization

Section 2 lists related documentation. Section 3 is devoted to general information concerning SSI&T including a road map for integrating a PGE into the ECS. Section 4 contains procedures related to configuration management. Section 5 covers the usage of the ECS Assistant GUI. Section 6 describes set up and use of the SSIT Manager. Section 7 gives instructions for performing a DAP insert. Section 8 gives instructions for performing a DAP acquire. Section 9 gives the procedures dealing with standards checking and compliance of the science software. Section 10 describes how to build the delivered science software and Section 11 describes how to run it in a simulated SCF environment.

Section 12 describes the procedures for integration of the science software into the ECS; it describes procedures for registering the PGE with the Planning and Data Processing System (PDPS) and for inserting files to the Science Data Server. Section 13 describes procedures for planning and running science software within the PDPS and retrieving the output. Section 14 then describes procedures for comparing the output produced to that delivered.

Section 15 contains procedures for examining output in more detail using some of the data visualization tools provided.

Section 16 covers the procedures for inserting an SSAP into Science Data Server and Section 17 covers the procedures for updating an SSAP.

Section 18 covers some of the considerations and procedures for managing ESDTs.

Appendix A contains troubleshooting and general investigation.

Appendix B overviews the relation between Pre-Release B Testbed and B.0 Architectures. Appendix C gives typical examples for PGE and ESDT ODL files for each instrument team. Finally, Appendix D lists responsible engineers for each section.

### 1.3 Changes from the Previous Version

The primary changes to this release of the "Greenbook" are to make the SSI&T procedures consistent with ECS Release 6A.

To assist the SSIT personnel in identifying changes, which are relevant to his/her work, the following major topics are highlighted:

<b>Section</b>	<b>Title</b>	<b>Change</b>
<b>in v007</b>		
<b>5. The ECS Assistant</b>		
Section 5.2	Using ECS to Perform System Monitoring	Function replaced by the Whazzup GUI. Refer to document 333-CD-600.
<b>6. DAP Insert</b>		
Section 7.4	An Example of a faked.hdf File.	Material in these sections is replaced by material included in the new Chapter 18.
Section 7.5	Mail Template	
Section 7.6	Examining the Validity of Product Metadata	
<b>9. PGE Checkout</b>		
Section 9.3	Checking for ESDIS Standards Compliance in C	Section amended to include C++ per ECSed28512
<b>10. Compiling and Linking Science Software</b>		
Section 10.1	Updating the Process Control Files (PCFs)	Section moved to section 11.1
Section 10.4	Building Science Software with the SCF Version	Consolidated with material from section 10.5
Section 10.5	Building Science Software with the DAAC	Consolidated with material from with section 10.4
<b>12. PGE Registration and Test Data Preparation</b>		
12.2	Operational Metadata (Item #6)	Information corrected per ECSed31076
<b>18. ESDT Management</b>		
		New chapter outlining EDST maintainance.
<b>Appendix C</b>		
Section C3	Typical MISR PGE & ESDT ODL	Updated to current operational scenario.

Section C4.7	Files Typical Terra MODIS PGE & ESDT ODL Files	Example of MODIS Terra version 3 PGE added.
Section C.5	Typical AIRS PGE & ESDT ODL Files	Updated to current operational scenario.
Section C.6	Typical Aqua MODIS PGE ODL File	Section added highlighting the differences between the Terra and Aqua scenarios

Some text has been revised for conciseness and readability. Quik-Step procedures are now used only in sections where a large number of steps are involved.

## 1.4 Review and Approval

This Technical Data Paper is an informal document approved at the Office Manager level. It does not require formal Government review or approval; however, it is submitted with the intent that review and comments will be forthcoming.

Questions regarding technical information contained within this paper should be addressed to the following ECS contact:

Karin Loya, SSI&T Manager  
(301) 925-1052

[kloya@eos.hitc.com](mailto:kloya@eos.hitc.com)

This page intentionally left blank.

## 2. Related Documentation

---

The following related documentation is available either in hard copy from the author or generating source or on the World Wide Web (WWW) at the URL listed. Additionally, most documents generated by the ECS project are available on the WWW via the ECS Data Handling System at <http://edhs1.gsfc.nasa.gov/>. The latest available versions should generally be referenced.

### 2.1 Parent Documents

The parent documents are the documents from which this document's scope and content are derived.

609-CD-001	Interim Release One (Ir1) Maintenance and Operations Procedures for the ECS Project
609-DR-002	Release A Operations Tools Manual for the ECS Project
162-TD-001	Science Software I&T Operational Procedures for the ECS Project (aka, the Green Book)

### 2.2 Applicable Documents

The following documents are referenced within and are directly applicable to this document:

205-CD-002	Science User's Guide and Operations Procedure Handbook for the ECS Project, Volume 4: Software Developer's Guide to Preparation, Delivery, Integration and Test with ECS
423-41-64	ESDIS Project Requirements for EOS Instrument Team Science Data Processing Systems
333-CD-600	Release 6A SDP Toolkit Users Guide
445-TP-006	EOSView Users Guide for the ECS Project
423-16-01	Data Production Software and Science Computing Facility (SCF) Standards and Guidelines
609-CD-600-001	Release 6A Operations Tools Manual

### 2.3 Information Documents

The following documents, although not referenced herein or directly applicable, do amplify or clarify the information presented in this document:

305-CD-026	Release B SDPS Planning Subsystem Design Specification for the ECS Project
------------	--

416-WP-001            Implementation Plan for the Pre-Release B Testbed for the ECS Project

X3.9-1978            ANSI FORTRAN 77 Programming Language Standard

FORCHECK for Sun/SunOS, A Fortran Verifier and Programming Aid, Installation Guide

FORCHECK for Sun/SunOS, A Fortran Verifier and Programming Aid, Users Guide

Research Systems, IDL Basics, Interactive Data Language, Version 4.0, 4/95 (ECS Library reference number 2100-591.1A)

Research System, IDL Reference Guide A-M, Volume 1, Interactive Data Language, Version 4, 3/95 (ECS Library reference number 2100-591.1C)

Research System, IDL Reference Guide N-Z, Volume 2, Interactive Data Language, Version 4, 3/95 (ECS Library reference number 2100-591.1D)

Research Systems, IDL User's Guide, Interactive Data Language, Version 4, 3/95 (ECS Library reference number 2100-591.1B)

Distributed Defect Tracking System (PureDDTS), The Automated Solution to Defect Tracking, Release 3.2, User's Manual, Pure Software, Part No.PDT320-XPX-UGD (ECS Library reference number 2100-723)

**Please note that Internet links cannot be guaranteed for accuracy or currency.**

Information on the SDP Toolkit, HDF-EOS, and EOSView at URL:

<http://newsroom.gsfc.nasa.gov/sdptoolkit/toolkit.html>

Prohibited Functions in the EOSDIS Core System at URL:

<http://ecsinfo.hitc.com/iteams/ProhibFunc/>

FORTRAN 77 in the EOSDIS Core System at URL:

<http://ecsinfo.hitc.com/iteams/Standards/F77std.html>

PGE Design Information Page at URL:

[http://ecsinfo.hitc.com/iteams/Science/pge\\_wp.html](http://ecsinfo.hitc.com/iteams/Science/pge_wp.html)

EOS Instrument Team Information Page at URL:

<http://ecsinfo.hitc.com/iteams/>

## 3. General Information

---

### 3.1 SSI&T Road Map

Science software integration and test (SSI&T) is the process by which science software is tested for production readiness in the DAACs. The goals are to assure (1) *reliability*, which means that the software runs to normal completion repeatedly over the normal range of data inputs and runtime conditions, and (2) *safety*, which means that the software executes without interfering with other software or operations.

SSI&T activities can be broadly separated into two categories: pre-SSI&T and formal SSI&T. Pre-SSI&T activities can be defined as those activities that do not involve the ECS Planning and Data Processing System (PDPS) or the Science Data Server. Formal SSI&T activities are defined as those that involve the full ECS including the PDPS and Science Data Server.

Table 3.1-1 can be used as a PGE road map for the remainder of the Green Book. It lists most SSI&T activities in the order in which they are most likely to occur. Column 1 describes the SSI&T task; column 2 lists the relevant sections in the Green Book; and column 3 represents an attempt to qualify the importance of the task. Tasks labeled as “Critical” must be done in order to have the PGE integrated into the ECS and run by the automated PDPS. Tasks labeled as “Essential”, “Valuable”, and “Optional” have lowering degrees of importance, respectively, and are somewhat subjective. This table should not be construed as overriding specific agreements between DAACs and Instrument Teams.

**Table 3.1-1. Road Map of SSI&T for a PGE (1 of 3)**

Step	Task	Sections	Importance
1	Acquire Delivered Algorithm Package (DAP).	8	Critical
2	Inspect the delivery and accompanying documentation.	N/A	Valuable
3	Place the DAP contents under configuration management.	4	Valuable
4	Check the source files for standards compliance.	9.1,9.2, 9.3	Valuable
5	Check the source files for prohibited functions.	9.4	Valuable
6	Extract and check prologs.	9.6	Valuable
7	Check the Process Control File (PCF).	9.5	Valuable

**Table 3.1-1. Road Map of SSI&T for a PGE (2 of 3)**

<b>Step</b>	<b>Task</b>	<b>Sections</b>	<b>Importance</b>
8	Revise the delivered PCF to comply with the local DAAC environment and the location of data files.	11.1	Essential
9	Compile the SDP Toolkit Status Message Facility (SMF) files.	10.2	Essential
10	Build the PGE linking it to the SCF version of the SDP Toolkit.	10.3	Essential
11	Run and profile the PGE from the command line. Save the profiling results. They will be used later when entering operational metadata into the PDPS.	11.3	Essential
12	Compare the output produced with test data delivered with the PGE.	14.1, 14.2, 14.3, 14.4	Essential
13	Examine the output products using data visualization techniques.	15	Optional
14	Verify that all required ESDTs have been successfully installed.	5.2, 18.1	Essential
15	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 <i>before</i> this step is done.	12.1.1	Critical
16	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 needed for: (1) All input and output data granules (2) The PGE executable tar file (SSEP)	12.1.2	Critical
17	Update the PDPS Database with Science Metadata	12.1.3	Critical
18	Supply operational metadata to the PDPS database.  This completes the PGE registration.	12.2	Critical
19	Build the PGE linking it to the DAAC version of the SDP Toolkit.	10.3	Critical
20	For each input dynamic data granule needed by the PGE, construct a Target MCF and Insert it to the Science Data Server.	12.3.1, 12.3.2, 12.3.3	Critical

**Table 3.1-1. Road Map of SSI&T for a PGE (3 of 3)**

Step	Task	Sections	Importance
21	For each input static data granule, construct a Target MCF and Insert it to the Science Data Server.	12.3.1, 12.3.2, 12.3.4	Critical
22	Assemble the SSEP (as a tar file) and Insert it to the Science Data Server.	12.4	Critical
23	Initiate a Production Request (PR) that will result in one or more DPRs.	13.1	Critical
24	Use the Planning Workbench to plan the PR and hence, run the PGE.	13.2	Essential
25	Monitor the PGE's progress using AutoSys.	13.3	Essential
26	Compare the output produced with test data delivered with the PGE.	14	Essential

## 3.2 How to Use the Procedures

The science software I&T operational procedures contained within this document are loosely ordered. The order is intended to suggest a logical sequence which, when used as a “road map”, represents an overall, sensible end-to-end SSI&T activity. In addition, since there are many factors that affect the actual SSI&T activities (*e.g.* Instrument Team deliveries, DAAC policies, and agreements between the Instrument Teams and the DAACs), the ordering in this document can only be suggestive.

Each procedure document is broken up into at least two sections. The first section contains the Activity Checklist. This is a table containing, in broad terms, the steps necessary for carrying out the procedure. It is intended to give an overview.

The second section describes each of these steps in more detail. This section is geared to someone unfamiliar with the procedure or someone needing verbose descriptions. Conventions used in this section are as follows: Text shown in **bold** represents what is to be typed in literally via the keyboard. Text shown in ***bold italics*** represents something to be filled in by something appropriate. For example:

***cd MyDir***

means type “cd” followed by a directory name which is appropriate to the task.

Text shown in **bold Helvetica** font represents names, labels, buttons, etc. on graphical user interfaces (GUIs). For example:

In the GUI labeled **Process Control File Checker**, click on the **OK** button.

In cases where a large number of steps are involved, a third section is added. This QuikStep table shows a condensed view of the steps required.

This page intentionally left blank.

## 4. Configuration Management

---

This section describes procedures for handling the configuration management of science software delivered to the DAACs for SSI&T. The COTS tool used for this purpose is ClearCase® by Atria Software, Inc. ClearCase can be run from the command line and via a graphical user interface (GUI).

All data managed under ClearCase are stored in VOBs (Versioned Object Bases), which are the “public” storage areas and views, which are the “private” storage areas. VOBs are data structures that can only be created by a VOB administrator using the `mkvob` (“make vob”) command. They are mounted as a file system and when viewed through a view, VOBs appear as standard UNIX directory tree structures. This file system, accessed through its mount point, has a version-control dimension that contains file elements and versions of file elements. ClearCase maintains multiple versions of a file organized into a hierarchical version tree that may include many branches. For the purposes of SSIT, it is suggested that branches not be created in managing the files of the science software. All versions of the software files should remain on their main branch of the tree and therefore easily accessed by the default configuration of any view that may be set by the user.

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 (in this case, a file) and either can be retrieved at a later time.

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 science software may be extremely large (multiple gigabytes) 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.

The administrator in charge of the VOB is referred to as the VOB administrator (VA).

All ClearCase procedures assume that the user’s `umask` is set to `002`.

For the Graphical User Interface version of the following procedures please refer to Release 6A Operations Tools Manual 609-CD-600-001 March 2001, section 4.3.1.

### 4.1 Creating a View in ClearCase

In order to make files and directories that are in a ClearCase VOB (Versioned Object Base) visible and accessible, a ClearCase view must be set. This procedure describes how to create a ClearCase view to be used later

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.

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

Assumptions:

1. The ClearCase VOB has been properly created and populated.
2. The VA has granted ClearCase privileges for the user.

To create a ClearCase view, execute the procedure steps that follow:

1. At a UNIX prompt, type **cleartool mkview -tag *ViewName* *ViewPath/ViewName.vws***.
  - The ***ViewPath*** is the full path name to the directory where ClearCase views are stored. The SA should supply this information. A typical example is `/net/mssg1sungsfc/viewstore/`.
  - The ***ViewName*** is the name of the view being created. View names should have `.vws` as the file name extension.

## 4.2 Setting a View in ClearCase

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

Section 6.1 describes how to create a ClearCase view.

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

Assumptions:

1. The ClearCase VOB has been properly created and populated.
2. A ClearCase view has been created for the user.

To set a ClearCase view, execute the procedure steps that follow:

1. At a UNIX prompt, type **cleartool setview *ViewName***.
  - The ***ViewName*** is the name of the view to be set.
  - The ***ViewName*** may be a view that another user has created and owns. The restriction, in this case, is that files cannot be checked in.

This 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 (*untar-red*) 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.

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view is not required to perform this procedure.

To import multiple files and/or directories into ClearCase from a UNIX directory structure, execute the procedure steps that follow:

- 1** At a UNIX prompt, type **cd *ParentPathname***.
  - The ***ParentPathname*** is the path name of the directory that *contains* the directory structure to be brought into ClearCase. This is *not* the VOB. For example, to bring the pge2 directory which sits under /MOPITT (i.e. its path name is /MOPITT/pge2) into ClearCase along with all files and subdirectories below it, **cd /MOPITT**.
- 2** At the UNIX prompt, type **clearcvt\_unix -r *DirName***.
  - The ***DirName*** is the name of the directory in which it and everything below it is to be brought into ClearCase. For example, based on the example in step 1, **clearcvt\_unix -r pge2**. A conversion script will be then be created. The **-r** causes all subdirectories to be recursively included in the script created.
- 3** Contact the VA and request that the utility script **cvt\_script** be run on the script created in step 2.
  - The VOB administrator (VA) is the only one who can run the **cvt\_script** because it modifies the VOB.

### 4.3 Entering a Single File into ClearCase

This procedure explains how to put a single file under configuration management using ClearCase.

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

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB directory can be seen.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1** At a UNIX prompt, type **cleartool setview *ViewName***.

- The *ViewName* is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe**.
- 2 At the UNIX prompt, type **cd *pathname***.
    - The *pathname* is the full path name of the subdirectory in the VOB into which the file is to be checked in. For example, to check a file into the VOB directory /VOB1/pge2/scripts/, type **cd /VOB1/pge2/scripts/** . 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 VA.
  - 3 At a UNIX prompt, type **cp *pathname/filename* .** , (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. For example, to copy over a file named MISR\_calib.c in directory /pge/pge34/ to be checked in, type **cp pge/pge34/MISR\_calib.c .** , (again, note the space and then “dot” at the end of the command).
  - 4 At the UNIX prompt, type **cleartool checkout -nc .** (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.
  - 5 At a UNIX prompt, type **cleartool mkelem -nc *filename***.
    - The *filename* is the name of the file that was copied over in step 4 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.
  - 6 At the UNIX prompt, type **cleartool checkin -nc *filename***.
    - 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 check in step.
  - 7 At the UNIX prompt, type **cleartool checkin -nc .** , (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 check in step.

**NOTE:** In the above **cleartool** commands and in some of the following sections, **-nc** stands for "no comment". To put comment in, use **-c "comment text"**. For example, **cleartool checkin -c "adding version 2" filename**. By default, **cleartool** expects a comment and in the case where neither option is used, **cleartool** will prompt for a comment. In such case, simply add the comment text and a dot at the end (indicating the end of comment).]

## 4.4 Entering a New Directory into ClearCase

This procedure explains how to put a new directory (empty) into ClearCase.

The assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

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

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB parent directory can be seen.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***,
  - The ***ViewName*** is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe**.
- 2 At the UNIX prompt, type **cd *pathname*, .**
  - The ***pathname*** is the full path name of the parent directory in the VOB in which the new directory is to be added. For example, if a new directory is to be added under /VOB1/pge4, type **cd /VOB1/pge4, .**
- 3 At the UNIX prompt, type **cleartool checkout -nc .**, (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. This directory will be the parent directory of the new directory. Adding a new directory (or element) to another directory represents a modification of the directory. Hence, the directory must be checked out before a new directory can be added and checked in.
- 4 At a UNIX prompt, type **cleartool mkdir -nc *DirectoryName*, .**
  - The ***DirectoryName*** is the name of the new directory being created. This command creates the new directory.

- The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the directory creation step.

**5** At the UNIX prompt, type **cleartool checkin -nc *DirectoryName*, .**

- The *DirectoryName* is the name of the new directory created in step 4. This command performs the check in of the new directory. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.

**6** At the UNIX prompt, type **cleartool checkin -nc .**, (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 directory) represents a modification to the current 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 check in step.

## 4.5 Checking Out an Element from ClearCase

This procedure explains how to check out an element from ClearCase so that it can be modified. 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 assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

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

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB element can be seen. See Section 6.7 for how to check in a modified element.

To enter a single file into ClearCase, execute the procedure steps that follow:

**1** At a UNIX prompt, type **cleartool setview *ViewName*,**

- The *ViewName* is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe, .**

**2** At the UNIX prompt, type **cleartool checkout -nc *element*, .**

- The *element* is the name of the file or directory (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.

- 3 This step is optional; it is performed to cancel a check out. At a UNIX prompt, type **cleartool uncheckout -nc *element***, .
  - The *element* is the name of the file or directory (full path name allowed) checked out (as in step 2, above). This command cancels the check out of an element. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the uncheck out step. Note that ClearCase will not allow an unmodified element to be checked in. Instead, the uncheck out command must be used.

## 4.6 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. See Section 6.6 on how to check out an element.

The assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

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

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB element can be seen. See Section 4.6 for how to check out an element before modifying.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***.
  - The *ViewName* is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe**.
- 2 At the UNIX prompt, type **cleartool checkin -nc *element***.
  - The *element* is the name of the file or directory (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.
- 3 This step is optional; it is performed when ClearCase does not accept a check in because the element was not modified. In this case, the check out must be canceled. At a UNIX prompt, type **cleartool uncheckout -nc *element***.

- The *element* is the name of the file or directory (full path name allowed) checked out. This command cancels the check out of an element. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the uncheck out step.

## 5. The ECS Assistant

---

The ECS Assistant is a GUI based program that simplifies the process of installation, testing and management of ECS. The utility is essentially an installation tool that has some practical applications that can be used to support SSIT functions. This utility provide users with a Graphical User Interface to perform functions such as subsystem server startup and shutdown and database review when using the ECS system. During the course of performing their tasks, SSI&T operators can use ECS Assistant to perform the following functions through its GUI:

- Start up and shut down servers for each subsystem
- Graphically monitor the server up/down status
- View the ECS Science Data Server database

The following sections, will address aspects of how to use the ECS Assistant in SSI&T activities. Section 5.1 explains how to use ECS Assistant to facilitate and manage the subsystems and their servers, including server start up and shut down. Section 5.2 contains information on viewing ESDT information and granule information, which allows reviewing the Science Data Server database. This capability is provided through the DB Viewer GUI in ECS Assistant.

The Whazzup GUI is now used to monitor and display the execution status and related performance statistics associated with ECS programs. Refer to 333-CD-600 for more information on the Whazzup GUI.

### 5.1 Using ECS Assistant to Start Up / Shut Down Servers

This procedure describes using the ECS Assistant GUI to start up and shut down subsystem servers. The procedure described here will apply to all the servers from different subsystems.

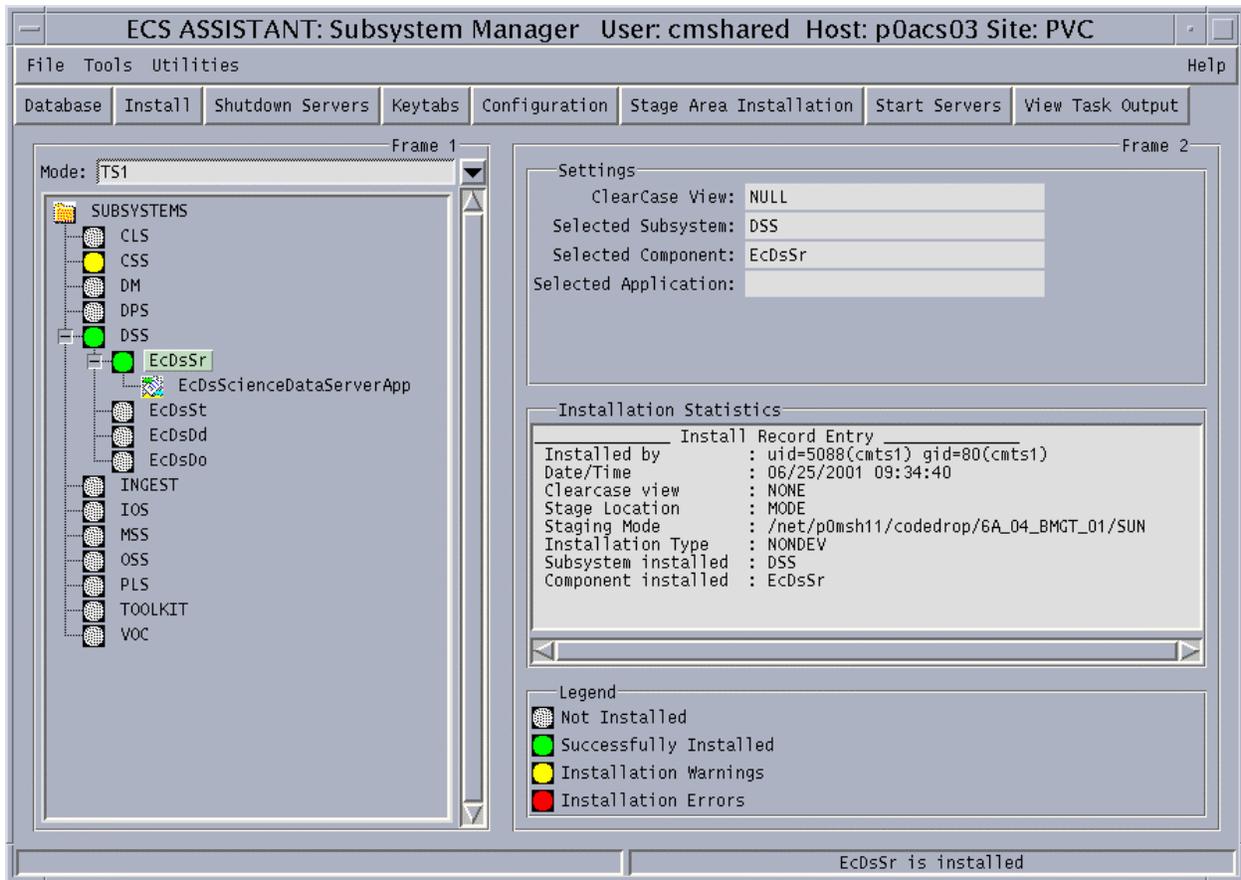
To run the ECS Assistant, execute the procedure steps that follow:

1. Log into the host where the desired server is installed. It is important that this use the same userID used to bring up the other servers.
2. **setenv ECS\_HOME /usr/ecs**
3. Type **/tools/common/ea/EcCoAssist /tools/common/ea &** or if the alias exists type **EA**. This will invoke the ECS Assistant GUI as indicated in Fig. 5.1-1.
4. At the ECS Assistant GUI, click the **Subsystem Manager** pushbutton. This will invoke the Subsystem Manager GUI, as indicated in Fig. 5.1-2.
5. Select a mode by clicking a mode in the mode listing.
6. Select a subsystem with the **Subsystem** window to locate servers of interest.

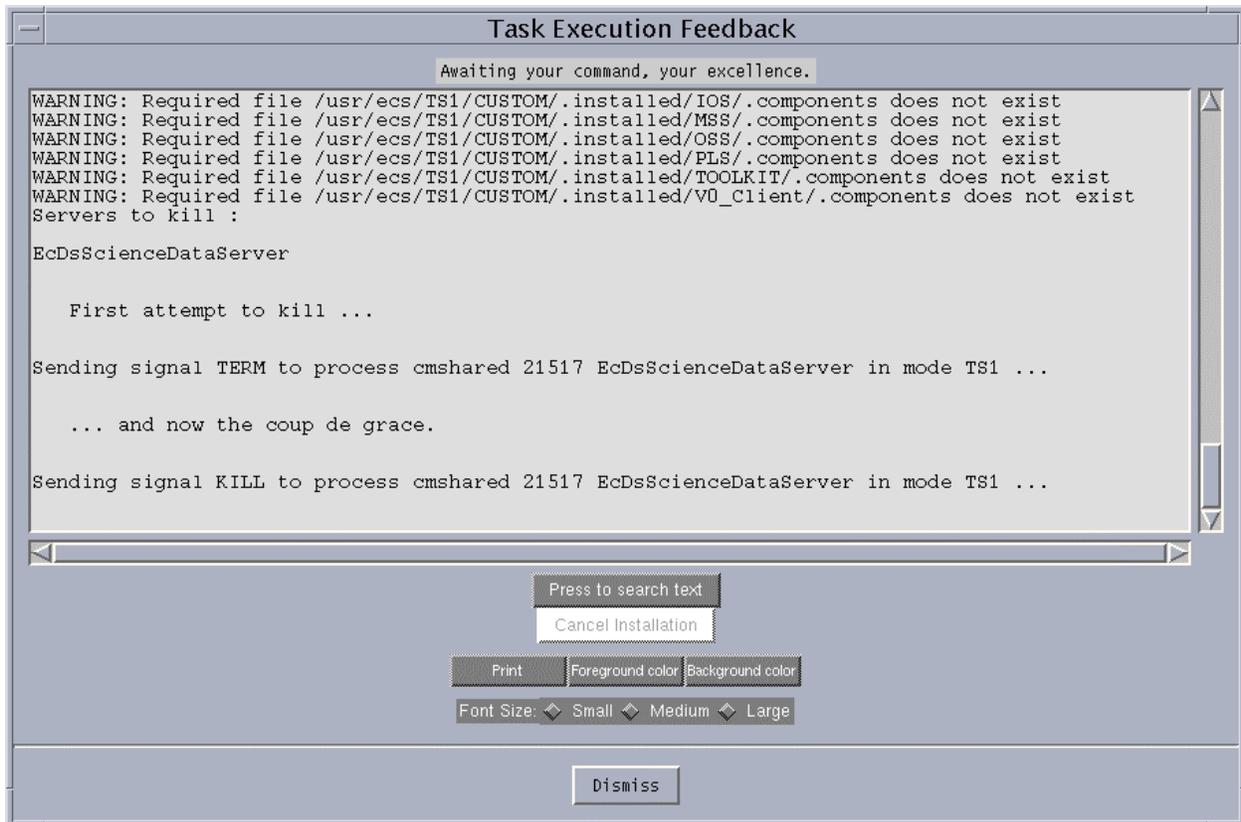
7. Select a component of the subsystem. At this level, all servers within a given component of the subsystem can be controlled. Selecting the server of interest can control an individual server.
8. To start the servers up or shut them down, click the **Shutdown Servers** or **Start Servers** button in the common task bar. This will bring start the Task Execution Feedback GUI, as indicated in Fig. 5.1-3. The GUI is used to monitor the startup or shutdown process of the selected servers. To exit the GUI select **DISMISS**.
9. To start up or shut down servers in other components on the same host, repeat steps 6-8. For components on other hosts, bring up ECS Assistant on that host and repeat steps 1-8.



**Figure 5.1-1. ECS Assistant GUI**



**Figure 5.1-2. Subsystem Manager GUI**



**Figure 5.1-3. Task Execution Feedback GUI**

## 5.2 Using ECS Assistant to View ECS Science Data Server Database

ESDTs and their granules stored in the archive are managed using an ECS 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.

To view the science data server database, execute the procedure that follows:

- 1 Invoke the ECS Assistant GUI as outlined in section 5.1.
- 2 At the ECS Assistant GUI, select the DB Viewer by clicking the **DB Viewer** button. See Fig. 5.1-1. This will start the **Login** GUI. See Fig. 5.2-1.
- 3 At the **Login** GUI, enter the appropriate information to view the Science Data Server Database appropriate for the mode of interest.
- 4 The **DB Viewer** GUI will appear as shown in Fig. 5.2-2. ESDTs that are installed are listed in the Collections window.
- 5 To view the inserted granules for a selected ESDT, first select an ESDT by clicking its short name in the Collections window. Any granules associated with the ESDT will be displayed in the Granules window.



**Figure 5.2-1. Database Login GUI**

**Datasever Database Viewer**

**Collections**

MOAPW1	Feb 14 2001	12:10:31:680PM	SC	MOAPW1.001	MODIS/Terra Ocean SemiAnalytic Primary Production 8-Day L4
MOAPYB	Feb 14 2001	12:12:12:946PM	SC	MOAPYB.001	MODIS/Terra Ocean SemiAnalytic Primary Production Yearly L4
MOD000	Feb 8 2000	10:33:47:200AM	SC	MOD000.001	MODIS Level 0 Raw Instrument Packets
MOD007	Sep 8 2000	9:08:54:530AM	SC	MOD007.001	MODIS Level 0 Raw Instrument Packets (APID 71)
MOD008	Sep 8 2000	9:10:41:620AM	SC	MOD008.001	MODIS Level 0 Raw Instrument Packets (APID 72)
MOD01	Oct 13 2000	12:03:02:336AM	SC	MOD01.199	MODIS/Terra Raw Radiances in Counts 5-Min L1A Swath
MOD01	Feb 14 2001	12:13:55:650PM	SC	MOD01.001	MODIS/Terra Raw Radiances in Counts 5-Min L1A Swath
MOD01LUT	Feb 24 2000	4:11:39:920PM	SC	MOD01LUT.001	MOD01 Engineering Telemetry Look-up Tables
MOD01SS	Mar 15 2001	3:41:39:943PM	SC	MOD01SS.001	MODIS/Terra Subsetted Raw Radiances in Counts Daily L1A Swa
MOD021KM	Oct 12 2000	11:45:44:193PM	SC	MOD021KM.199	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 1km
MOD021KM	Feb 15 2001	11:27:57:400AM	SC	MOD021KM.001	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 1km
MOD021QA	Oct 12 2000	11:38:38:893PM	SC	MOD021QA.199	MODIS/Terra QA Summary of Calibrated Radiances 5-Min L1B 1k
MOD021QA	Feb 15 2001	11:42:08:866AM	SC	MOD021QA.001	MODIS/Terra QA Summary of Calibrated Radiances 5-Min L1B 1k
MOD023	Sep 8 2000	9:12:17:670AM	SC	MOD023.001	MODIS Level 0 Raw Instrument Packets (APID 87)
MOD02HKM	Oct 12 2000	11:44:06:406PM	SC	MOD02HKM.199	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 500m
MOD02HKM	Feb 15 2001	11:31:21:176AM	SC	MOD02HKM.001	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 500m
MOD02HQA	Oct 12 2000	11:37:44:606PM	SC	MOD02HQA.199	MODIS/Terra QA Summary of Calibrated Radiances 5-Min L1B 50
MOD02LUT	Feb 24 2000	4:07:51:340PM	SC	MOD02LUT.001	MODIS Instrument Calibration Parameters Lookup Table for Pr
MOD02OBC	Oct 12 2000	11:42:30:730PM	SC	MOD02OBC.199	MODIS/Terra On-Board Calibrator and Engineering Data 5-Min
MOD02OBC	Feb 15 2001	11:33:01:723AM	SC	MOD02OBC.001	MODIS/Terra On-Board Calibrator and Engineering Data 5-Min
MOD02QKM	Oct 12 2000	11:41:01:246PM	SC	MOD02QKM.199	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 250m
MOD02QKM	Mar 19 2001	10:42:46:036AM	SC	MOD02QKM.001	MODIS/Terra Calibrated Radiances 5-Min L1B Swath 250m

**Granules**

dbID	insertTime	LocalGranuleID	BeginningDateTime	EndingDateTime
20755	Feb 15 2001	4:47:06:313PM	MOD01.A1997226.0030.	Aug 14 1997 12:30AM
21369	Mar 15 2001	11:09:53:480AM	MOD01.A1997226.0030.	Aug 14 1997 12:30AM
22026	Apr 25 2001	5:55:43:093PM	MOD01.A2000105.1820.	Apr 14 2000 6:20PM
22203	Apr 26 2001	3:23:21:970PM	MOD01.A2000105.1820.	Apr 14 2000 6:20PM
22027	Apr 25 2001	5:56:58:766PM	MOD01.A2000105.1825.	Apr 14 2000 6:25PM
22204	Apr 26 2001	3:25:32:346PM	MOD01.A2000105.1825.	Apr 14 2000 6:25PM
22028	Apr 25 2001	5:58:23:646PM	MOD01.A2000105.1830.	Apr 14 2000 6:30PM
22206	Apr 26 2001	3:27:12:970PM	MOD01.A2000105.1830.	Apr 14 2000 6:30PM
23955	May 18 2001	2:53:27:540PM	MOD01.A2000105.1830.	Apr 14 2000 6:30PM
23986	May 18 2001	4:34:09:096PM	MOD01.A2000105.1830.	Apr 14 2000 6:30PM
24119	May 21 2001	12:51:18:886PM	MOD01.A2000105.1830.	Apr 14 2000 6:30PM
23034	May 4 2001	5:06:22:863PM	MOD01.A2000105.1835.	Apr 14 2000 6:35PM
23957	May 18 2001	2:54:43:200PM	MOD01.A2000105.1835.	Apr 14 2000 6:35PM
23991	May 18 2001	4:35:13:366PM	MOD01.A2000105.1835.	Apr 14 2000 6:35PM
24127	May 21 2001	12:52:37:543PM	MOD01.A2000105.1835.	Apr 14 2000 6:35PM

Select a line in the collections pane to see granules for that collection  
 Select a line in the granules pane to see more data for the granule

Show Validation Rules	Show PSA Info	Show Summary Info
Exit	Show ESDTs	Help

**Figure 5.2-2. DB Viewer GUI**

## 6. The SSIT Manager

---

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.

### 6.1 Description of SSIT Manager Tools

Across the top of the SSIT Manager are the toolbar items File, Tools, and Run. Clicking on each of these invokes a pull-down menu.

Under the File pull-down menu, the only item is Quit. Clicking on this causes the SSIT Manager to terminate.

The Tools pull-down menu has most of the SSIT Manager's tools. The menu items are:

- Xterm – opens xterm in Korn shell
- Code Analysis contains
- SPARCwork - A COTS package provided by Sun that allows for various coding activities including memory checking and debugging.
- Office Automation contains
- Ghostview - for viewing PostScript formatted documents.
- Netscape - WWW browser and useful for viewing HTML formatted documents.
- Acrobat - for viewing PDF formatted documented.
- Standards Checkers contains
- FORCHECK - for standards checking for FORTRAN 77 and FORTRAN 90 science software source code.
- Prohibited Function Checker - for checking science software source code for prohibited functions.
- Process Control File Checker - for checking Process Control Files (PCFs) delivered with science software.
- Prolog Extractor - for extracting prologs from science software source code.
- Product Examination contains
- IDL - Interactive Data Language tool.
- EOSView - for viewing HDF and HDF-EOS files.

- File Comparison contains
  - ASCII - for comparing two output products that are in ASCII format.
  - Binary - for comparing two output products that are in binary format.
  - HDF (GUI) - for comparing two output products that are in HDF or HDF-EOS format, GUI version.
  - HDF (hdiff) - for comparing two output products that are in HDF or HDF-EOS format, command line tool.
- Text Editors contains
- Emacs
- Xedit
- PDPS Database contains
- PCF ODL Template - for converting delivered PCFs into ODL during PGE registration.
- Check ODL - for verifying ODL syntax of ODL files.
- SSIT Science Metadata Update - for updating the PDPS database with PGE information during PGE registration.
- SSIT Opnl Metadata Update - GUI for updating the PDPS database with PGE information during PGE registration.
- Data Server contains
- Acquire DAP - for acquiring a Delivered Algorithm Package (DAP).
- Get MCF – for generating MCF for the input and output ESDTs.
- Insert Static - for inserting a static data file to the Data Server.
- Insert Test Dynamic - for inserting a dynamic test data file to the Data Server.
- Insert EXE TAR - for inserting a Science Software Executable Package (SSEP) to the Data Server.
- SSAP Editor - for editing and creating a Science Software Archive Package (SSAP) and inserting it to the Data Server.

The Run 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.

## 6.2 Starting the SSIT Manager

The SSIT Manager requires a configured environment within which to run, and only on the AIT Suns. This section describes the steps to follow to start the SSIT Manager on the appropriate AIT Sun.

To run the SSIT Manager, execute the procedure steps that follow. Note that in the following, ECS Custom Software is nominally in the directory `/usr/ecs/<mode>/CUSTOM/utilities` on the AIT Sun:

1. Log onto the AIT Sun.
2. Change directory to `/usr/ecs/<mode>/CUSTOM/utilities` and type `EcDpAtMgrStart mode &`. This starts the SSIT Manager, and displays the SSIT Manager GUI.

This page intentionally left blank.

## 7. 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 the 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 ECS and the metadata into the Data Server, Advertising, Data Dictionary and Subscription databases. Once this has been done, the **acquire** service is used to retrieve it.

### 7.1 Performing a DAP Insert

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

Assumptions:

1. The DAP ESDT has been installed on the Data Server. (See section 18 on how to install ESDT's.)
2. The Target MCF for the DAP has been created for the Insert. (see section 7.2)
3. The SSIT Manager is running. (See section 6 on how to bring it up)

To Insert a DAP to ECS, execute the following steps:

1. From the SSIT Manager, click on the Tools -> Data Server -> Insert Test Dynamic. A xterm with title "SSIT: PGE Test Dynamic Input File Insertion" will be displayed.
2. At the program prompt **Configuration filename? (enter for default: ../.cfg/EcDpAtInsertTestFile.CFG)**, press Return.
3. At the program prompt **ECS Mode of operations?**, type in the *<mode>* you are working in.
4. At the program prompt **ESDT short name for the file(s) to insert?**, type *DAP*.
5. At the program prompt **ESDT Version for the file(s) to insert?**, type in the *ESDT version*.

- 6 At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?**, type *N*. Typically the DAP is a single tar file. If it is composed of multiple files, type *Y*.
- 7 At the program prompt **Single Filename to Insert? (including FULL path)**, type the full path to the DAP that is to be inserted.
- 8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)**, type the full path to the metadata file that corresponds to the DAP.

## 7.2 An Example of a DAP Metadata File

The following is an example of a DAP target metadata file. Three attribute values needs to be modified based on each DAP. The attributes are DAPPGEName, DAPPGVersion, and DAPSWVersion.

```

GROUP = INVENTORYMETADATA
GROUPTYPE = MASTERGROUP
  GROUP = CollectionDescriptionClass
    OBJECT = VersionID
      NUM_VAL = 1
      Value = 001
    END_OBJECT = VersionID
  OBJECT = ShortName
    NUM_VAL = 1
    Value = "DAP"
  END_OBJECT = ShortName
END_GROUP = CollectionDescriptionClass
GROUP = ECSDDataGranule
  OBJECT = ProductionDateTime
    NUM_VAL = 1
    TYPE = "1997-12-24T17:45:19.000Z"
  END_OBJECT = ProductionDateTime
END_GROUP = ECSDDataGranule
OBJECT = DAPID
  NUM_VAL = 1
  Value = "SomeID"
END_OBJECT = DAPID
OBJECT = DAPInsertDate
  NUM_VAL = 1
  Value = "1997-12-24"
END_OBJECT = DAPInsertDate
GROUP = PGEGroups
  OBJECT = PGEGroupContainer
    CLASS = "0"
  
```

```

OBJECT = DAPPGEVersion
  CLASS = "0"
  NUM_VAL = 1
  Value = "1.0"
END_OBJECT = DAPPGEVersion
OBJECT = DAPSWVersion
  CLASS = "0"
  NUM_VAL = 1
  Value = "2.0"
END_OBJECT = DAPSWVersion
OBJECT = DAPPGEName
  CLASS = "0"
  NUM_VAL = 1
  Value = "Joe PGE"
END_OBJECT = DAPPGEName
END_OBJECT = PGEGroupContainer
END_GROUP = PGEGroups
END_GROUP = INVENTORYMETADATA
END

```

### 7.3 Insert Testing of Products

This same utility is used to insert dynamic products that are delivered with the DAP into the Data Server. Once the product is in the Data Server, the DAP acquire service can be used to retrieve it.

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

Assumptions:

1. The product ESDT has been installed on the Data Server. (See section 5.3 on how to install ESDT's.)
2. The SSIT Manager is running. (See section 6 on how to bring it up)

To Insert a DAP to the Science Data Server, execute the following steps:

1. From the SSIT Manager, click on the **T**ools -> **D**ata **S**erver -> **I**nsert **T**est **D**ynamic.
  - A xterm with title "SSIT: PGE Test Dynamic Input File Insertion" will be displayed.
2. At the program prompt Configuration filename? (enter for default: ../../cfg/EcDpAtInsertTestFile.CFG), press Return.
3. At the program prompt **ECS Mode of operations?**, type in the <mode> you are working in.
4. At the program prompt ESDT short name for the file to insert? type *ShortName* for the product to test.
5. At the program prompt ESDT Version for the file(s) to insert?, type in the *ESDT version*.

6. At the program prompt Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?, type *Y* if it is a multifile granule or *N* if it is a single granule.
7. At the program prompt Single Filename to Insert? (including FULL path), type *full path to the dynamic granule*.
8. At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** , type the *full path to the metadata file* corresponding to dynamic granule being inserted. If the insert is successful, a UR will be returned.

## 8. 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 the 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 and granules into the Data Server, as outlined in Chapter 7. Once the files are in the Data Server, the DAP acquire service is used to retrieve them.

Note that 2 files will be acquired, the file itself and the metadata associated with the file. Section 8.1 describes the procedure to acquire a DAP or product using SSIT Manager.

Note that the DAP acquire methods discussed in this section can be applied to any type of files that have been inserted into the Science Data Server, such as science data granules and PGEEEXE.tar file.

### 8.1 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:

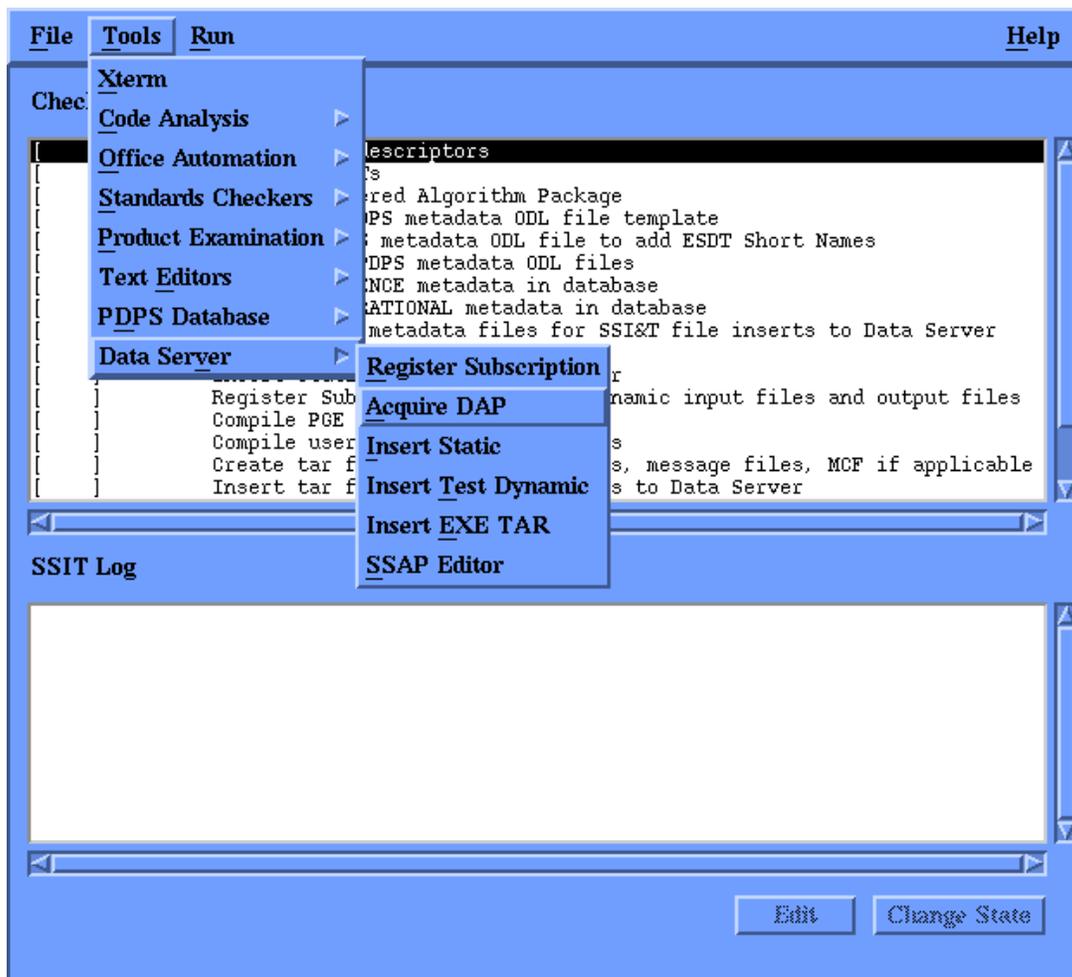
1. All required Servers are up running properly.
2. SSI&T Manager is up running. (see section 6.2 how to bring up SSI&T Manager)
3. The following actions have occurred:
  - The DAP has been inserted into the Science Data Server.
  - Subscription Server has sent email to the SSIT operator providing notification of DAP insertion and the UR of the DAP

Bring up the SSIT Manager GUI (see section 6 for detailed instructions).

1. From the SSIT Manager top menu bar, select **Tools -> Data Server -> Acquire DAP**. See figure 8.1-1.
2. At the prompt **Configuration filename? (enter for default: DpAtAA.CFG)**, press return.
3. At the prompt **ECS Mode of operations?**, type *<mode>*.

4. At the prompt Name of email message file (including path)?, type the *full path to the file* *ascii file which contains the UR for the file of interest*.
5. At the prompt **Directory to receive staged file?**, type *the full path to the directory the acquired file is to be staged*.

Notes: If the message indicates failure to acquire, see Appendix A.3 to diagnose the problem.



**Figure 8.1-1. Selecting DAP Acquire from SSIT Manager**

## 9. PGE Checkout

---

All science software delivered to the DAACs to be run in the ECS must comply with the NASA Earth Science Data and Information Systems (ESDIS) Project standards. The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996 (423-16-01)*.

The current standards mandate that all science software are to be written in C, Fortran 77 or Fortran 90. Allowed script languages include C shell, Bourne shell, Korn shell and Perl. The standards for C, Fortran 77, and Fortran 90 are, in general, those of ANSI compliance. Certain extensions, particularly in Fortran 77, are allowed by the guidelines.

In addition, the ESDIS standards and guidelines document mandates that certain functions are prohibited from science software. These are generally functions (or script utilities) that are a problem for the ECS.

### 9.1 Checking for ESDIS Standards Compliance in Fortran 77

This procedure describes how to use the COTS tool FORCHECK to check science software written in Fortran 77 for ESDIS standards compliance. FORCHECK can be accessed from the UNIX prompt using command-line operations or from the SSIT Manager.

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

Assumptions:

1. The Fortran 77 science software source code is available, accessible, and has read permission for the user.
2. SSIT Manager is available for use.

FORCHECK is available only on the AIT Suns.

To check for ESDIS standards compliance in Fortran 77 code, execute the procedure steps that follow:

1. Launch the SSIT Manager (refer to Chapter 7 for detailed instructions for doing this).
2. From SSIT Manager top menu, select **Tools->Standards Checkers->FORCHECK**.
3. A separate FORCHECK window will now open, prompting the user for input.
  - The first prompt will be **global option(s) and list file?**The second prompt will be **local option(s) and file(s)?**

- The second prompt will be repeated until there is a blank line and carriage return.
  - In order to understand what the proper responses should be, the user is encouraged to consult the documentation for FORCHECK.
4. Examine the contents of the *FORCHECKoutput* file. The *FORCHECKoutput* is the file name for the output file produced in step 3. This file will contain any warnings, errors, and other messages from FORCHECK.

## 9.2 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:

1. The Fortran 90 science software source code is available, accessible, and has read permissions for the user.
2. Required Status Message Facility (SMF) files have been compiled (see Section 10.3).
3. The C shell (or a derivative) is the current command shell.
4. The Fortran 90 compiler is available on the SPR SGI.

To check for ESDIS standards compliance in Fortran 90 code, execute the procedure steps that follow:

1. Locate the PGS toolkit and then type **source \$PGSBIN/FLAVOR/pgs-dev-env.csh**, where \$PGSBIN is the full path to the bin directory of the toolkit and FLAVOR is the toolkit to be used, ie the scf or daac flavor of the fortran 90 toolkits. (See Table 10.2-2).
2. At the UNIX prompt on the SPR SGI, type `f90 -c -ansi [-I$PGSINC] [-I$HDFINC] [I-IOtherIncFiles]...] FullPathToSourceFiles >& ReportFile`.
  - The terms in square brackets (*[ ]*) are used to optionally specify locations of include. \$PGSINC is the path to the SDP Toolkit include directory and \$HDFINC contains the HDF include directory. OtherIncFiles represents one or more additional directories that contain header files used by the files being checked.

- ***FullPathToSourceFiles*** is a list (space delimited) of Fortran 90 source files or a wildcard template (*e.g.* \*.f90).
- ***ReportFile*** is the file name under which to save the results of the compilation.
- The -c flag causes only compilation (no linking).
- The -ansi flag enables ANSI checking.

3. Examine the ***ReportFile*** for warnings and errors.

### 9.3 Checking for ESDIS Standards Compliance in C and C++.

The C compiler is used to perform the checking ESDIS standards compliance for C.

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

Assumptions:

1. The C science software source code is available, accessible, and has read permissions for the user.
2. Required Status Message Facility (SMF) files have been compiled (see Section 10.3).
3. The C shell (or a derivative) is the current command shell.
4. The C compiler is available on the SPR SGI.

To check for ESDIS standards compliance in C code, execute the procedure steps that follow:

- 1 At the UNIX prompt on the SPR SGI, type **cc -c -ansi [-I\$PGSINC] [-I\$HDFINC] [[-OtherIncFiles]...] *FullPathToSourceFiles* >& *ReportFile*.**
  - The terms in square brackets (*I J*) are used to optionally specify locations of include. \$PGSINC is the path to the SDP Toolkit include directory and \$HDFINC contains the HDF include directory. OtherIncFiles represents one or more additional directories that contain header files used by the files being checked.
  - ***FullPathToSourceFiles*** is a list (space delimited) of C or C++ source files or a wildcard template (*e.g.* \*.c or \*.cpp).
  - ***ReportFile*** is the file name under which to save the results of the compilation.
  - The -c flag causes only compilation (no linking).
  - The -ansi flag enables ANSI checking.

## 9.4 Checking for Prohibited Functions

The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996* (423-16-01). This ESDIS document mandates that science software delivered to the DAACs to be integrated into the ECS be free from prohibited functions. The procedures that follow describe how to use the Prohibited Function Checker, available with and without a GUI.

Section 9.4.1 describes how to use the Prohibited Function Checker GUI. Section 9.4.2 describes how to use the Prohibited Function Checker from the command line. Both versions of the checker are functionally identical.

### 9.4.1 Checking for Prohibited Functions: GUI Version

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

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

Assumptions:

1. The SSIT Manager is running and that the source files to be checked are available, accessible, and have read permissions for the operator.
2. Source files to be checked are C, C++, Fortran 77, Fortran 90, C shell, Korn shell, Bourne shell, or Perl, and have recognized file name extensions (Table 9.4.1-1).

**Table 9.4.1-1. File Name Extensions Recognized**

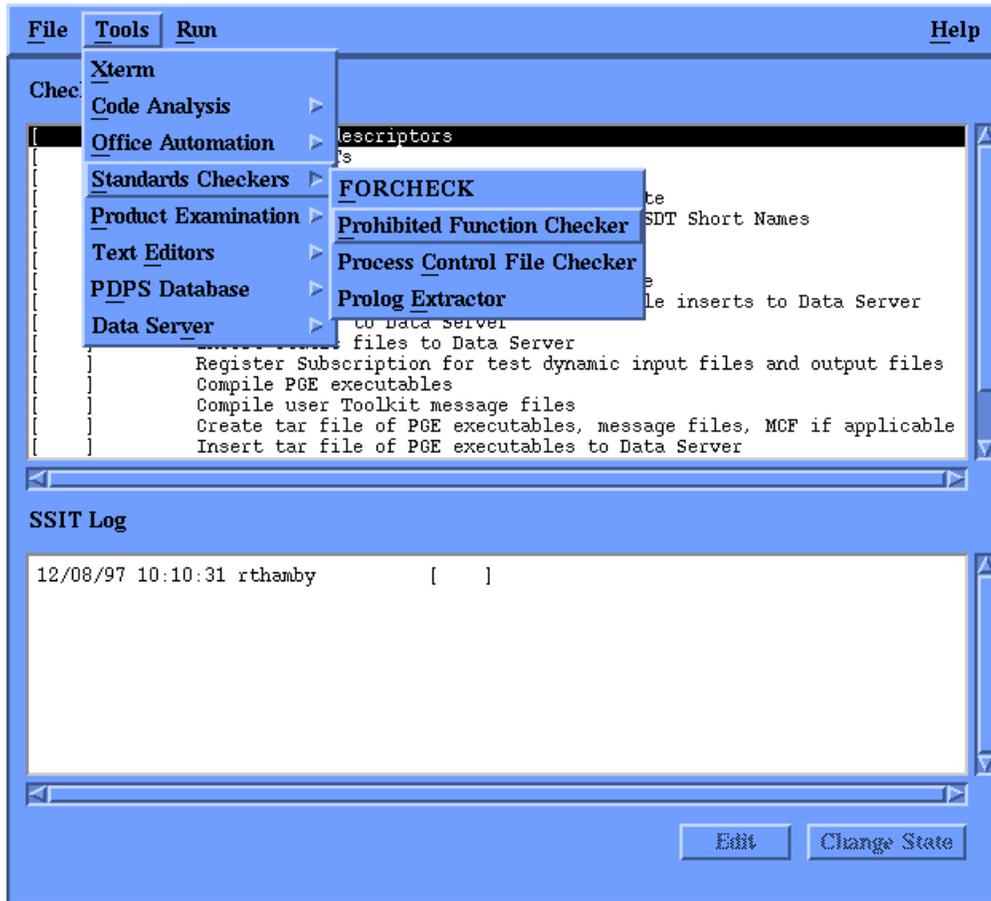
Language	File Name Extensions
C	.c, .h
C++	.cpp, .h
Fortran 77	.f, .f77, .ftn
Fortran 90	.f90
C Shell	.csh
Korn Shell	.ksh
Bourne Shell	.sh
Perl	.pl

To check Prohibited Functions, execute the procedure steps that follow:

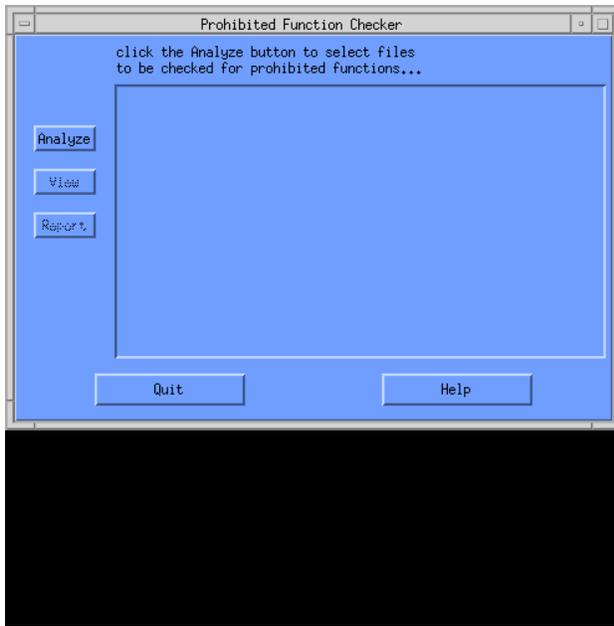
1. From the SSIT Manager, click **Tools-> Standards Checkers -> Prohibited Function Checker**. The Prohibited Function Checker GUI will be displayed. See Figures and 9.4.1-1 and 9.4.1-2.

2. In the Prohibited Function Checker GUI, click on the **Analyze** button. The File Selector GUI will be displayed. See Figure 9.4.1-3.
3. Within the **Directories** subwindow, work your way to the desired directory.
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. See Figure 9.4.1-4 for an example of the **Files:** subwindow after several files have been selected and subsequently highlighted.
5. In the **File Selector GUI**, click on the **Ok** button. The **File Selector GUI** will disappear.
  - The files selected in step 5 will be displayed in the Prohibited Function Checker GUI window as they are being checked (Figure 9.4.1-5).
  - If no prohibited functions are found, the **Prohibited Function Checker GUI** will be displayed with the message no prohibited functions found. You can then go to step 9 (to quit) or go to step 2 and pick more files to analyze. If prohibited functions were found, proceed to step 6.
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 or 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.
  - Click on the **Done** button to close the **Source Code** GUI. Other source files may be examined similarly, one at a time.

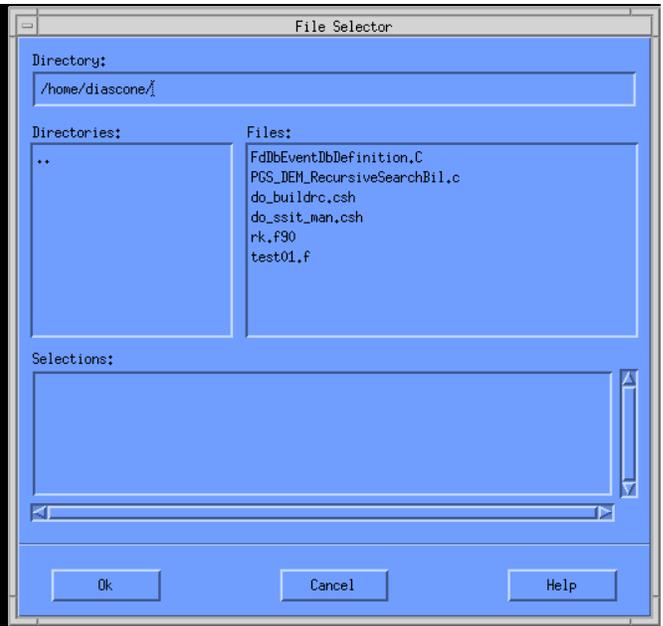
9. In the Prohibited Function Checker GUI, click on the **Quit** button.
- The **Prohibited Function Checker GUI** will disappear.



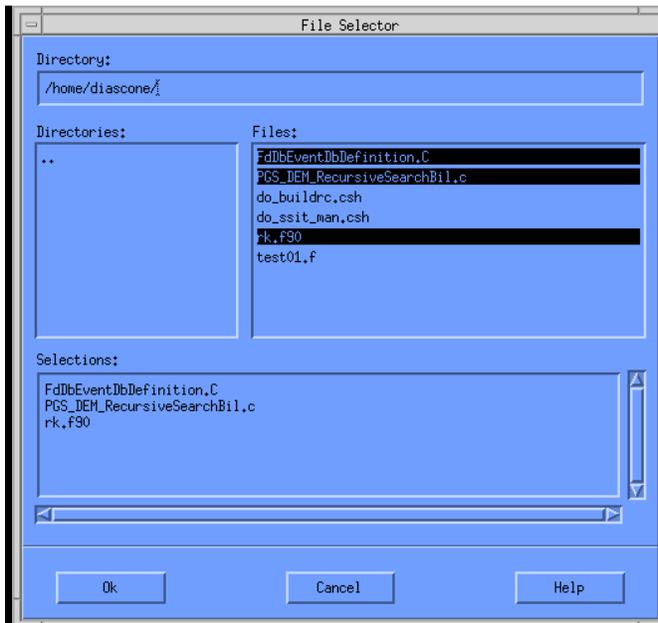
**Figure 9.4.1-1. Invoking the Prohibited Function Checker**



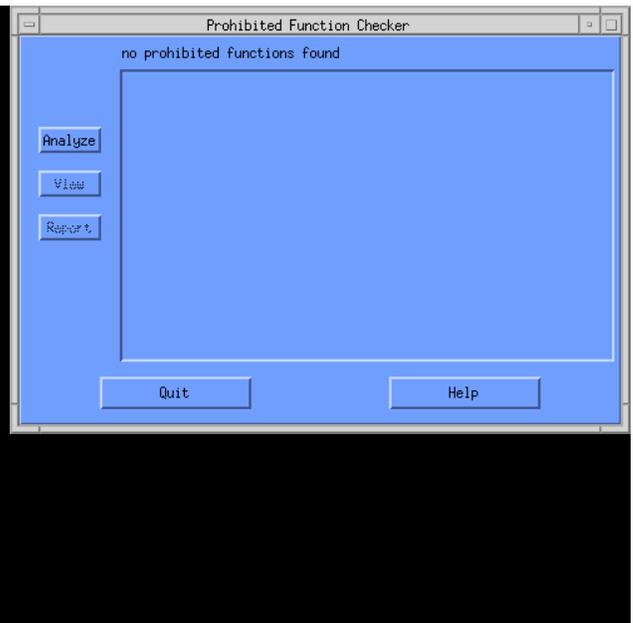
**Figure 9.4.1-2. Starting Screen for Prohibited Function Checker**



**Figure 9.4.1-3. File Selection Menu**



**Figure 9.4.1-4. Selected Files**



**Figure 9.4.1-5. Results Screen**

## 9.4.2 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 prohibited functions.

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

Assumptions:

1. The SSIT Manager and its various tools have been installed in the standard directories.
2. The source files to be checked are available, accessible, and have read permissions for the operator.
3. Source files to be checked are C, C++, Fortran 77, Fortran 90, C shell, Korn shell, Bourne shell, or Perl, and have recognized file name extensions (Table 9.4.1-2).

To check for prohibited functions in delivered source files, execute the procedure steps that follow:

- 1 At the UNIX prompt on the AIT Sun, type  
**`/usr/ecs/<mode>/CUSTOM/bin/DPS/DpAtMgrBadFunc ConfigFile`**  
**`/usr/ecs/<mode>/CUSTOM/cfg/EcDpAtBadFunc.CFG ecs_mode mode`**  
**`FilesOrDirectories > ResultsFile.`**
  - FilesOrDirectories is a list of names of source file or of directories containing source files.
  - ResultsFile is the file name for the results that are output.
- 2 Examine the results in the *ResultsFile*.

## 9.5 Checking Process Control Files

Process Control Files (PCFs) should be delivered for each Product Generation Executive (PGE). Only one PCF can be associated with a PGE; however, more than one may be delivered. This procedure describes how to check PCFs for valid syntax and format using the Process Control File Checker, available with and without a GUI.

Section 9.5.1 describes how to use the Process Control File Checker GUI. Section 9.5.2 describes how to use the Process Control File Checker from the command line. Both versions of the checker are functionally identical.

### 9.5.1 Checking Process Control Files: GUI Version

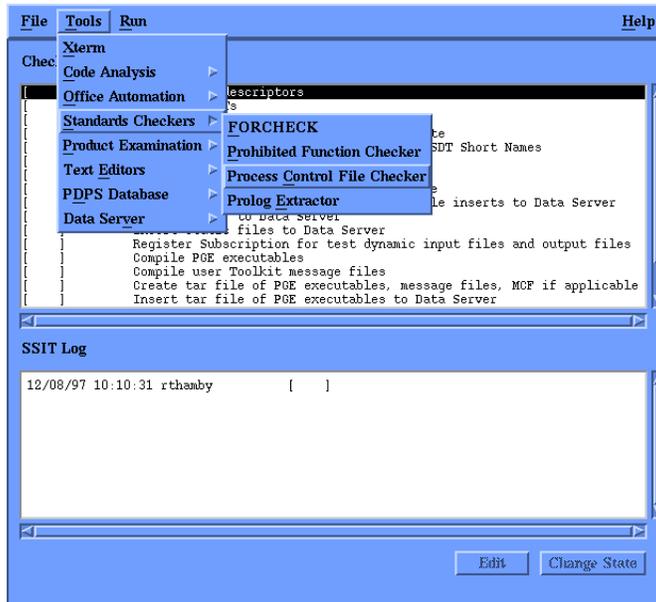
Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

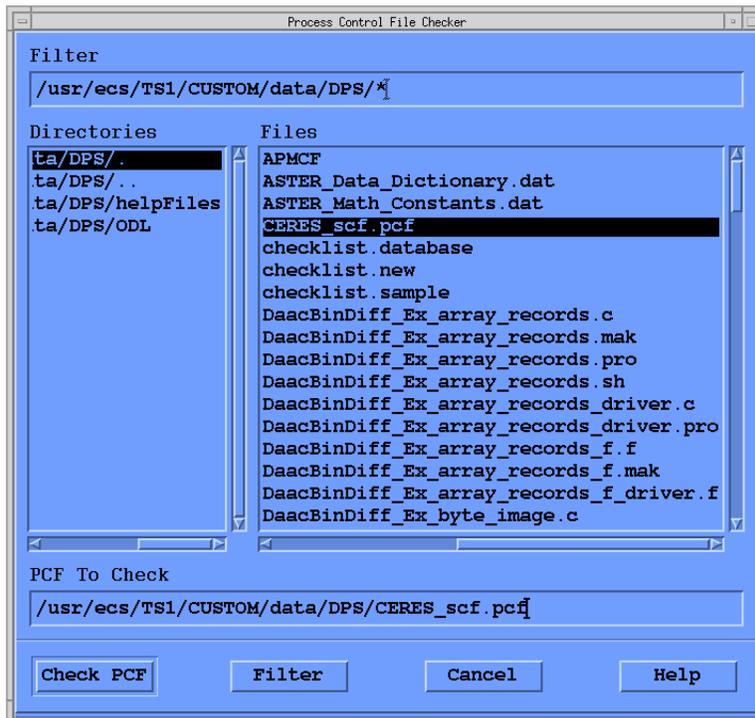
1. The SSIT Manager is running.
2. The PCFs to be checked are available, accessible, and have read permissions for the operator.

To check Process Control Files, execute the procedure steps that follow:

- 1** From the SSIT Manager, click **T**ools->**S**tandards Checkers -> **P**rocess Control File Checker. See Figure 9.5.1-1.
  - The Process Control File Checker GUI will be displayed. (See Figure 9.5.1-2.)
- 2** In the **D**irectories 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 **F**ilter subwindow to limit which files are displayed.
- 3** Within the **F**iles 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 **C**heck PCF button.
  - A GUI labeled **PCF Checker Results** will be displayed.
  - Results will be displayed in this window.
- 5** Optionally, click on the **S**ave button or on the **P**rint button.
  - Choose **S**ave to save the results to a file;
  - Choose **P**rint to have the results printed on the default printer.
  - Choosing **S**ave will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.
  - Choosing **P**rint and then clicking on the **O**K button will send the results to the default printer.
- 6** Click on the **C**heck Another button or on the **Q**uit button.
  - Choosing **C**heck Another allows another PCF to be checked. Repeat steps 2 through 5.



**Figure 9.5.1-1. Invoking the Process Control File Checker**



**Figure 9.5.1-2. Selecting a File to Check**

## 9.5.2 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:

1. The PCF files to be checked are available, accessible, and have read permissions for the operator.
2. You will need the command **pccheck.sh**. One way to see if this is available is to type **which pcccheck.sh**. If a path is displayed, then the directory is in your path. In this case, you will have to set a ClearCase view to access that area.

To check Process Control Files, execute the procedure steps that follow:

- 1 On the AIT Sun, type **cd PCFpathname**.
  - **PCFpathname** is the full path name to the location of the Process Control File(s) to be checked.
- 2 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, see Section 10.2) and type **\$PGSBIN/FLAVOR/pccheck.sh -i PCFfilename > ResultsFile**.
- 3 Examine the ResultsFile.

## 9.6 Extracting Prologs

The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996* (423-16-01). This ESDIS document mandates that science software delivered to the DAACs to be integrated into the ECS 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 extract the prologs it finds. It does not check the contents of prologs.

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

Assumptions:

1. The SSIT Manager is running.
2. The source files from which prologs are to be extracted are available, accessible, and have read permissions for the operator.

3. Prologs are delimited by particular delimiters depending on the language type. Delimiters are listed in the Table 9.6-1.
4. Source files have file name extensions shown in Table 9.6-2.

**Table 9.6-1. Extracting Prologs - 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	

**Table 9.6-2. Extracting Prologs - File Name Extensions**

File Type	File Name Extensions
Fortran 77	f, f77, ftn, for, F, F77, FTN, FOR
Fortran 90	f90, F90, f, F
Fortran 77/Fortran 90 include	inc, INC
C	c
C/C++ header	h

To extract prologs from delivered source files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click **Tools->Standards Checkers -> Prolog Extractor**. This will start a xterm (on the AIT Sun) within which the Prolog Extractor will be run.
- 2 At the program prompt **Configuration filename?**, type return.
  - The default configuration file for this tool will be used.
- 3 At the program prompt **ECS Mode of Operations?**, type in the *<mode>* you are working in.
- 4 At the program prompt **OutputFile?**, type in the *directory and the name of your output file*.
- 5 Examine the output file specified above. The default location of this file is the directory from which the SSIT Manger was invoked

# 10. 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 for the ECS at independent SCFs. Science software delivered to the DAACs for SSI&T is in the form of source files. In order to be run and tested within the ECS, this science software has to be compiled and linked to one of the SDP Toolkit versions resident at the DAAC to form the binary executables that run within the Product Generation Executives (PGEs).

## 10.1 Setting Up the SDP Toolkit Environment

Science software developed and tested at the SCFs is built using the SCF version of the SDP Toolkit. This version of the SDP Toolkit enables the SCFs to simulate to some extent a DAAC environment. The DAAC version of the SDP Toolkit uses an identical Applications Programming Interface (API) as the SCF version, however, it contains additional hooks into the ECS and provide full ECS functionality. Science software developed at the SCFs with the SCF version of the SDP Toolkit should be able to build and run with the DAAC version of the SDP Toolkit with no changes.

Each SDP Toolkit version (SCF and DAAC) has different flavors depending upon the object type mode and upon the language. This operating system allows compilers to build binaries in old 32-bit mode, new 32-bit mode, or in 64-bit mode (default). Table 10.1-1 lists the modes available and the corresponding compiler flags to enable them.

**Table 10.1-1. Object Types on the SGI**

Object Type	Compiler Flag Used
Old 32-bit Mode	-32
New 32-bit Mode	-n32
64-bit Mode	-64

In addition, the SDP Toolkit uses different libraries depending upon what language of source code is being linked. If only C source code is being linked, then either language library can be used.

There are three object types (see Table 10.1-1) and a thread safe version for each object type. There are three language types for each object type and thread safe version. Therefore, there are twelve versions of the DAAC Toolkit at the DAACs. There is a corresponding SCF version of the Toolkit available for each DAAC version making a total of 24 Toolkit versions available. Table 10.1-2 lists these versions. This table lists the Toolkit version (SCF or DAAC), language type, library object mode, and the directory of the appropriate SDP Toolkit.

Note - Each Toolkit library comes with a debug version, for example: sgi32\_daac\_cpp/ and sgi32\_daac\_cpp\_debug/

**Table 10.1-2. SDP Toolkit Versions at the DAAC**

<b>SDP Version</b>	<b>Language Type</b>	<b>Library Object Type</b>	<b>FLAVORS Located in \$PGSBIN/</b>
SCF	C++ or C	Old 32-bit mode	sgi_scf_cpp
SCF	FORTRAN 77 or C	Old 32-bit mode	sgi_scf_f77
SCF	Fortran 90 or C	Old 32-bit mode	sgi_scf_f90
SCF	Thread	Old 32-bit mode	sgi_scf_r
SCF	C++ or C	New 32-bit mode	sgi32_scf_cpp
SCF	FORTRAN 77 or C	New 32-bit mode	sgi32_scf_f77
SCF	Fortran 90 or C	New 32-bit mode	sgi32_scf_f90
SCF	Thread	New 32-bit mode	sgi32__scf_r
SCF	C++ or C	64-bit mode	sgi64_scf_cpp
SCF	FORTRAN 77 or C	64-bit mode	sgi64_scf_f77
SCF	Fortran 90 or C	64-bit mode	sgi64_scf_f90
SCF	Thread	64-bit mode	sgi64_scf_r
DAAC	C++ or C	Old 32-bit mode	sgi_daac_cpp
DAAC	FORTRAN 77 or C	Old 32-bit mode	sgi_daac_f77
DAAC	Fortran 90 or C	Old 32-bit mode	sgi_daac_f90
DAAC	Thread	Old 32-bit mode	sgi_daac_r
DAAC	C++ or C	New 32-bit mode	sgi32_daac_cpp
DAAC	FORTRAN 77 or C	New 32-bit mode	sgi32_daac_f77
DAAC	Fortran 90 or C	New 32-bit mode	sgi32_daac_f90
DAAC	Thread	New 32-bit mode	sgi32_daac_r
DAAC	C++ or C	64-bit mode	sgi64_daac_cpp
DAAC	FORTRAN 77 or C	64-bit mode	sgi64_daac_f77
DAAC	Fortran 90 or C	64-bit mode	sgi64_daac_f90
DAAC	Thread	64-bit mode	sgi64_daac_r

\$PGSBIN is the bin directory of the appropriate SDP toolkit.

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 still six choices. 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.

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 bin directory in the environment variable PGSBIN. The second step is to source (run) the set up script in the appropriate bin directory. This step result in a number of other environment variables getting set that will be needed.

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

Assumptions:

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

To check set up a SDP Toolkit environment, execute the procedure steps that follow:

1. On the SGI, type **setenv PGSBIN *ToolkitPathname/bin***. The *ToolkitPathname* is the home directory of the particular SDP Toolkit version being used.
2. On the SGI, type **source \$PGSBIN/*FLAVOR*/pgs-dev-env.csh**. The FLAVOR is the particular toolkit being used. Refer to Table 10.1-2, SDP Toolkit Versions at the DAAC.

## 10.2 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. Running a SMF “compiler” on a message text file produces both the header and message files. These message text files should be part of the science software delivery to the DAAC. They typically have a .t file name extension.

Deliveries may come with a make file that compiles PGE source code and MCF files at the same time. In that case, the following procedure is optional.

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:

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

To check compile status message facility (SMF) files, execute the procedure steps that follow:

- 1 Login to the SGI.
- 2 Type **source \$PGSBIN/FLAVOR /pgs-dev-env.csh**.
  - PGSBIN is the full path to the SDP Toolkit bin directory.
  - FLAVOR is the specific toolkit use. The PGE documentation should specify which compiler is required to run the PGE. The appropriate FLAVOR will correspond to the SCF version of this compiler type. See Table 10.1-2, SDP Toolkit Versions at the DAAC, for a list of these subdirectories.
- 3 Change directory to the directory containing the SMF text files. The SMF text files will typically have .t file name extensions.
- 4 Type **\$PGSBIN/FLAVOR/smfcompile -f *textfile.t***
  - 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.

### 10.3 Building Science Software with 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 ECS.

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 ECS.

This procedure describes some general principals 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.

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

Assumptions:

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

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

- 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 *Processing Files Description* document Production Rules and the *Operations Guide*. 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.
- 2** Type **source \$PGSBIN/FLAVOR /pgs-dev-env.csh**.
  - PGSBIN is the full path to the SDP Toolkit bin directory.
  - FLAVOR is the specific toolkit use. The PGE documentation should specify which compiler is required to run the PGE. The appropriate FLAVOR will correspond to the SCF or DAAC FLAVOR of this compiler type depending on your requirements. See Table 10.2-2, SDP Toolkit Versions at the DAAC, for a list of these subdirectories
- 3** Examine and alter (if necessary) any make files. Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.
- 4** Verify that the messages facility (SMF) files and the header file(s) in the proper directory for building. See Section 10.2.
- 5** 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.
- 6** Build the software in accordance with instructions delivered. The 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.

This page intentionally left blank.

# 11. 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 ECS, 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.

Running a PGE in a Simulated SCF Environment is often called “Command line run of a PGE”. The Planning and Data Processing System (PDPS) and the Science Data Server are not involved in this run.

The procedure involves the following steps.

- Updating the Process Control File (PCF)
- Setting up the Environment for running PGE
- Running and profiling the PGE

## 11.1 Updating the Process Control File (PCF)

Process Control Files defines the input and output file locations and other control parameters for the PGE and is required to run the PGE. The process control files delivered to the DAACs for SSI&T were created and used at the SCFs. It will be necessary to change the path names specified in the PCF to reflect path names in the DAAC environment. The same may be true of file names if they get changed after delivery to the DAAC. Please note that this edited PCF is not used when running in PDPS. The system will automatically generate a PCF file.

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

Assumptions:

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

To update the PCF, execute the procedure steps that follow:

- 1 Login to the SGI.
- 2 Locate the PCF file of interest and run the Process Control File Checker on the PCF (see Section 9.6). This will verify that the delivered PCF is correct before editing.

- 3 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
    - ? MCF Files (Metadata Configuration File)
    - ? 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.
  
- 4 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|L1A.granule/MODIS/run/input|||1
 change /MODIS/run/input to the appropriate path name in the DAAC where the file L1A.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.
  
- 5 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

- If this entry is not present, then add it.
- 6 Again, run the Process Control File Checker on the PCF (see Section 9.6) to make sure that no errors were introduced during editing.

## 11.2 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:

1. The Process Control File (PCF) exists and has been tailored for the DAAC environment (Section 11.1).
2. The C shell or a derivative (*e.g.* T shell) is the current user shell.

To set up an environment for running the PGE, execute the procedure steps that follow:

- 2 Login to the SPR SGI.
- 3 Type **source \$PGSBIN/*FLAVOR* /pgs-dev-env.csh**.
  - PGSBIN is the full path to the SDP Toolkit bin directory.
  - FLAVOR is the specific toolkit use. The PGE documentation should specify which compiler is required to run the PGE. The appropriate FLAVOR will correspond to the SCF version of this compiler type. See Table 10.2-2, SDP Toolkit Versions at the DAAC, for a list of these subdirectories.
- 4 Type **setenv PGS\_PC\_INFO\_FILE *PCFpathname/PCFfilename***.
  - The *PCFpathname* is the full path name to the location of the Process Control File (PCF) to be associated with this PGE and *PCFfilename* is the file name of the PCF.
- 5 If necessary, set any other shell environment variables needed by the PGE by sourcing the appropriate scripts or setting them on the command line. Refer to documentation included in the delivery.

## 11.3 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.

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

Assumptions:

1. The PGE has been built successfully with the SCF version of the SDP Toolkit (Section 10.4).
2. The required SMF runtime message files have been produced and placed in the correct locations (Section 10.3).
3. The Process Control File (PCF) exists and has been tailored for the DAAC environment (Section 10.1).
4. The required environment for running the PGE has been set up (Section 11.1).
5. The required input files are available and accessible.

To run and profile the PGE, execute the procedure steps that follow:

- 1 In the SPR SGI window whose environmental variables were the set up in Section 11.2, change directory to *PGEbinPathname*, the directory containing the built PGE binary executable.
- 2 Type **`/usr/ecs/<mode>/CUSTOM/bin/DPS/EcDpPrRusage PGE.exe >& ResultsOut`**.
  - *PGE.exe* is the name given to the PGE binary executable or top level shell script used to run the PGE is the PGE is composed of more than one executable.
  - *ResultsOut* is the file name in which to capture the profiling results.
  - The EcDpPrRusage is the profiling program that outputs information about the runtime behavior of the PGE.
  - Note: Depending upon the PGE, it may take some time before the UNIX prompt returns.
- 3 Immediately following the return of the prompt, type **`echo $status`**.
  - 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.
- 4 Examine the contents of the *ResultsOut* file. These results will be needed when PGE is registered in the PDPS.

## 12. PGE Registration and Test Data Preparation

---

The integration of science software with the ECS 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 (input data files) need to be placed on the Science Data Server. These steps must be accomplished before the science software can be run and tested within the ECS. The procedures in this section describe how to register a new PGE with the ECS. This involves updating the PDPS database with information needed to plan, schedule, and run the PGE.

### 12.1 Science Metadata

In order to update the PDPS Database with PGE and ESDT metadata, the metadata must first be prepared in ODL files, one for the PGE itself and one for each ESDT associated with a particular PGE.

#### 12.1.1 PGE ODL Preparation

This section describes how to prepare PGE ODL files. It is assumed that the SSIT Manager is running (for reference, see section 6.2).

- 1 From the SSIT Manager, click on **T**ools -> **P**DPS Database -> **P**CF ODL Template. A xterm with title "SSIT: Science Metadata ODL Template Creation" will be displayed.
- 2 At the prompt **Configuration Filename (enter for default: *../cfg/EcDpAtCreatODLTemplate.CFG*)?**, press Return.
- 3 At the prompt **ECS mode of operations?**, type *<mode>*.
- 4 At the prompt **PCF (Process Control file) to generate template from (including full path)?**, type *FullPathToPCFFile*.
- 5 At the prompt **PGE name (max 10 characters)?**, type the name of the PGE that will be registered.
- 6 At the prompt **PGE version (max 5 characters)?**, type the version of the PGE that will be registered.
- 7 At the prompt **PGE Profile ID (0 for Null, max 99)?**, type a valid response.
- 8 At the prompt **Directory for output template file (including full path)?**, type the *full path to the directory for the output template file*.

- A templete PGE ODL will be generated in the form  
PGE\_PGEname#PGEversion#ProfileID.tpl.
- 9** Make a copy of this PGE Template ODL file using the following convention.
- Type **cp PGE\_PGEname#PGEversion#ProfileID.tpl  
PGE\_PGEname#PGEversion#ProfileID.odl.**
  - The PGE\_PGEname#PGEversion#ProfileID.tpl is the file name of the ODL template file referred to in created in step 9.
  - The PGE\_PGEname#PGEversion#ProfileID.odl is the file name of a copy which can be safely edited. This is the required file naming convention.
- 10** Edit the PGE ODL file created in step 10 by filling in the required information.
- Note that the ShortNames typed into this file must each have a corresponding PDPS ESDT metadata ODL file (sec. 12.1.2).
  - 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.
  - See Appendix C for an example of PCF and corresponding PGE ODL and ESDT ODL files.
- 11** Move the edited PGE ODL file to **/usr/ecs/<mode>/CUSTOM/data/DPS/ODL** on the AIT Sun.
- The executable that populates the PDPS database will look for the PGE ODL file in this directory.

### 12.1.2 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.

- 1** Determine ShortName for required ESDTs corresponding to a Logical Unit Number (LUN) in the PGE ODL file.

- 2 Determine if the desired ESDT ODL exist by looking for it in `/usr/ecs/<mode>/CUSTOM/data/DPS/ODL`. The PGE ODL file will have the following naming convention. `ESDT_ShortName#Version.odl`
  - *ShortName* is the ESDT's ShortName and *Version* is the ESDT version.
- 3 If the desired ESDT ODL does not exists, create one from the `ESDT_ODL.template` found in the directory `/usr/ecs/<mode>/CUSTOM/data/DPS`. Do not edit this file directly but make a copy of it to the copy it to the `/usr/ecs/<mode>/CUSTOM/data/DPS/ODL` directory using the following naming convention. `ESDT_ShortName#Version.odl`. Note: This file naming convention *must* be observed.
- 4 Edit this file and add the required metadata. 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 (see Section 12.1.1) must match the ShortNames in these files.
- 5 Repeat steps 1 through 4 for each ESDT required by a particular PGE. .

### 12.1.3 Updating the PDPS Database with Science Metadata

To update the PDPS Database with PGE/ESDT Metadata, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on **Tools -> PDPS Database -> SSIT Science Metadata Update**. A xterm with title "SSIT: Science Metadata Database Update" will be displayed.
- 2 At the prompt **Configuration Filename? (enter for default: `../..cfg/EcDpAtRegisterPGE.CFG`)**, press Return.
- 3 At the prompt **ECS mode of operation?**, type *mode*.
- 4 At the prompt **PGE name (max 10 characters)?**, type *PGEname*. This name must match the PGE name specified in Section 12.1.1.
- 5 At the prompt **PGE version (max 10 characters)?**, type *PGEversion*. This version must match the PGE version specified in step section 12.1.1.
- 6 At the prompt **PGE Profile ID (0-99, 0 means null)?**, type a valid response. This version must match the PGE version specified in step section 12.1.1.

The system will now update the PDPS database the information supplied in the PGE and ESDT ODL files.

## 12.2 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, SSIT Operational Metadata refers to PGE information, which 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.

Before running the SSIT Operational Metadata Update from the SSIT Manager, you must first update the PDPS with SSIT Science Metadata (see Section 12.1.3). In addition, to get initial PGE Performance data which will be entered into the GUI, you need to run the profiling utility, EcDpPrRusage on the PGE or have the information on profiling provided (see Section 11.2).

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

Assumptions:

1. The required UNIX environment variables have been set.
2. The Science metadata has been updated to the PDPS database for this PGE (Sections 12.1.3).

To update the SSIT version of the PDPS database with operational metadata, execute the steps that follow:

- 1 From the SSIT Manager, click on **T**ools -> **P**DPS Database -> **O**pnI Metadata Update. The PDPS/SSIT Operational Database GUI 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. In the subwindow labeled **PGE Versions**, click on the PGE version for that PGE. Select the up or down arrow in the **Profile ID** subwindow to select profile ID for that PGE. Finally, click on the button labeled **EDIT**.
  - The page tabs **PROFILE**, **RUNTIME**, and **ESDT** will change from gray (indicating disabled) to black (indicating enabled).
- 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.

- 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 (see Section 11.2).
  - 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 (see Section 11.2).
  - 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 (see Section 11.2).
  - In the field labeled **Swaps**, enter the integer number of page swaps from the PGE. This value should come from profiling the PGE (see Section 11.2).
  - In the field labeled **Page faults**, enter the integer number of page faults from the PGE. This value should come from profiling the PGE (see Section 11.2).
- 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 logfiles. 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.

## 12.3 Insertion of Data Granules

This section describes how to prepare and insert test data granules 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 Science Data Server, a Target MCF is needed (also known as an ASCII metadata ODL file or a .met file).

In isolation or stand-alone testing of a PGE, the needed inputs will not have been inserted by a previous PGE in the chain. This Insertion must be done manually. Section 12.3.3 then describes how to do the Insert. In this way, a dynamic data granule can be inserted to the Science Data Server as if a PGE had produced it.

### 12.3.1 Generating a Source Metadata Configuration File (MCF)

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

Assumptions:

1. The SSIT Manager is running.
2. ESDT's are installed onto the Science Data Server. (See section 18 on how to install ESDT's.)

To Generate the Metadata Configuration File (MCF) for the input and output ESDT's, execute the steps that follow. The MCF's for the output files are to be generated only for the purpose of comparison with the delivered MCF's. This way an SSIT operator can notice any metadata mismatch between the two files and notify the instrument team and ECS ESDT group. In actual run of the PGE, the system will create the MCF's for the output files.

- 1 From the SSIT Manager, click on **T**ools -> **P**DPS Database -> **G**et MCF. A xterm with title "SSIT: Acquire MCF" will be displayed.
- 2 At the prompt **Configuration Filename? (Enter for default: ../cfg/EcDpAtGetMCF.CFG)**, press **Return**.
- 3 At the prompt **ECS mode of operations?**, type *mode*.
- 4 At the prompt **ESDT Short Name of desired MCF?**, type *ESDT ShortName*.
- 5 At the prompt **ESDT Version of desired MCF?**, type *ESDTversion*. The *ESDTversion* is the version of the ESDT.
- 6 At the prompt **Directory to receive MCF? (must be full path)**, type full path to directory you wish to have the MCF written to.

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

In stand-alone or isolation testing of a PGE, the needed inputs 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 stand-alone PGE. This way a dynamic data granule can be inserted to the Science Data Server as if a PGE had produced it.

The following procedures or steps can be used to obtain .met files for stand-alone PGE runs.

- 1 Use the MCF file generated in the previous section as a template for the .met file. Information delivered with the DAP can be used to construct the required metadata.
- 2 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.
- 3 Have the granule ingested into ECS. Procedures for ingest are not described in this document.

### 12.3.3 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:

1. The ESDTs have been installed on the Data Server. (See section 5.3 on how to install ESDTs.)
2. The Target MCF for this data granule has been created for the insertion.

To insert a dynamic granule to the Data Server, execute the following steps:

- 1 From the SSIT Manager, click on **T**ools -> **P**DPS Database -> **I**nsert **T**est Dynamic. A xterm with title "SSIT: PGE Test Dynamic Input File Insertion" will be displayed.
- 2 At the prompt **C**onfiguration filename? (**e**nter for default: `../cfp/EcDpAtInsertTestFile.CFG`), press Return.
- 3 At the prompt **E**CS Mode of operations?, type *mode*.
- 4 At the prompt **E**SDT short name for the file(s) to insert?, type *ESDTShortName* of the corresponding to this granule to be inserted.
- 5 At the prompt **E**SDT Version for the file(s) to insert?, type in the *ESDT version*.
- 6 At the prompt **I**s there more than one data file to this Dynamic Granule (**Y** = Yes, **N** = No)? (**e**nter for default: **N**) If this is a multifile granule, press *Y* otherwise press *N*.

- 7 For a multifile granule, you will be prompted for a full path to the directory that contains the granules and corresponding .met file. For a single file granule, you will be prompted for the full path to the granule and the full path to the corresponding .met file.
  - The dynamic data granule will be inserted to the Data Server. The Data Server Universal Reference (UR) will be seen on the screen.

### 12.3.4 Inserting Static Data Granules to 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.

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

Assumptions:

1. The SSIT Manager is running.
2. The ESDTs have been installed on the Science Data Server.
3. The Target MCF (.met) for this data granule has been created for the insertion.

To Insert the static granule data file to the Science Data Server, execute the steps that follow:

- 1 From the SSIT Manager, click **T**ools -> **P**DPS Database -> **I**nsert Static. A xterm with title “SSIT: PGE Static Input File Insertion” will be displayed.
- 2 At the prompt **C**onfiguration filename? (enter for default: `../EcDpAtInsertStaticFile.CFG`), press **R**eturn.
- 3 At the prompt **E**CS mode of operations?, type *mode*.
- 4 At the prompt **E**SDT Short Name for the file(s) to insert?, type *ESDTShortName*.
- 5 At the prompt **E**SDT Version for the file(s) insert?, type the *version number*.
- 6 At the prompt **S**cience Group for Static file (one of {C, L, D, O} followed by up to a 4 digit number)?, type *ScienceGroupID*.
  - The 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#profileID.odl file (see Section 13.1.1).

- The Science Group ID must match what was edited into the PGE metadata ODL file for that PCF entry (Section 12.1.1).

- 7 At the 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, enter *N*. If there are more than one data files, type *Y*.
- 8 For a multifile granule, you will be prompted for a full path to the directory that contains the granules and corresponding .met file. For a single file granule, you will be prompted for the full path to the granule and the full path to the corresponding .met file.
- The static data granule will be inserted to the Data Server. The Data Server Universal Reference (UR) will be seen on the screen.

## 12.4. Placing Science Software Executable onto Science Data Server

In order to be able to run a PGE within the ECS 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.

### 12.4.1 Assembling a Science Software Executable Package

This section describes how to assemble a Science Software Executables Package (SSEP) and create a corresponding Target MCF.

In order to Insert a PGEEEXE tar file into the Science Data Server (Section 12.4.2), 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 PGEEEXE 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:

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

To create an SSEP, execute the steps that follow:

- 1 At the UNIX prompt on an AIT Sun, type **mkdir *SSEPpathname***. 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.
- 2 Copy the top-level shell script used to run the PGE, if any, compiled message files and the PGE executables to the directory created above.

- 3 At the UNIX prompt on the AIT Sun, type **tar cvf *SSEPfilename.tar FullPathToSSEPDirectory*/\***. The *SSEPfilename.tar* is the file name for the SSEP tar file. The file name extension *.tar* is recommended but not required. The *FullPathToSSEPDirectory* is the full path to the SSEP directory.
- 4 Create a corresponding *.met* file. The ShortName to use is PGEEEXE. This *.met* file can be created from the PGEEEXE MCF file or by using the template in Appendix C.

#### 12.4.2 Inserting Science Software Executable Package onto Science Data Server

Science software, like any other data that are managed in the ECS, 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 (SSEP) into the Data Server.

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

Assumptions:

1. The ESDT called PGEEEXE has been installed on the Science Data Server. (see section 5.3 for reference)
2. A Target MCF (*.met*) for this PGEEEXE tar file has been created for the Insert.
3. The PGEEEXE tar file has been created according to the procedures in Section 12.4.1.

To Insert the SSEP to the Science Data Server, execute the steps that follow:

- 1 From the SSIT Manager, click **Tools -> Data Server -> Insert EXE TAR**. A xterm with title "SSIT: PGE Executable Tar File Insertion" will be displayed.
- 2 At the prompt **Configuration filename? (enter for default: *../EcdpAtInsertExeTarFile.CFG*)**, press **Return**.
- 3 At the prompt **ECS mode of operations?**, type *mode*.
- 4 At the prompt **Name of PGE?**, type *PGENAME*. The *PGENAME* must match exactly the PGE name entered into the PDPS for this PGE (see Section 12.1.1).
- 5 At the prompt **Science software version of PGE?**, type *SSWversion*. The *SSWversion* is the version of the science software that is being inserted in this SSEP.
- 6 At the prompt **Staged filename to insert (including Full path)?**, type *FullPathToSSEPFile*.
- 7 At the prompt **Associated ASCII metadata filename to insert (including Full Path)? ( *pathname/SSEPFileName.met*)?**, type *FullPathToSSEPFileName.met*.

- 8** At the prompt **Top-level shell filename within tar file?**, type *TopLevelShellScript* if PGE uses a script or *ExecFileName* if only a single executable is used to run PGE. If successful, a universal reference (UR) will be returned.

This page intentionally left blank.

# 13. PGE Planning, Processing, and Product Retrieval

---

## 13.1 Using the Production Request Editor

When stand-alone tests (Run from the command line) have completed successfully and information about the PGE has been entered into the PDPS Database (through PGE registration), the PGE is ready to be run through the automated ECS PDPS environment.

To process Science data, a Production Request (PR) must be submitted to the ECS system. The Production Request Editor GUI accomplishes this function. Only one PR may be submitted at a time. A single PR is exploded by the PDPS into one or more jobs called Data Processing Requests (DPRs). The number of DPRs that are created for a single PR is determined by the number needed to cover the requested time interval, orbital extent and tile schema. Some PRs may only require one DPR.

### 13.1.1 Invoking the Production Request Editor

The Production Request Editor is invoked from a command line script. Once the Production Request Editor is invoked, it brings up a screen with five tabs at the top for selection (as shown in Fig. 13.1.1-1). The first tab is labeled “Planning”. Selection of this tab displays a list of four capabilities available for the PR Editor by selecting the other tabs at the top of the primary GUI screen: PR Edit, PR List, DPR View, and DPR List.

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

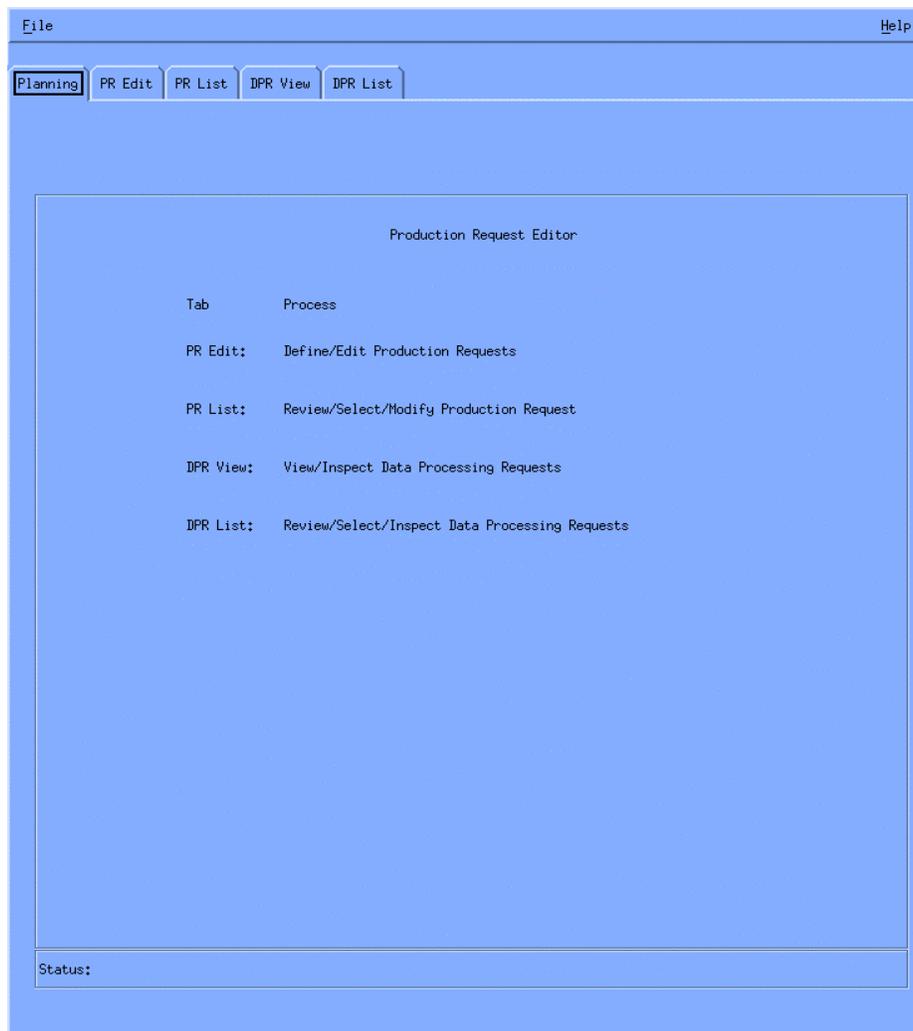
Assumptions:

1. The PGE has been registered in the PDPS Database.
2. The PGE has been successfully compiled and linked with the DAAC version of the SDP Toolkit.
3. The required servers are up and running.

To invoke the Production Request Editor GUI, execute the procedure steps that follow:

- 1 Login to the PLS host.
- 2 Type **dce\_login** *DCE\_user\_name* *DCE\_password*.
- 3 Change directory to `/usr/ecs/<mode>/CUSTOM/utilities` and type **EcPIPRE\_IFStart** *mode* **&**

- 4 In the Production Request Editor, click on one of the tabs PR Edit, PR List, DPR View, or DPR List corresponding to desired task.
  - To define a new Production Request or edit a Production Request, click on PR Edit. Proceed to Section 13.1.2.
  - To review or list a Production Request, click on PR List.
  - To view or inspect a Data Processing Request, click on View.
  - To review or inspect a Data Processing Request, click on List.
  
- 5 When tasks are completed in the Production Request Editor GUI, click on the **File** menu, and then choose **Exit**. Refer to Appendix A (Troubleshooting and General Investigation) if fail to bring up the Production Request Editor GUI.



**Figure 13.1.1-1. Production Request Editor (Planning) GUI**

### 13.1.2 Defining a New Production Request

A Production Request (PR) is a request for data production of granules between a start date/time and an end date/time for the time-based PGE (or a start orbit and an end orbit for the orbit-based PGE). A PR will explode into one or more Data Processing Requests (DPR) depending upon the time interval (for the time-based PGE) or orbit range (for the orbit-based PGE) involved. Each DPR corresponds to the execution of a PGE at one time. Therefore, a PR results in the execution of a PGE one or more times. Only one PGE is involved in a single Production Request.

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

Assumptions:

1. The Production Request Editor is displayed. (Section 13.1.1).
2. The PGE involved in the Production Request has been registered in the PDPS database (Section 12.1).

To define a new Production Request, execute the procedure steps that follow:

- 1 From the Production Request Editor GUI click on the **PR Edit** tab. The **PR Edit** page will be displayed as shown in Fig. 13.1.2-1.
- 2 In the field labeled **PR Name**, enter *NewPRName*.
- 3 The PGE for the Production Request must be selected from a list. To do this, click on the **PGE...** button. A GUI labeled **PGE Selection** will be displayed within which registered PGEs will be listed. The appropriate PGE can then be selected by clicking on it, then on the **OK** button. The selected PGE will then be used to populate **Satellite Name**, **Instrument Name**, **PGE Name**, **PGE Version** and **Profile ID** fields of the Production Request Editor GUI.
- 4 In the Production Request Editor GUI, a **Duration** option is selected automatically based on the PGE registered into the PDPS.
  - Two options are provided: **UTC Time** for time range.
  - **Orbit** for orbit number range.
- 5 In the case of UTC Time duration, enter *StartDate* and *StartTime* in fields labeled **Begin**, respectively. Enter *EndDate* and *EndTime* in fields labeled **End**, respectively. The time of the Production Request and should be entered in the mm/dd/yy and hh:mm:ss formats.
- 6 In the case of Orbit duration, enter *StartOrbit* and *EndOrbit* in fields labeled **From** and **To**, respectively.
- 7 When Production Request is complete, click on **File** menu and select **Save As...**
  - A GUI labeled **EcPIPREditor** will be displayed. In the field labeled **Selection**, enter a user-defined name to be assigned to the Production Request. Then

click on the **OK** button. A message box will be displayed stating “**Production Request Explosion into DPRs ok,  $n$  DPRs Generated**”, where  $n$  will be the number of DPRs (e.g. a 2-hr PR time will generate 24 DPRs for the 5-min processing period). Click on the **Ok** button.

- Refer to Appendix A (Troubleshooting and General Investigation) if fail to generate the DPRs.

The screenshot displays the PR Editor GUI with the following sections and fields:

- Production Request Identification:**
  - PR Name:
  - Origination Date:
  - PR Type:
  - Originator:
  - Priority:
- Request Definition:**
  - Satellite Name:
  - Instrument Name:
  - PGE Name:
  - PGE Version:
  - Profile Id:
  - PGE ...
  - PGE Parameters...
  - Metadata Checks...
  - Alternate Input Values..
- Duration:**
  - UTC Time:
  - Orbit:
  - Begin:
  - End:
  - From:
  - To:
  - Tile Id:
- Intermittent DPR:**
  - Skip:
  - Keep:
  - SkipFirst:
- Comment:**
- Status:**

**Figure 13.1.2-1. PR Editor GUI**

### 13.1.3 Listing Production Requests

This procedure describes how to view PRs that have already been defined (see Section 13.1.2). It assumes that the **PR List** tab has been selected from the Production Request Editor.

The information listed for each PR is:

- PR Name - The name assigned to the Production Request when it was defined.
- PGE ID - The name of the PGE involved in the PR.
- Priority - The priority (0 - 10) of the PR assigned when it was defined.
- StartTime - The start date and start time of the PR.
- EndTime - The end date and end time of the PR.
- Comment - Any comment that was entered when the PR was defined.

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

Assumptions:

1. The Production Request Editor is displayed. (Section 13.1.1).

To view Production Requests, execute the procedure steps that follow:

1. From the Production Request Editor GUI, click on the **PR List** tab. The PR List page will be displayed as shown in Fig. 13.1.3-1.



**Figure 13.1.3-1. PR List GUI**

### 13.1.4 Listing Data Processing Requests

This section describes how to view the data processing requests that have been defined in Section 13.1.2. Detailed procedures are given below.

- 1 From the PR Editor GUI, click on **DPR list** tab.
- 2 In the field labeled **PR Name** bring down a list of PR names by clicking on the appropriate PR name. Highlight the PR name that one needs so that it appears on the PR Name field.
- 3 Click the button labeled **Filter** to bring the all information of the PR down to the **Data Processing Requests** window where the following information of each DPR of the highlighted PR is displayed in one line:
  - DPR Id.PGE Id.
  - PR Name.Tile Id.

- Data Start Time (UTC).
  - Data Stop Time (UTC).
- 4 Click on the DPR of interest.
  - 5 Click on **File-Open** button in the PR Editor GUI.
  - 6 Click the **DPR View** tab and the following information will be displayed:
    - Data Processing Request Identification.
    - PGE ID and its parameters.
    - Request Data and Status.

## 13.2 Using the Production Planning Workbench

The Production Planner uses the Production Planning Workbench to create new production plans and display a planning timeline.

### 13.2.1 Using the Planning Workbench to Run One PGE

Once a PGE has been fully registered (Sections 12.1 and 12.2), its test data files have been inserted to the Science Data Server (Section 12.3), and a single Data Processing Request (DPR) has been generated (Section 13.1), the Planning Workbench can be used to plan for one execution run of a single PGE.

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

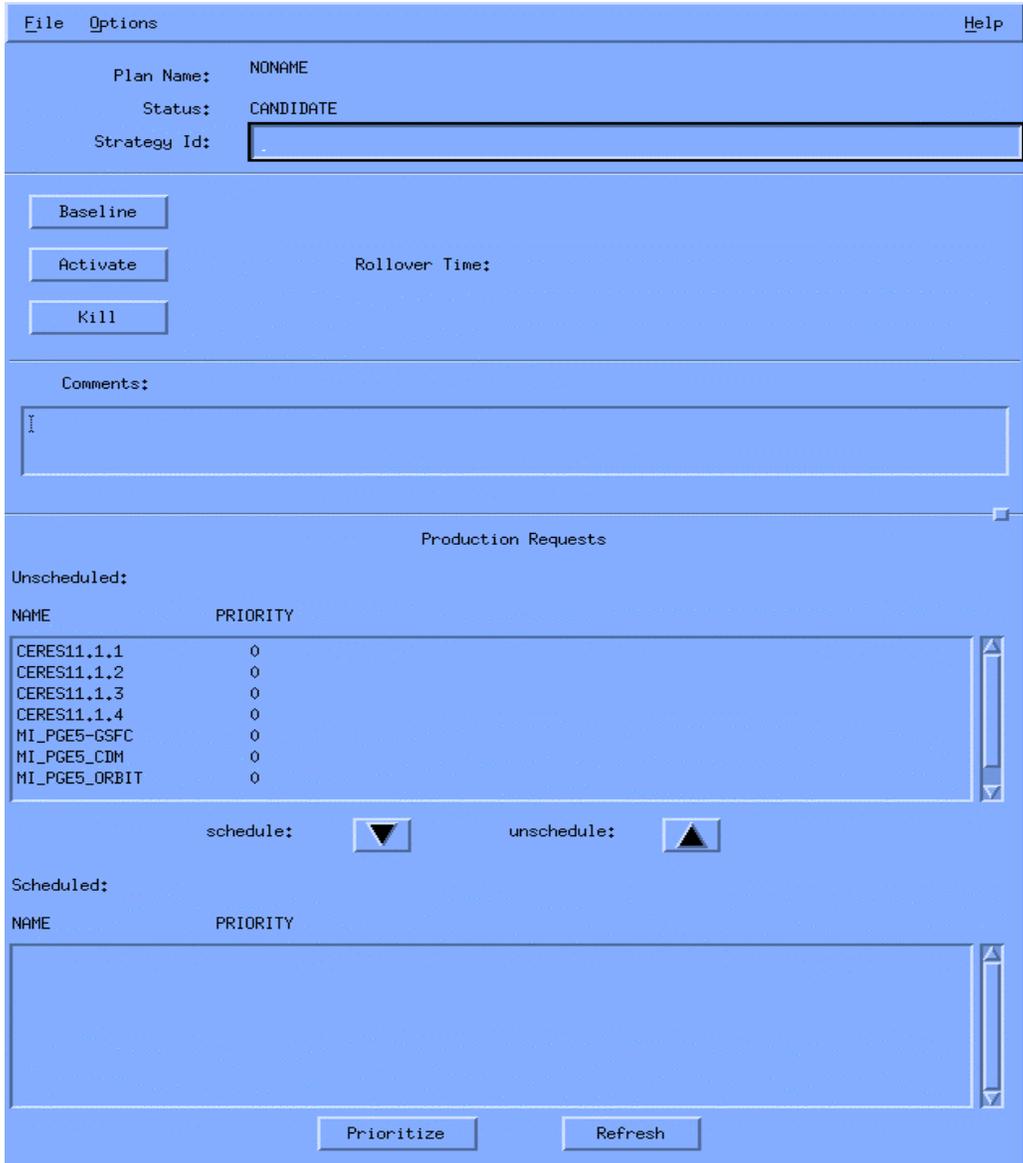
Assumptions:

1. The required servers of the ECS System are up and running.
2. A DPR has been generated successfully.

To use the Planning Workbench to run one PGE, execute the procedure steps that follow:

- 1 Login to the PLS host.
- 2 Type **dce\_login DCE\_user\_name DCE\_password**.
- 3 Change to the directory `/usr/ecs/<mode>/CUSTOM/utilities` directory.
- 4 Type **EcPIAllStart mode application\_id &**, where the `application_id` is a numerical number typically from 1 to 5. The Planning Workbench GUI will be appeared as shown in Fig. 13.2.1-1. Note any currently scheduled production requests.
- 5 Click on the **File->New** button to popup a New Plan GUI. Type a plan name in the Plan Names field and click **OK**.

- 6 In the Planning Workbench GUI, go to the subwindow labeled **Unscheduled** and highlight the Production Requests noted in step 4 and any additional Production Requests that you would like to add to the previously scheduled plan. Click on the **Schedule** button.
- 7 In the Planning Workbench GUI, click on the **Activate** button. A small GUI labeled **Saving Plan** will be displayed. Click on the **Save** button. A small GUI labeled **Confirm Activation Win** will be displayed. Click on the **Yes** button. Use the JobScape GUI to monitor production (Section 13.3).
- 8 To terminate the processes of Planning Workbench GUI, type *EcPISlayAll mode*.



**Figure 13.2.1-1. Planning Workbench GUI**

### 13.3 Monitoring Production

The progress of one or more PGEs running within the PDPS may be monitored. The COTS tool used for this purpose is AutoSys® by Atria Software. Each Data Processing Request results in three AutoSys jobs that are boxed together. An AutoSys job name follows the template:

*PGEname#Suffix*

where *PGEname* is replaced by the name of the PGE, and *Suffix* is a character indicating the job phase of the DPR. Refer to Table 13.3-1.

**Table 13.3-1. AutoSys Jobs for a DPR**

Job Name Suffix	Description
R	Resource allocation, Staging and Pre-processing
E	Execution of the PGE itself
P	Post-processing, De-staging and De-allocation

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

Assumptions:

1. The required servers of the ECS System are up and running.
2. A DPR has been scheduled successfully.

To monitor production, execute the procedure steps that follow:

- 1 Login to the DPS host.
- 2 Change directory to `/usr/ecs/<mode>/CUSTOM/utilities` and type `EcDpPrAutosysStart mode Autosys_Instance &`, where Autosys\_Instance is the instance of autosys. A Gui labeled **AutoSys** will be displayed.
- 3 In the AutoSys GUI, click on the **Ops Console** button. A GUI labeled **Job Activity Console** GUI will be displayed. The main subwindow of this GUI will contain a dynamically updated list of AutoSys jobs (three jobs make up one DPR) currently scheduled. To select a subset of the possible jobs that can be displayed, click on the **View** → **Select Jobs**. Under the label **Select by Name**, click on the square labeled **All jobs**, then click on the **OK** button.
  - DPRs will be listed in the column labeled **Job Name** and their statuses (e.g. SUCCESS) will be listed in the column labeled **Status**.
  - To view job status information for a particular DPR, click on a DPR. Below the main subwindow, available job status information will be displayed.
  - To view the existing event report, under the label **Reports**, click on the middle diamond labeled **Event**. The current event status for the selected DPR will be displayed in the subwindow labeled **Event Report**. Alternatively, the summary report for the selected DPR will be displayed by clicking on the upper diamond labeled **Summary**.
  - To view the job definition, under the label **Show**, click on the **Job Definition** tab. A GUI labeled **Job Definition** will be displayed. The selected DPR is shown in **Job Name**.

- Exit the **Job Activity Console** by clicking on the **Exit** button.

4 In the AutoSys GUI, click on the **JobScape** button.

- A GUI labeled **JobScape** will be displayed.
- The main subwindow of this GUI will contain a dynamically updated list of AutoSys jobs (three jobs make up one DPR) currently scheduled. The colors indicate the job statuses for DPRs and their jobs at present.
- There are eleven job statuses: **ACTIVATED, STARTING, RUNNING, SUCCESS, FAILURE, TERMINATED, RESTART, QUE\_WANT, ON\_ICE, OFF\_HOLD, and INACTIVE.** The color chart is on the left of GUI.
- To view job status information for a particular DPR (or job) in detail, click on a DPR (or job), then click on the **Job Console** button. A GUI labeled **Job Console** will be displayed. The information shown here is similar to that shown in **Ops Console** as mentioned above.
- To change the status of a job for a particular DPR, click on a job under that DPR, then press the right button of the mouse. Choose one of the functions in the lower part of the menu. For example, to hold a job, choose **On Hold**, then click the **Yes** button. **Suggestion:** For a scheduled DPR, hold all jobs for that DPR except the first one (Allocation). After the first job runs successfully, release the next job (Execution) by choosing **Off Hold**. Repeat until all jobs are done. Therefore, if a job fails, it can be fixed and rerun without interfering with the others. This is especially important for the Postprocessing job. If the PGE fails and the Postprocessing job is not **On Hold**, the DPR will have to be deleted and a new one created. This happens because PDPS will have deleted all references to the output granules in the PDPS database.
- Refer to Appendix A (Troubleshooting and General Investigation) if any job fails on AutoSys.

## 13.4 Using the Q/A Monitor

The Q/A Monitor allows the output products produced during a PGE run to be accessed and examined. Input test data granules and Production History files can be retrieved in the same manner. The Q/A Monitor retrieves output products based on the collection name (*i.e.* the ESdT) and time of Insertion to the Science Data Server.

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

Assumptions:

1. The required servers of the ECS System are up and running.
2. The desired output products have been successfully Inserted to the Science Data Server.

To use the Q/A Monitor, execute the procedure steps that follow:

- 1 Login to the PLS host.
- 2 Change to the `/usr/ecs/<mode>/CUSTOM/utilities` directory and type **EcDpPrQaMonitorGUIStart *mode application\_id* &**, where the `application_id` is a numerical number. The Q/A Monitor GUI will be displayed.
- 3 In the Q/A Monitor subwindow labeled **Data Types**, select an ESDT from the list.
- 4 Under the label **Data Granule Insert (mm/dd/yyyy)**, set the date range within which the search for granules of the ESDT selected will be conducted. The dates refer to date of granule Insert to the Science Data Server.
- 5 Click on the **Query** button. The results of the query will be displayed in the bottom window, labeled **Data Granules**.
- 6 In the subwindow labeled **Data Granules**, select one of the data granules listed.
- 7 To retrieve the data granule's Production History file, click on the **Retrieve ProdHistory** button. The Production History (PH) tar file corresponding to the selected data granule will be retrieved from the Science Data Server and placed on the local machine (a PLS Host) in the directory `/var/tmp`.
  - The PH can then be moved or copied manually from the `/var/tmp` directory to a user working directory for examination.
  - Only the PH file is retrieved with the **Retrieve ProdHistory** button.
- 8 To retrieve the data granule, click on the **Retrieve DataGranule** button.
  - The data granule selected will be retrieved from the Science Data Server and placed on the local machine (a PLS Host) in the directory `/var/tmp`.
  - A granule of any format (binary, ASCII, HDF, HDF-EOS) may be retrieved in this manner. Only HDF and HDF-EOS granules, however, may be further visualized using EOSView as described in the next steps.
- 9 To examine the data granule, click on the **Visualize data** tab. The **Visualize data** page will be displayed.
- 10 In the main subwindow on the **Visualize data** page, locate the file name of the granule retrieved in step 8 and click on it. The item selected will be highlighted and will appear in the **Selection** subwindow below.
- 11 Click on the **Visualize** button. This will invoke EOSView with the granule selected. The granule must be HDF or HDF-EOS format. See Sections 16.1 for use of EOSView.

## 14. File Comparison

---

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.

### 14.1 Using the GUI HDF File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. The GUI File Comparison Tool allows comparison of two HDF or HDF-EOS files.

To compare two HDF or HDF-EOS files, execute the procedure steps that follow:

- 1 Start the SSIT Manager. See section 6.
- 2 From the SSIT Manager, click on **the Tools -> Product Examination -> File Comparison -> HDF(GUI)**. The **HDF File Comparison Tool GUI** will be displayed.
- 3 In the HDF File Comparison Tool GUI, follow the Systems Description document and the Operations Manual. Both of these or their equivalent should be in the delivery.

### 14.2 Using the hdiff HDF File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. 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.

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

To compare two HDF or HDF-EOS files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on **the Tools -> Product Examination -> File Comparison -> HDF (hdiff)**. This will open a xterm window running hdiff.

- 2 At the prompt **Options for comparison?** , type in any desired options. To view a list of available options, type **-h** at the prompt.
- 3 Enter the full path and the file name of the two files to compare when prompted.
- 4 The differences in the files, if any, will be displayed.

Note: hdiff can also be invoked from command line. This done by executing the command `/usr/ecs/<mode>/CUSTOM/bin/DPS/hdiff filename1 filename2`

### 14.3 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.

To compare two ASCII files, execute the procedure steps that follow.

- 1 From the SSIT Manager, click on the **T**ools -> **P**roduct Examination **F**ile Comparison -> **A**SCII. A xterm window running *xdiff* will be displayed.
- 2 Enter the appropriate mode when prompted.
- 3 Enter the full path and the file name of the two files to compare when prompted.
- 4 Only sections of file in which there are differences will be displayed. A “bang” character (!) at the beginning of a line indicates that a difference was found.

Note: *xdiff* can also be invoked from command line. This done by executing the command `/usr/ecs/<mode >/CUSTOM/bin/DPS/xdiff filename1 filename2`

### 14.4 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 a SCF-supplied format specification.

To compare two binary files, execute the procedure steps that follow.

- 1 From the SSIT Manager, click on the **T**ools -> **P**roduct Examination -> **F**ile Comparison -> **B**inary. The **Binary File Difference Assistant tool GUI** will be displayed.
- 2 The template for the a code example for comparing binary code can be viewed by clicking on either the **Image** button or the **Structure** button located under the label **Compare Function**. Clicking on the **Image** button will display a code example for comparing binary files containing images. Clicking on the **Structure** button will display a code example for comparing binary files containing structures or records.
- 3 The template for the Driver code can be viewed by clicking on either the **Image** button or the **Structure** button located under the label **Driver**. Clicking on the **Image** button will display a code example for a driver invoking the compare function for binary files containing images. Clicking on the **Structure** button will display a code example for a driver invoking the compare function for binary files containing structures or records.
- 4 Click on the **Copy** button. A GUI labeled **Enter Unique ID** will be displayed. In the field labeled **Enter unique file identifier:** type *fileID*, 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\_*fileID*.c Compare function
    - DaacBinDiff\_*fileID*\_driver.c Driver
    - DaacBinDiff\_*fileID*.mak Makefile
  - **FORTTRAN**
    - DaacBinDiff\_*fileID*.f Compare function
    - DaacBinDiff\_*fileID*\_driver.f Driver
    - DaacBinDiff\_*fileID*.mak Makefile
  - **IDL:**
    - DaacBinDiff\_*fileID*.pro Compare function
    - DaacBinDiff\_*fileID*\_driver.pro Driver
    - DaacBinDiff\_*fileID*.sh Shell script

Note: The files will be copied into the directory from which the SSIT Manager is being run. These files are templates that can be used to assist SSIT personnel in designing customized code to perform the binary file comparison.

This page intentionally left blank.

# 15. Data Visualization

---

In order to fully ascertain the success of science software in producing scientifically valid data sets, the data need to be displayed in forms that convey the most information easily. Data visualization enables this to be done. This section describes the data visualization tools available at the DAACs during SSI&T and how to use them for detailed inspection of data sets produced by PGEs. Only some aspects of data visualization will be addressed in these procedures. For further information, see the references below.

The two visualization tools provided at the DAACs are EOSView and IDL. EOSView was developed by ECS and is a 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 exporting data as ASCII, jumping to a row, thumbnail-panning, colorization, zooming, plotting, animation, and flatfile output. However, it has color palette problem since HDF-EOS data does not come with its own color palette and EOSView uses available default color palette from the operating system. This causes some problem visualizing images by hiding or losing some of the data. EOSView also cannot export displayed images in commonly used graphics format such as JPEG, GIF and so on. Instead, displayed image has to be transferred as screen shots by third party software packages.

IDL (Interactive Data Language) 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 CSDTs (or formats) in addition to HDF. IDL has additional features including flexible input/output, custom colorization, plotting, scripting, raster-math, geographic registration, map projection transformation, and vector map overlay.

## 15.1 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. See Section 14.2 for how to inspect the science data. An HDF-EOS file is defined a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

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

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
2. EOSView has been properly installed and is accessible to the user.

To view product metadata with the EOSView tool, execute the procedure steps that follow:

- 1 a From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **E**OSView.
  - The EOSView GUI will be displayed.
- 1 b Alternately, if EOSView isn't available from the SSIT Manager GUI, invoke EOSView from the command-line.
  - Go to the proper area by typing `cd /usr/ecs/<mode>/CUSTOM/eosview`  
<RETURN>
  - Start EOSView by typing `EOSView` <RETURN>
- 2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **F**ile menu and select **O**pen.
  - The **F**ilter GUI will be displayed.
- 3 In the subwindow labeled **Filter**, enter full path name and file name wildcard template. For example, enter `/home/MyDirectory/MySubdirectory/*`.  
(Simply, locate your file from dialog box by entering directory path and choose the file by clicking the mouse button).
  - The `/home/MyDirectory/MySubdirectory/*` represents the location to the directory containing the HDF-EOS files to examine.
  - The asterisk (\*) is a wildcard template that represents all files in that directory; other wildcard templates can narrow the search further, *e.g.* `*.hdf`.
  - Use the **D**irectories field to further select the correct directory.
  - Files found matching the wildcard template in the chosen directory will be displayed in **F**iles sub-window.
- 4 In the **F**iles subwindow, click on the file name of the HDF-EOS file to examine. Then click on the **O**K button.
  - A GUI labeled **EOSView - MyOutputFile.hdf** will be displayed where `MyOutputFile.hdf` is the file name of the file chosen in step 3.
  - Be patient - this GUI may take some time to appear, particularly for large files.
  - Once displayed, a list of HDF objects will appear in the main window. If nothing is listed, it means that no HDF objects were found within the file.
- 5 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on an object listed for which metadata is to be inspected.
  - The object selected will be highlighted.
  - Do not double click on object since this will cause a **D**imension GUI to be displayed instead.
- 6 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **A**tttributes menu and select **G**lobal.
  - A GUI labeled **EOSView - Text Display** will be displayed.
  - The global metadata associated with the object selected (in step 5) will be displayed in a scrollable field.

- If instead, the message “Contains no Global Attributes” appears, then the selected object contains no global metadata.
- 7 Repeat steps 5 and 6 for each HDF object within the selected HDF-EOS file for which metadata is to be examined.
- 8 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **File** menu and select **Close**.
- The **EOSView - MyOutputFile.hdf** GUI will disappear.
  - Be patient - this GUI may take some time to disappear, particularly for large files.
- 9 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Exit**.
- The **EOSView - EOSView Main Window** GUI will disappear.

## 15.2 Viewing Product Data with the EOSView Tool

This procedure describes how to use the EOSView tool to inspect the science data in the HDF-EOS output file from a PGE. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

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

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
2. EOSView has been properly installed and is accessible to the user.

To view product data with the EOSView tool, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **E**OSView.
  - The EOSView GUI will be displayed.
- 2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **F**ile menu and select **O**pen.
  - The **F**ilter GUI will be displayed.
- 3 In the **F**ilter subwindow, enter full path name and file name wildcard template. For example, enter **/home/MyDirectory/MySubdirectory/\***.
  - The **/home/MyDirectory/MySubdirectory/\*** represents the location to the directory containing the HDF-EOS files to examine.
  - The asterisk (\*) is a wildcard template that represents all files in that directory; other wildcard templates can narrow the search further, *e.g.* **\*.hdf**.
  - Use the **D**irectories field to further select the correct directory.

- Files found matching the wildcard template in the chosen directory will be displayed in **Files** subwindow.
- 4 In the **Files** subwindow, click on the file name of the HDF-EOS file to examine. Then click on the **OK** button.
    - A GUI labeled **EOSView - MyOutputFile.hdf** will be displayed where *MyOutputFile.hdf* is the file name of the file chosen in step 3.
    - Be patient - this GUI may take some time to appear, particularly for large files.
    - Once displayed, a list of HDF objects will appear in the main window. If nothing is listed, it means that no HDF objects were found within the file.
  - 5 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an object listed for which data is to be inspected.
  - 6 Go to procedure depending upon the type of the object selected in step 5.
    - Proceed to Section 15.2.1 if object is an HDF Image.
    - Proceed to Section 15.2.2 if object is an HDF-EOS Grid.
    - Proceed to Section 15.2.3 if object is an HDF-EOS Swath.
    - Proceed to Section 15.2.4 if object is an HDF SDS.
    - Proceed to Section 15.2.5 if object is in another format.
  - 7 Repeat steps 5 and 6 for each HDF object within the selected HDF-EOS file for which data is to be examined.
  - 8 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **File** menu and select **Close**.
    - The **EOSView - MyOutputFile.hdf** GUI will disappear.
    - Be patient - this GUI may take some time to disappear, particularly for large files.
  - 9 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Exit**.

### 15.2.1 Viewing HDF Image Objects

This procedure describes how to use the EOSView tool to view science Images (typically, browse images) in the HDF-EOS output file from a PGE. See Section 15.2 for how to select an HDF Image object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Once an HDF Image is displayed, a number of options are available. These include colorization, zooming, panning, and animation. Each is described in the procedures below as an optional step.

EOSView will display an HDF Image in its referenced default color palette if the file contains one. If the display looks “blacked out”, it may mean that no default color palette is available or referenced by the Image object. In this case, a palette will have to be selected by the user.

Zooming allows both zoom in and zoom out according to a Bilinear Interpolation or Nearest Neighbor resampling method. Bilinear Interpolation involves averaging to produce a “smoothed”

display appropriate for gray scale band Images or derived geophysical parameter Images (e.g. temperature, vegetation index, albedo). Nearest Neighbor produces a non-smoothed display appropriate for derived thematic image-maps (e.g. land cover, masks).

Panning allows user to select the portion of a zoomed Image to be displayed.

Animation allows multiple Images to be displayed in an animated fashion in a number of modes. Stop-at-End mode allows Images to be displayed to the end just once. Continuous Run mode lets the animation run through the Images repeatedly and Bounce mode does a forward/reverse run through any set of Images repeatedly. All animation runs can be stopped at any time and then resumed in either the forward or reverse directions. The speed of an animation can be adjusted as well.

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

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF image (RIS8, RIS24, *i.e.* Browse data).
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 15.2.

To view an HDF-EOS Image object with the EOSView tool, execute the procedure steps that follow:

- 1** In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an Image object listed for which data is to be inspected.
  - A GUI labeled **EOSView - Image Display Window - MyImageObject** will be displayed where *MyImageObject* will be replaced by the name of the object selected as listed. For example, **EOSView - Image Display Window - Image [512x512] ref=2 (palette)**.
  - Be patient - this GUI may take some time to appear, particularly for large files.
- 2** 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**.
  - The selection of palette will not result in any change to the data in the file or to the object's default palette.
  - This selection may be repeated until the desired palette is chosen.
- 3** Optional zooming. In the GUI labeled **EOSView - Image Display Window - MyImageObject**, click on the **Zooming** menu, then select **Select** and then select one of the resampling

methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.

- The selection of re-sampling method and zoom will not result in any change to the data in the file.
  - The zooming options may be repeated as desired.
- 4 Optional panning while zooming. In the GUI labeled **EOSView - Image Display Window - MyImageObject**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates the portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
- The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
  - The panning will not result in any change to the data in the file.
  - The panning option may be repeated as desired.
- 5 To end the session with colorization, zooming, or panning, in the GUI labeled **EOSView - Image Display Window - MyImageObject**, click on the **File** menu and select **Close**.
- The **EOSView - Image Display Window - MyImageObject** GUI will disappear.
- 6 Optional animation. In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **Options** menu, then select **Animated images**.
- A GUI labeled **EOSView - Image Animation Window - MyOutputFile.hdf** will be displayed.
  - Be patient - this GUI may take some time to appear, particularly for large files.
  - Optionally, click on the **Palette** menu to select a palette.
  - Optionally, click on the **Options** menu and then select **Mode** to select how the animation is to be run. Choose **Stop at end**, **Continuous run**, or **Bounce**.
  - Click on the Stop button, denoted by the || symbol (center) to halt the animation.
  - Resume the animation by clicking on either the Forward Play (denoted by the >> symbol) or the Reverse Play (denoted by the << symbol).
  - There is also Forward Increment (>|) and Reverse Increment (|<) button.
  - The animation speed may be adjusted by moving the **Speed** slider in either the “+” or “-“ direction.
  - To end animation session, click on the **File** menu and then select **Close**.

### 15.2.2 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. See Section 15.2 for how to select an HDF-EOS Grid object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

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



## Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF-EOS Grid.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 15.2.

To view an HDF-EOS Grid object with the EOSView tool, execute the procedure steps that follow:

- 1** In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an Grid object listed for which data is to be inspected.
  - A GUI labeled **EOSView - Grid Select** will be displayed.
  - Be patient - this GUI may take some time to appear, particularly for large Grid objects.
  - Clicking on the **Grid Information** checkbox and then clicking on the **OK** button displays grid information about the selected Grid object such as X-Dimension value, Y-Dimension value, Upper Left Point values, and Lower Right Point values.
  - Clicking on the **Projection Information** checkbox and then clicking on the **OK** button displays projection information about the selected Grid object such as the Projection Name and the Zone Code.
  - Clicking on the **Dimensions** checkbox and then clicking on the **OK** button displays dimension information about the selected Grid object such as the Dimension Names and Sizes.
  - Clicking on the **Attributes** checkbox and then clicking on the **OK** button displays attribute information about the selected Grid object such as the attribute names and values.
- 2** In the GUI labeled **EOSView - Grid Select**, click on the **Data Fields** checkbox and then click on the **OK** button. Then double click on one of the data fields listed.
  - A GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge** will be displayed where *GridObjectName* will be replaced by the name of the Grid object selected in step 1.
- 3** In the GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge**, click on the checkboxes for both **YDim** and **XDim** and then click on the **OK** button.
  - A GUI labeled **MyDataField** will be displayed where *MyDataField* will be replaced by the name of the data field selected in step 2.
  - The data will be displayed in this GUI in the form of a table of values.

- 4 In the GUI labeled *MyDataField*, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
  - Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.
  - A GUI labeled **EOSView - Swath/Grid Image** will appear, displaying the data field in image form.
- 5 Optional colorization. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
  - The selection of palette will not result in any change to the data in the file or to the object's default palette.
  - This selection may be repeated until the desired palette is chosen.
- 6 Optional zooming. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
  - The selection of resampling method and zoom will not result in any change to the data in the file.
  - The zooming options may be repeated as desired.
- 7 Optional panning while zooming. In the GUI labeled **EOSView - Swath/Grid Image**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
  - The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
  - The panning will not result in any change to the data in the file.
  - The panning option may be repeated as desired.
- 8 Optional flatfile output. From the GUI *EOSView—Grid Select*, select **Grid Dimensions** and take note of the PixelsXTrack value (the number of elements in the Grid object's raster); you will need to remember this value in order to make later use of your output flatfile (since the flatfile by definition contains no structural metadata). In the GUI labeled *MyDataField*, click on the **File** menu and then select **Save**. Then select either **Binary** or **ASCII** flatfile type, and assign the output file a name.
  - This procedure will create an output flatfile (which may be quite large, depending on the size of the original Grid object!), but will not result in any change to the data in the original HDF-EOS file.
  - You will later need to use the IDL Tool (q.v.) to view or manipulate the output flatfile.
  - The flatfile output option may be repeated as desired.

- 9 To end the session with displaying Grid object, in the GUI labeled **EOSView - Swath/Grid**, click on the **File** menu and select **Close**.
  - The **EOSView - Swath/Grid** GUI will disappear.

### 15.2.3 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). See Section 15.2 for how to select an HDF-EOS Swath object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

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

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF-EOS Swath.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 15.2.

To view an HDF-EOS Swath object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on a Swath object listed for which data is to be inspected.
  - A GUI labeled **EOSView - Swath Select** will be displayed.
  - Be patient - this GUI may take some time to appear, particularly for large Swath objects.
  - Clicking on the **Dimensions** checkbox and then clicking on the **OK** button displays dimension information about the selected Swath object such as Dimension Names and Sizes.
  - Clicking on the **Geolocation Mappings** checkbox and then clicking on the **OK** button displays geolocation information about the selected Swath object such as the Geolocation Dimensions, Data Dimensions, Offsets, and Increments.
  - Clicking on the **Indexed Mappings** checkbox and then clicking on the **OK** button displays index mapping information about the selected Swath.
  - Clicking on the **Geolocation Fields** checkbox and then clicking on the **OK** button displays geolocation fields information about the selected Swath object.
  - Clicking on the **Attributes** checkbox and then clicking on the **OK** button displays attribute information about the selected Swath object.

- 2 In the GUI labeled **EOSView - Swath Select**, click on the **Data Fields** checkbox and then click on the **OK** button. Then double click on one of the data fields listed.
  - A GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge** will be displayed where *SwathObjectName* will be replaced by the name of the Swath object selected in step 1.
- 3 In the GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge**, click on the checkboxes for both **ScanLineTra** and **PixelsXtrac** and then click on the **OK** button.
  - A GUI labeled **MyDataField** will be displayed where *MyDataField* will be replaced by the name of the data field selected in step 2.
  - The data will be displayed in this GUI in the form of a table of values.
- 4 In the GUI labeled **MyDataField**, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
  - Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.
  - A GUI labeled **EOSView - Swath/Grid Image** will appear, displaying the data field in image form.
- 5 Optional colorization. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
  - The selection of palette will not result in any change to the data in the file or to the object's default palette.
  - This selection may be repeated until the desired palette is chosen.
- 6 Optional zooming. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
  - The selection of resampling method and zoom will not result in any change to the data in the file.
  - The zooming options may be repeated as desired.
- 7 Optional panning while zooming. In the GUI labeled **EOSView - Swath/Grid Image**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
  - The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
  - The panning will not result in any change to the data in the file.
  - The panning option may be repeated as desired.

- 8 Optional flatfile output. From the GUI *EOSView—Swath Select*, select **Swath Dimensions** and take note of the PixelsXTrack value (the number of elements in the Swath object's raster); you will need to remember this value in order to make later use of your output flatfile (since the flatfile by definition contains no structural metadata). In the GUI labeled *MyDataField*, click on the **File** menu and then select **Save**. Then select either **Binary** or **ASCII** flatfile type, and assign the output file a name.
- This procedure will create an output flatfile (which may be quite large, depending on the size of the original Swath object!), but will not result in any change to the data in the original HDF-EOS file.
  - You will later need to use the IDL Tool (q.v.) to view or manipulate the output flatfile.
  - The flatfile output option may be repeated as desired.
- 9 To end the session with displaying Swath object, in the GUI labeled **EOSView - Swath/Grid**, click on the **File** menu and select **Close**.
- The **EOSView - Swath/Grid** GUI will disappear.

#### 15.2.4 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. These are generally the science data and not browse images). See Section 15.2 for how to select an HDF SDS object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

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

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF SDS.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 15.2.

To view an HDF SDS object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on a SDS object listed for which data is to be inspected.
  - A GUI labeled **EOSView - Multi-Dimension SDS** will be displayed.
  - A number of checkboxes will be displayed, one for each of the dimensions in the selected SDS (there will be at least two, an X and a Y).

- Be patient - this GUI may take some time to appear, particularly for large SDS objects.
- 2 In the GUI labeled **EOSView - Multi-Dimension SDS**, click on two of the dimension checkboxes and then click on the **Table** button. Then double click on one of the data fields listed.
    - A GUI labeled **MySDS** will be displayed where *MySDS* will be replaced by the name of the SDS object selected in step 1.
  - 3 In the GUI labeled **MySDS**, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
    - Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.
    - A GUI labeled **EOSView - Image Display Window - MySDS** will appear, displaying the data field in image form.
  - 4 Optional colorization. In the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default, Greyscale, Antarctica, Rainbow, or World Colors**.
    - The selection of palette will not result in any change to the data in the file or to the object's default palette.
    - This selection may be repeated until the desired palette is chosen.
  - 5 Optional zooming. In the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
    - The selection of resampling method and zoom will not result in any change to the data in the file.
    - The zooming options may be repeated as desired.
  - 6 Optional panning while zooming. In the GUI labeled **EOSView - Image Display Window - MySDS**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
    - The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
    - The panning will not result in any change to the data in the file.
    - The panning option may be repeated as desired.
  - 7 To end the session with displaying Swath object, in the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **File** menu and select **Close**.
    - The **EOSView - Image Display Window - MySDS** GUI will disappear.

## 15.3 Viewing Product Data with the IDL Tool

This procedure describes 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 JPEG. Consult the IDL references for details on these other formats. See Section 15.2 for how to inspect the science data and metadata in an HDF-EOS file.

The major activities addresses here include creating an image display (Section 15.3.1), saving an image display (Section 15.3.2), creating a plot display (Section 15.3.3), and saving a plot display (Section 15.3.4).

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

Assumptions:

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

To view product data with the IDL tool, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools -> **P**roduct Examination -> **I**DL. A xterm (on the AIT Sun) will be displayed within which the IDL command interpreter will be run.
- 2 Go to the following procedures depending upon the activity to perform.
  - Proceed to Section 15.3.1 to create an image display.
  - Proceed to Section 15.3.2 to save an image display.
  - Proceed to Section 15.3.3 to create a plot display.
  - Proceed to Section 15.3.4 to save a plot display.
- 3 To end the IDL session, close any display windows remaining, then at the IDL prompt type **quit**.

### 15.3.1 Creating an Image Display Using IDL

This procedure describes how to use the IDL Tool to create an image display. The next procedure, Section 15.3.2, describes how to save an image display (once created) to either a data file or a graphic file. For creating and saving plot displays, see Sections 15.3.3 and 15.3.4, respectively.

The Activity Checklist table that follows provides an overview of the process for creating an image display using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four

(Section) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

**Table 15.3.1-1. Creating an Image Display Using IDL - Activity Checklist**

Order	Role	Task	Section	Complete?
<b>Binary Data:</b>				
1	SSI&T	Open binary data input file.	(I) 16.3.1	
2	SSI&T	Create an image array for the data.	(I) 16.3.1	
3	SSI&T	Read binary data into image array.	(I) 16.3.1	
4	SSI&T	Display the image.	(I) 16.3.1	
5	SSI&T	Load a color table.	(I) 16.3.1	
6	SSI&T	Close the input file.	(I) 16.3.1	
<b>ASCII Data:</b>				
1	SSI&T	Open ASCII data input file.	(I) 16.3.1	
2	SSI&T	Create an image array for the data.	(I) 16.3.1	
3	SSI&T	Read ASCII data into image array.	(I) 16.3.1	
4	SSI&T	Display the image.	(I) 16.3.1	
5	SSI&T	Load a color table.	(I) 16.3.1	
6	SSI&T	Close the input file.	(I) 16.3.1	
<b>JPEG Data:</b>				
1	SSI&T	Read JPEG formatted file into image array.	(I) 16.3.1	
2	SSI&T	Load a color table.	(I) 16.3.1	
3	SSI&T	Display the image.	(I) 16.3.1	

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

Assumptions:

1. IDL is running (see Section 15.3).
2. The PGE output file to be examined is of an IDL-supported type/format (if in doubt, consult the *IDL Reference Guide*).
3. For binary files, data is assumed to be 8-bit characters. Follow the steps listed under **Binary Data**.
4. For ASCII files, data is assumed to be comma-delimited characters. Follow the steps listed under **ASCII Data**.
5. For JPEG (Joint Photographic Expert Group) formatted files, dimension information is carried in the file header and therefore it does not need to be known. Follow the steps listed under **JPEG Data**.

To create an image display using IDL, execute the procedure steps that follow:

## Binary Data:

- 1 At the IDL prompt, type **OPENR,1,('MyBinaryFilename'), .**
  - The *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.
  
- 2 At the IDL prompt, type **MyImage=BYTARR(dimx,dimy), .**
  - The *MyImage* is the name to be given to the image once created.
  - The *dimx* and *dimy* are respectively the X and Y dimensions of the input data.
  
- 3 At the IDL prompt, type **READU,1,MyImage, .**
  
- 4 At the IDL prompt, type **TV,MyImage, .**
  - The image, *MyImage*, should then be displayed.
  - Alternatively, type **TV,MyImage,offsetx,offsety, .** where *offsetx* and *offsety* are respectively the element and line offsets from *MyImage*'s origin (because of the way IDL defines an image's origin, you may need to use negative values for these offsets).
  
- 5 At the IDL prompt, type **LOADCT,3, .**
  - This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.
  
- 6 At the IDL prompt, type **CLOSE,1, .**
  - This closes logical unit 1.
  - Always close logical units or an error will result the next time an access is attempted.

## ASCII Data:

- 1 At the IDL prompt, type **OPENR,1,('MyASCIIfilename'), .**
  - The *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 **1** is the logical unit number.
- 2 At the IDL prompt, type **MyImage=BYTARR(dimx,dimy), .**
  - The *MyImage* is the name to be given to the image once created.
  - The *dimx* and *dimy* are respectively the X and Y dimensions of the input data.
- 3 At the IDL prompt, type **READF,1,MyImage,**
- 4 At the IDL prompt, type **TV,MyImage, .**
  - The image, *MyImage*, should then be displayed.
  - Alternatively, type **TV,MyImage,offsetx,offsety,** where *offsetx* and *offsety* are respectively the element and line offsets from *MyImage*'s origin (because of the way IDL defines an image's origin, you may need to use negative values for these offsets).
- 5 At the IDL prompt, type **LOADCT,3, .**
  - This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.
- 6 At the IDL prompt, type **CLOSE,1, .**
  - This closes logical unit 1.
  - Always close logical units or an error will result the next time an access is attempted.

## JPEG Data:

- 1 At the IDL prompt, type **READ\_JPEG,"MyJPEGfilename.jpg",MyImage, .**
  - The *MyJPEGfilename* is the full path name and file name of the JPEG formatted data file.
  - The double quotes (") must be included around the path/file name.
  - The *MyImage* is the name to be given to the image created.
- 2 At the IDL prompt, type **TVLCT,r,g,b, .**
  - Note that r,g,b color table syntax is used for most formatted file types in IDL.
- 3 At the IDL prompt, type **TV,MyImage, .**
  - The image, *MyImage*, should then be displayed.

### 15.3.2 Saving an Image Display Using IDL

This procedure describes how to use the IDL Tool to save an image display. The previous procedure, Section 15.3.1, describes how to create an image display. For creating and saving plot displays, see Sections 15.3.3 and 15.3.4, respectively.

The Activity Checklist table that follows provides an overview of the process for saving an image display using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

**Table 15.3.2-1. Saving an Image Display Using IDL - Activity Checklist**

Order	Role	Task	Section	Complete?
<b>Binary Data:</b>				
1	SSI&T	Open binary data output file.	(I) 15.3.2	
2	SSI&T	Write the output file.	(I) 15.3.2	
3	SSI&T	Close the output file.	(I) 15.3.2	

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

Assumptions:

1. IDL is running (see Section 15.3).
2. The image display is to be saved in a binary (8-bit) or ASCII (comma-delimited characters) format. For other output formats, consult the *IDL Reference Guide*.
3. For binary file option, data will be output as 8-bit characters. Follow the steps listed under **Binary Data**.
4. For ASCII file option, data will be output as comma-delimited characters. Follow the steps listed under **ASCII Data**.
5. For JPEG (Joint Photographic Expert Group) file option, data will be output as a JPEG-formatted file (lossy-compressed). Follow the steps listed under **JPEG Data**.

To save an image display using IDL, execute the procedure steps that follow:

#### **Binary Data:**

- 1 At the IDL prompt, type **OPENW,1,('MyBinaryFilename.bin'), .**
  - The *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.
- 2** At the IDL prompt, type **WRITEU,1,MyImage, .**
- The *MyImage* is the name of the image to save.
- 3** At the IDL prompt, type **CLOSE,1, .**
- This closes logical unit 1.
  - Always close logical units or an error will result the next time an access is attempted.

### ASCII Data:

- 1** At the IDL prompt, type **OPENW,1,('MyASCIIfilename.asc'), .**
- The *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 **1** is the logical unit number.
- 2** At the IDL prompt, type **PRINTF,1,MyImage, .**
- The *MyImage* is the name of the image to save.
- 3** At the IDL prompt, type **CLOSE,1, .**
- This closes logical unit 1.
  - Always close logical units or an error will result the next time an access is attempted.

### JPEG Data:

- 1** At the IDL prompt, type **WRITE\_JPEG,"MyJPEGfilename.jpg",MyImage, .**
- The *MyJPEGfilename.jpg* is the full path name and file name of the JPEG data file to write out.

### 15.3.3 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.

#### Setting 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],.**
- The *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 will then be displayed to the screen.

### Setting 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',.**

- The *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 will then be displayed to the screen.

### 15.3.4 Saving a Plot Display Using IDL

#### Saving a displayed plot to a permanent file:

1 At the IDL prompt, type **MyPlotDisplay=SURFACE,MyPlot,AX=80,AZ=20,.**

- The *MyPlotDisplay* is session name for the displayed plot of *MyPlot*.
- The *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',.**

- The *MyPlotDisplay* is the session name of the plot display.
- The *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).

### 15.3.5 Raster Math Fundamentals Using IDL

Most of this subject is covered in the IDL Manuals; some exceptions are described below.

#### Putting Raster Math results to a temporary display:

The following steps assume that you have already created two IDL session images, say “imageA” and “imageB”, having the same raster size (same number of lines and elements).

1 At the IDL prompt, type **imageC=imageA+imageB-14.8.**

- The *imageC* is session name for the result of the Raster Math shown above. The operation indicated is performed on each pixel.
- IDL also supports other Raster Math operators—multiplication (\*), division (/), exponent (^).

2 At the IDL prompt, type **TV, imageC, .**

- This displays the session image imageC, which can then be saved to a permanent file, etc.

3 Alternatively (to steps 1 and 2), at the IDL prompt, type **TV,imageA\*7.4/(imageB^8.2+1), .**

- This shortcut displays the result of the indicated operation, but does not give you the option to either print the result from a session image or to save it as a permanent file. It may be useful (since your IDL session has a limited capacity for holding session images) if you are just experimenting with different Raster Math operations and do not intend to print or save your results.

## 15.4 Raster Mapping Fundamentals

This procedure describes how to use the IDL Tool to perform basic raster mapping functions. These are two-dimensional spatial functions involving map projections, rather than surface modeling (also called “2.5D”—for which see Plotting section and consult IDL manuals), volumetric modeling (also called “3D”—for which consult the IDL manuals), or two-dimensional spectral functions (for which see Raster Math section and consult the IDL manuals). Note that the pixel-level effects of Raster Mapping functions on an image are typically non-reversible; you can change an image’s map projection from Projection A to Projection B and back again, but you won’t get exactly the same image you started with (hint--you should make a back-up copy of your original image before engaging in Raster Mapping).

The Activity Checklist table that follows provides an overview of the process for raster mapping fundamentals using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

**Table 15.4-1. Raster Mapping Fundamentals - Activity Checklist (1 of 2)**

Order	Role	Task	Section	Complete?
<b>Global Data Set Image:</b>				
1	SSI&T	Display the global data set image.	(I) 15.4	
2	SSI&T	Set the map projection.	(I) 15.4	
3	SSI&T	Create a new image using map projection.	(I) 15.4	
4	SSI&T	Display new image.	(I) 15.4	
5	SSI&T	Optionally, overlay Lat/Lon grid.	(I) 15.4	
6	SSI&T	Optionally, overlay world coastlines.	(I) 15.4	

**Table 15.4-1. Raster Mapping Fundamentals - Activity Checklist (2 of 2)**

Order	Role	Task	Section	Complete?
<b>Sub-Global Data Set Image:</b>				
1	SSI&T	Display the sub-global data set image.	(I) 15.4	
2	SSI&T	Set the map projection with limits.	(I) 15.4	
3	SSI&T	Create a new image within limits using map projection.	(I) 15.4	
4	SSI&T	Display new image.	(I) 15.4	
5	SSI&T	Optionally, overlay Lat/Lon grid.	(I) 15.4	
6	SSI&T	Optionally, overlay world coastlines.	(I) 15.4	

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

Assumptions:

1. IDL is running (see Section 15.3).
2. For a global data set image, one having coordinates defined from -180 to 180 degrees East Longitude and 90 to -90 degrees North Latitude, follow the steps listed under **Global Data Set Image**.
3. For a sub-global data set image, one having coordinates defined for subintervals of longitude and latitude (*e.g.* from -88 to -77 degrees East Longitude and 23 to 32 degrees North Latitude), follow the steps listed under **Sub-Global Data Set Image**.

To perform basic raster mapping for the cases listed above using IDL, execute the procedure steps that follow:

**Global Data Set Image:**

- 1 At the IDL prompt, type **TV,MyImage, .**
  - The **MyImage** is the image name of the global image data set.
  - The image, **MyImage**, should then be displayed.
- 2 At the IDL prompt, type **MAP\_SET,/ORTHOGRAPHIC, .**
  - IDL also supports other map projections. Refer to *IDL Reference Guide*.
- 3 At the IDL prompt, type **MyNewImage=MAP\_IMAGE(MyImage,startx,starty,/BILIN), .**
  - The **MyNewImage** is the name to assign to the resulting image.
  - The **MyImage** is the name of the original global image data set.
  - IDL also supports other resampling methods besides Bilinear Interpolation (**/BILIN**). Refer to *IDL Reference Guide*.
- 4 At the IDL prompt, type **TV,MyNewImage,startx,starty, .**

- The image *MyNewImage* should then be displayed.
- 5 Optional overlay Lat/Lon. At the IDL prompt, type **MAP\_GRID**, .
- This overlays Lat/Lon graticule onto *MyNewImage*.
- 6 Optional overlay world coastlines. At the IDL prompt, type **MAP\_CONTINENTS**, .
- This overlays world coastlines onto *MyNewImage*.
  - Note that the IDL world coastline vector file is itself approximate; the match between this vector coastline and your image's own coastline will therefore also be approximate, but the potential mismatch will usually be too small to notice on a global display.

### Sub-Global Data Set Image:

- 1 At the IDL prompt, type **TV,MyImage**, .
- The *MyImage* is the image name of the sub-global image data set.
  - The image, *MyImage*, should then be displayed.
- 2 At the IDL prompt, type **MAP\_SET,/MERCATOR,LIMIT=[lat1,lon1,lat2,lon2]**, .
- The *lat1*, *lon1*, *lat2*, and *lon2* specify the latitude and longitude intervals of the sub-global image data set. For example, type **MAP\_SET,/MERCATOR,LIMIT=[23,-88,32,-77]**, .
- 3 At the IDL prompt, type **MyNewImage=MAP\_IMAGE(MyImage,startx,starty,/BILIN,LATMIN=lat1,LATMAX=lat2,LONMIN=lon1,LONMAX=lon2)**, .
- The *MyNewImage* is the name to assign to the resulting image.
  - The *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.
  - IDL also supports other resampling methods besides Bilinear Interpolation (*/BILIN*). Refer to *IDL Reference Guide*.
- 4 At the IDL prompt, type **TV,MyNewImage,startx,starty**, .
- The image *MyNewImage* should then be displayed.
- 5 Optional overlay Lat/Lon. At the IDL prompt, type **MAP\_GRID**, .
- This overlays Lat/Lon graticule onto *MyNewImage*.
- 6 Optional overlay world coastlines. At the IDL prompt, type **MAP\_CONTINENTS**, .
- This overlays world coastlines onto *MyNewImage*.
  - Note that the IDL world coastline vector file is itself approximate; the match between this vector coastline and your image's own coastline will therefore also be approximate, but the potential mismatch will usually be too small to notice on

a continental display. If your display is subcontinental, you may observe a noticeable mismatch between the vector and image coastlines—at this level of detail the approximate nature of the IDL vector file becomes noticeable, and the difference (if any) between underlying Earth Model (ellipsoid or horizontal datum) of the vector file and your map-projected image can add a substantial (up to 1+ km) additional mismatch to the display.

This page intentionally left blank.

## 16. Inserting a Science Software Archive Package

---

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).

What follows is a list of items in the SSAP (taken from the DID205). 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 don't currently have official names.

Other files:

A change log created by the SSAP GUI to track changes to the SSAP.

## 16.1 Inserting a 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.

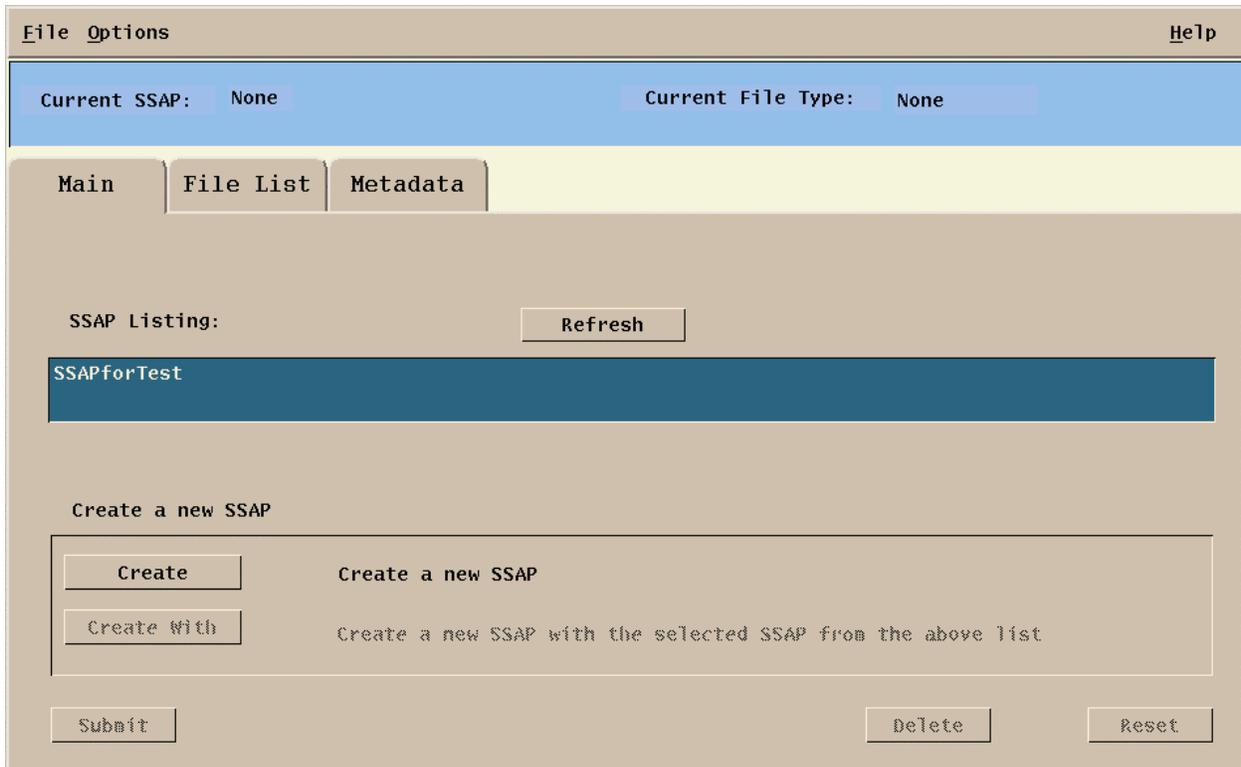
Assumptions:

1. All required Servers are up running.
2. SSI&T is up and running. (See section 6 how to bring up SSIT Manager)
3. The C shell (or a derivative) is the current command shell.

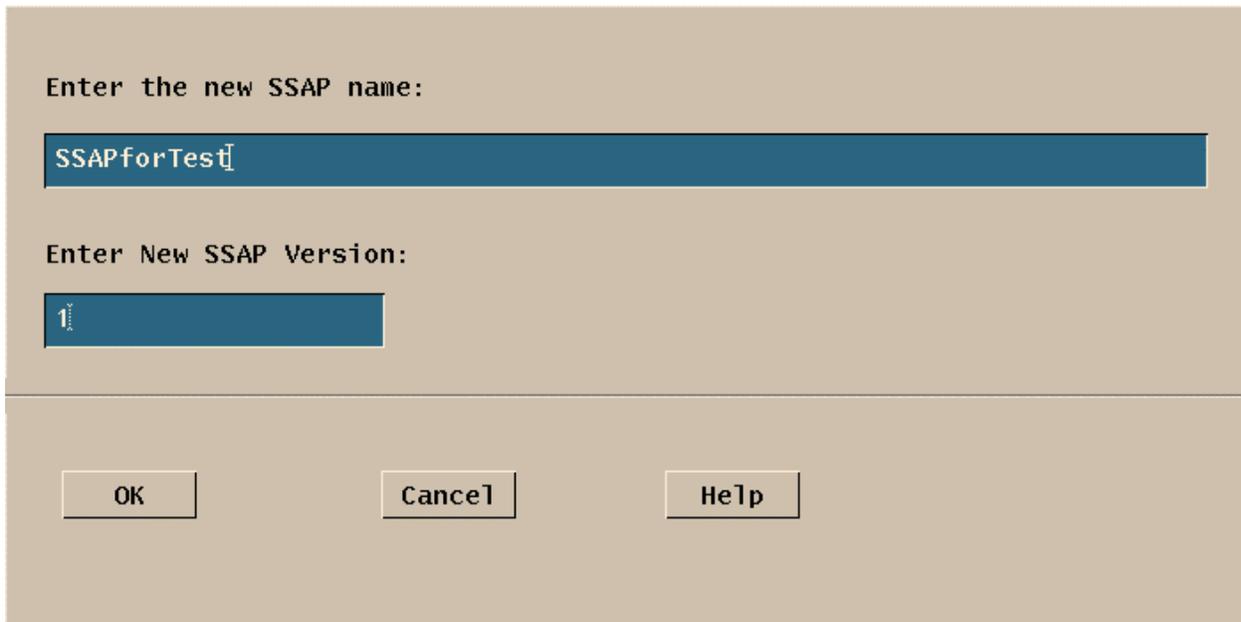
FORCHECK is available only on the AIT Suns.

To create the SSAP, execute the procedure steps that follow:

- 1 If not already on an AIT Sun, log onto one from your machine.
- 2 Launch the SSIT Manager. (see section 6).
- 3 From the SSIT Manager choose *Tools* menu and then *Data Server* submenu. Choose *SSAP Editor*.
  - The GUI starts (Fig. 16.1-1). 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 (Fig. 16.1-2). 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 the SSAP version in the second field. Note that version has a limit of 20 characters.

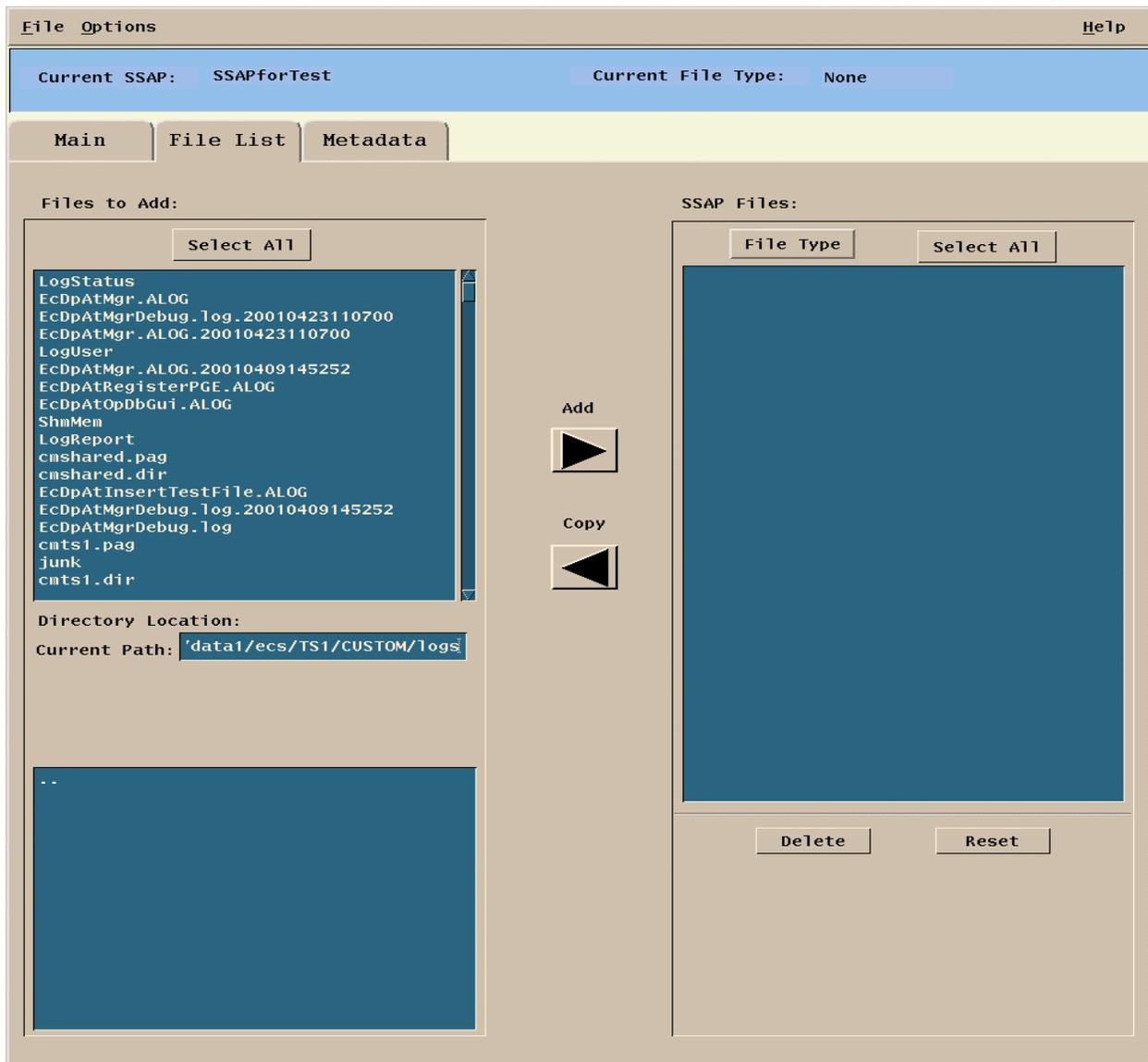


**Figure 16.1-1. SSAP Main GUI**



**Figure 16.1-2. Window to define new SSAP**

- 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 (Fig. 16.1-3) displays files in the local directory to the left and files in the selected SSAP component to the right. On the bottom left is 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.



**Figure 16.1-3. File list tab for file manipulation in defining new SSAP**

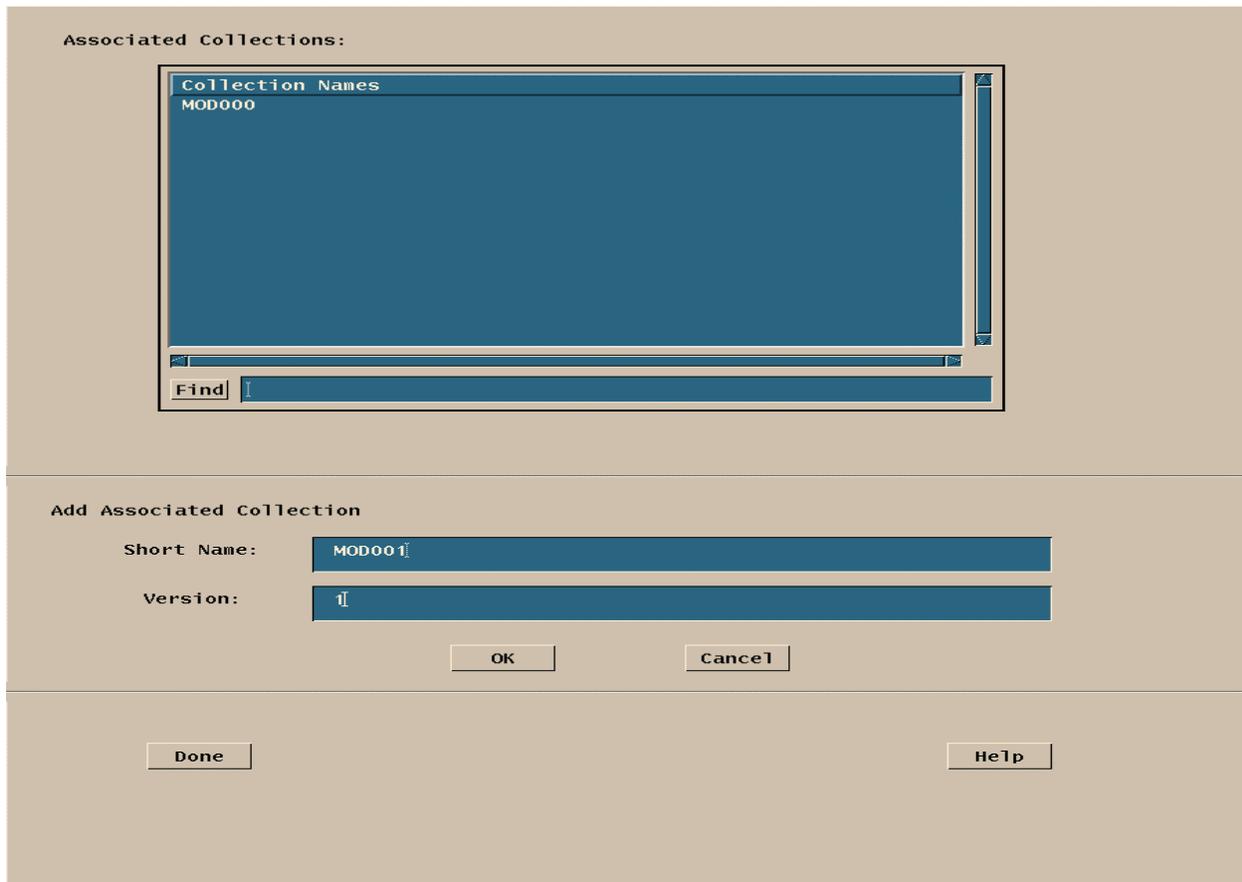
- 9 Click on the *File Type* button to select the SSAP component to manipulate.
- 10 Choose one of the menu items.

- 11 Select a file (or files) from the left window to add to the component.
- 12 Click the *Add Arrow* 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 (Fig. 16.1-4) 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.

The screenshot shows the 'File Options' dialog box with the 'Metadata' tab selected. The 'Current SSAP' is 'SSAPforTest' and the 'Current File Type' is 'None'. The 'Metadata' tab is active, showing two sections: 'Algorithm Information' and 'PGE Information'. In the 'Algorithm Information' section, the 'Algorithm Package Name' is 'SSAPforTest', 'Algorithm Package Version' is '002', 'Maturity Code' is 'Pre-launch', 'Delivery Purpose' is 'Initial Delivery', and 'Acceptance Date' is '01 / 01 / 1997'. In the 'PGE Information' section, the 'PGE Name' is '<pge name>', 'PGE ID' is '<pge id>', 'PGE Function' is '<pge function>', 'PGE Version' is '<pge versi', 'SW Version' is '<SW version>', 'PGE Date Last Modified' is '01 / 01 / 1997', and 'SW Date Last Modified' is '07 / 27 / 2000'. At the bottom, there are buttons for 'Edit Assoc Collections', 'Save', 'Reset', and 'OK'.

**Figure 16.1-4. Metadata tab to enter metadata for new SSAP**

- 14 Click the *Edit Assoc. Collections* button.
- The Edit Associated Collections window (Fig. 16.1-5) displays a list of associated collections and fields for the entry of new ShortNames and Versions (which make up an Associated Collection).



**Figure 16.1-5. The Edit Associated Window**

- 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.

# 17. Updating a Science Software Archive Package

---

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, see Section 16, “SSAP insert.”

## 17.1 Updating a 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:

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

FORCHECK is available only on the AIT Suns.

To update the SSAP, execute the procedure steps that follow:

- 1 If not already on an AIT Sun, log into one from your machine.
- 2 Launch the SSIT Manager. (See section 6).
- 3 From the SSIT Manager choose Tools menu and then Data Server submenu. Choose SSAP Editor.
  - 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.
- 4 Click on the Metadata tab to update the SSAP.
  - The Metadata Tab displays the metadata for the SSAP (Fig.17.1-1). 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.

The screenshot shows a window titled "File Options" with a "Help" button in the top right corner. Below the title bar, there are two status fields: "Current SSAP: SSAPforTest" and "Current File Type: None". The window has three tabs: "Main", "File List", and "Metadata", with "Main" currently selected. The main content area is divided into two sections: "Algorithm Information" and "PGE Information".

**Algorithm Information:**

- Algorithm Package Name: [SSAPforTest]
- Algorithm Package Version: [003]
- Maturity Code: [Operational]
- Delivery Purpose: [Second Delivery]
- Acceptance Date: [01 / 01 / 1998] with up/down arrow buttons and an OK button.

**PGE Information:**

- PGE Name: [BTS]
- PGE ID: [1]
- PGE Function: [Create temperature profile]
- PGE Version: [001]
- SW Version: [2.40]
- PGE Date Last Modified: [01 / 01 / 1998] with up/down arrow buttons and an OK button.
- SW Date Last Modified: [07 / 27 / 2000] with up/down arrow buttons and an OK button.

At the bottom of the window, there are three buttons: "Edit Assoc Collections", "Save", and "Reset".

**Figure 17.1-1. Metadata window for updating metadata**

- 5 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).

- 6** Update any other fields that you wish to change. You can even add a new Associated Collection by clicking on the Assoc. Collection button and following the steps described in creating an SSAP.
- 7** Before you leave the Metadata tab, click Save to save the updated metadata.
- 8** 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 is 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 Arrow button to add the files. They will appear in the right window because they are now part of that SSAP Component.
- 13** Click Main to get back to the Main tab.
- 14** 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.

This page intentionally left blank.

# 18. ESDT Management

---

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

Entries are made in the SDSRV database to define the ESDT, each of its attributes, and each of its services (references to the executable DLLs). The (Sybase) database managed by the SDSRV uses this information to satisfy queries sent to the SDSRV.

The ESDT descriptor file text contains the same information mentioned above in an ODL format. The bulk of the descriptor files are placed in a given mode during the ECS 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.

ESDTs may need to be re-installed because ESDT Descriptor files may contain errors and/or the ESDT information may have evolved such that the old descriptor information needs to be replaced/updated in the relevant databases.

This section will describe how to Inspect, Install, Remove, Add, Update an ESDT., and associate the ESDT with a volume group.

## 18.1 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.

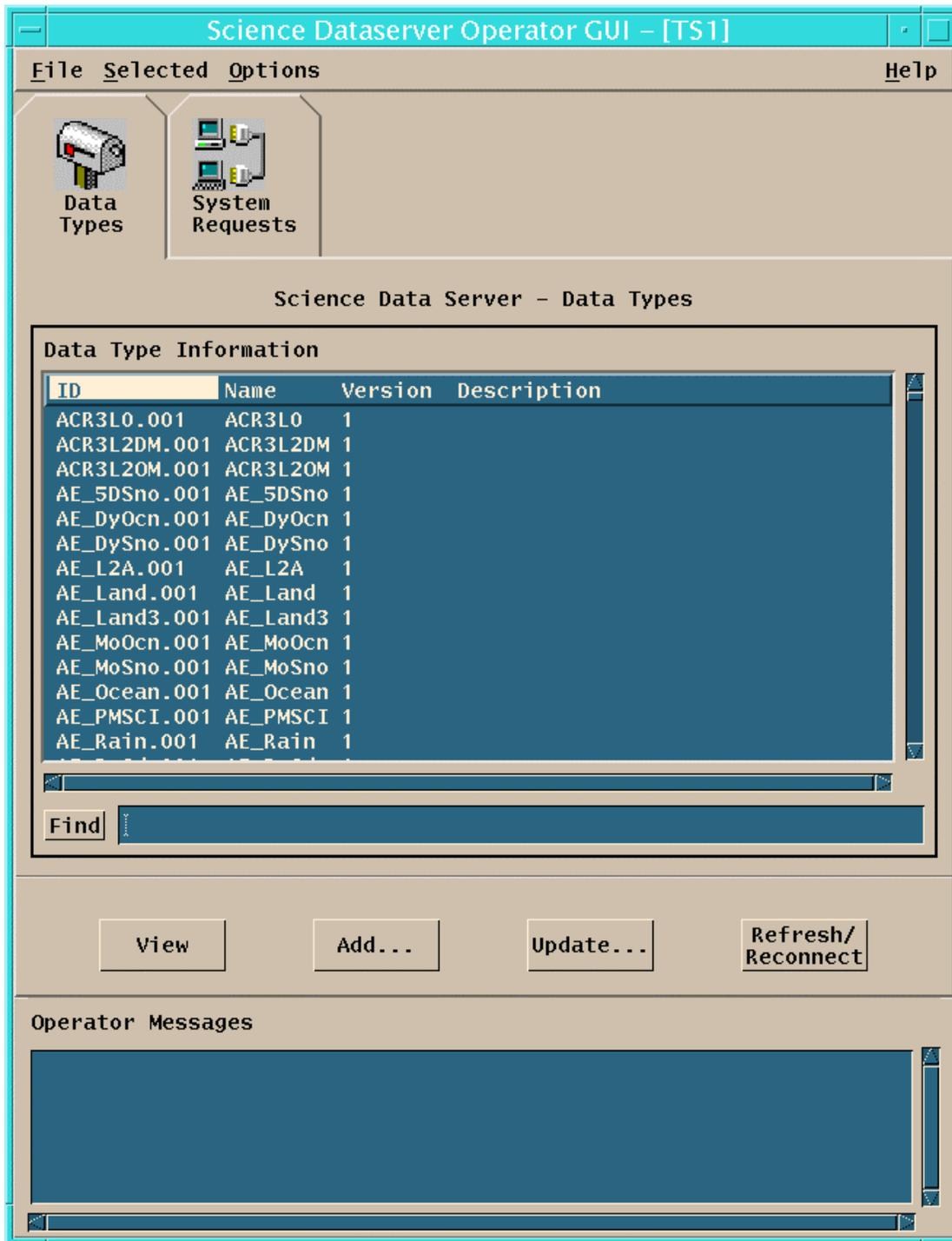
Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Science Data Server (EcDsScienceDataServer) for the desired mode is running.
3. The Sybase server for the Science Data Server database (e.g., for PVC: `p0acg05_svr`) is running.

To bring up the Science Data Server GUI, execute the steps that follow:

- 1 Log into the Science Data Server host. The file **.sitemap** under `/usr/ecs/<MODE>/CUSTOM` contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - `p0acs03`, for the GDAAC - `g0acs03`.
- 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 .**

- 3 Next, type **EcDsSdSrvGuiStart** <MODE>.
  - The Science Data Server GUI will be presented.



**Figure 18.1-1. Science Data Server Operator GUI**

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 to ECS, there are three other databases that need to have been successfully affected by the installation process. These are the Advertising, 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, select the shortname/version in the box and click on View.
- 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/DsESDTDesc. During the ESDT installation process, the descriptor file is copied from /usr/ecs/<MODE>/COMMON/data/ESS into /usr/ecs/<MODE>/COMMON/cfg/DsESDTDesc.
  - 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 ECS.

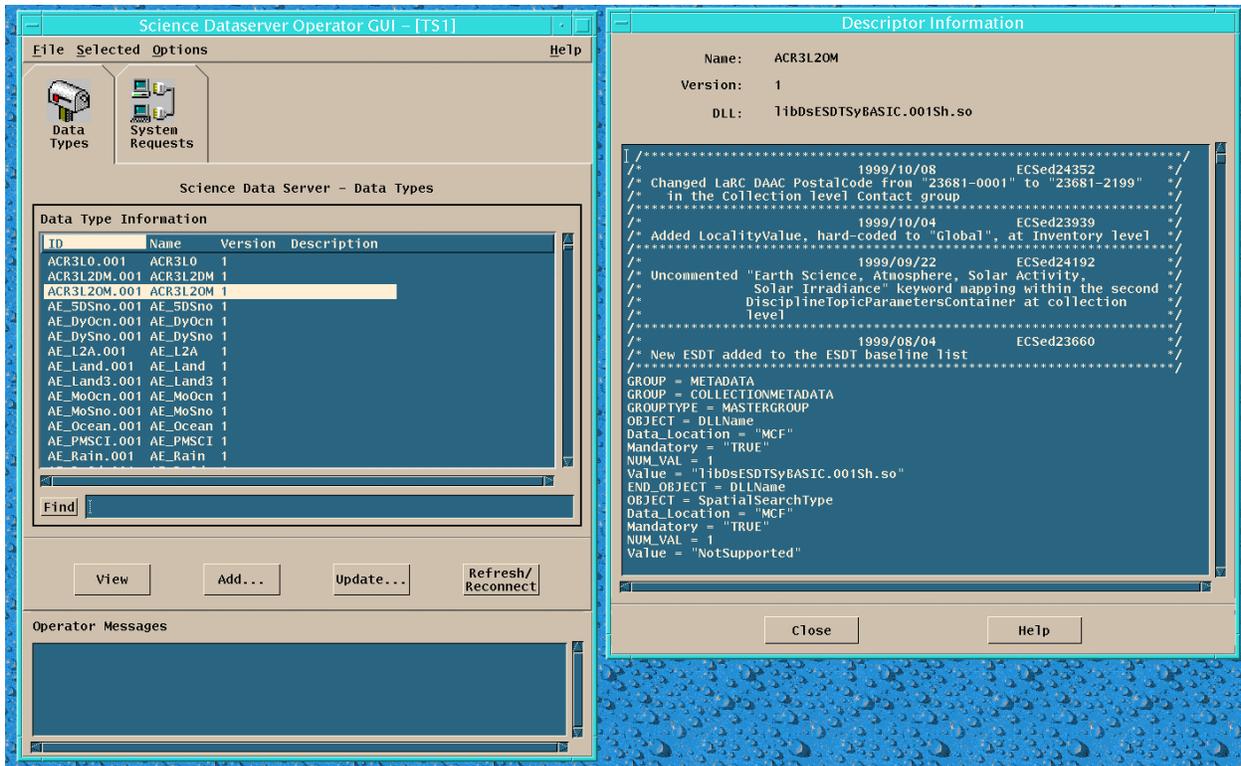


Figure 18.1-2. Viewing An ESDT Descriptor File

**Table 18.1-1. ESDT Inspection Using Science Data Server GUI –  
Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	Logon to Science Data Server platform	Log onto SDSRV host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</code>	Press <b>Return</b>
3	type <code>EcDsSdSrvGuiStart &lt;MODE&gt;</code>	Press <b>Return</b>
4	Scroll through Data Type Information box	none
5	Select shortname/version	Click the <b>View</b> button

## 18.2 Removing ESDTs

If one wants to install a modified version of an ESDT, retaining the shortname and version number, there are two approaches. The first way is to completely replace the descriptor file and corresponding ESDT data in the four 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:

### ESDT ADD Advantages

Can be done with any ESDT

### Disadvantages

Prior to Add, ESDT removal scripts must be run for all four 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 "StartTemperature=maintenance". This means it is unusable for normal processing until it is recycled with a StartTemperatur set to a value of "warm" or "cold".

Many other conditions must be met.

The removal of an ESDT can follow two different procedural paths. This difference centers on the ESDT removal from the (IOS) Advertising database. There are two different scripts available to accomplish this. One script, ContributionDriverStart, resides in ../utilities on the IOS server platform (e.g., p0ins02 in the PVC). The other script, EcfoDbDeleteCollection, resides in ../dbms/IOS on the IOS server platform. A separate set of procedural steps will be given for each scenario.

## 18.2.1 ESDT Removal Scenario 1 - Using ContributionDriverStart

Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase servers for the four involved databases are working properly. E.G. for the PVC:  
p0acg05\_srvr ---- supports the (DSS) Science Data Server databases  
p0ins01\_srvr ---- supports the (IDG/CSS) Subscription Server databases  
p0ins02\_srvr ---- supports the (IOS) Advertising and (DMS) Data Dictionary databases
3. The Advertising server (XXX ) for the mode to be affected is up.
4. The user knows the various ids, passwords and parameters required by the four scripts that will be used.

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

### Removal From the Advertising Database

1. Log into the (IOS) Advertising Server host. The file **.sitemap** under `/usr/ecs/<MODE>/CUSTOM` contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins02, for the GDAAC - g0ins02.
2. At the UNIX prompt on said host , type **cd /usr/ecs/<MODE>/CUSTOM/utilities**, press **Return** .
  1. This subdirectory contains the script to be run - **ContributionDriverStart**.
3. At the UNIX prompt, type **ContributionDriverStart <MODE>**, press **Return**.
  - This starts the heavily interactive script to remove the given ESDT from the Advertising database.
  - It is assumed a Carriage Return is pressed after each entry below.
4. The user will be prompted for a DCE\_Login id and password. Respond with the appropriate values.
  - Please enter DCE user name : **XXXXXXXX** ← use appropriate value
  - Please enter DCE password : **XXXXXXXX** ← use appropriate value
5. The user will be prompted to select the appropriate action from a menu.
  - Select contribution menu : **3** ← use this value
  - "3" selects Delete Advertisement
6. The user will be prompted to select the appropriate manner in which to make the advertisement deletion.
  - **> 3** ← use this value
  - "3" indicates a ShortName and VersionID will be used to make the advertisement deletions.
7. The user will be prompted for the specific ShortName and VersionID.
  - Please enter the ShortName : **AE\_5Dsno** ← use an appropriate value
  - Please enter the VersionID : **001** ← use an appropriate value
8. The user will be prompted if they are certain they want to take this course of action.

- MESSAGE : Do you really want to delete the ads related to AE\_5DSno (y/n)? y ← use this value to start processing
- Please enter the VersionID : 001 ← use an appropriate value

**Note:** Since running ContributionDriverStart is highly interactive, a sample session follows to put everything in context. .

- At the end of the session, the user is asked what is to be done next. The sample session illustrates the course of action to take to end the Remove ESDT session. However, the user can elect to continue and choose options other than Advertisement removal.

### Sample ContributionDriverStart Session

```
p0ins02{cmts1}[214]->ContributionDriverStart TS1
Warning: Could not open message catalog "oodce.cat"
06/12/01 14:10:43: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdCoreCat.dat.rcat
06/12/01 14:10:43: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdEcHtmlLibCat.dat.rcat
06/12/01 14:10:43: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdHtmlCoreCat.dat.rcat
06/12/01 14:10:43: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdHtmlSubsCat.dat.rcat
06/12/01 14:10:44: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdPersistentLibCat.dat.rcat
06/12/01 14:10:44: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdSearchLibCat.dat.rcat
06/12/01 14:10:44: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdServerLibCat.dat.rcat
06/12/01 14:10:44: Thread ID : 1 : loading resource catalog file from
/usr/ecs/TS1/CUSTOM/data/IOS/ResourceCatalogs/IOAdSubsCat.dat.rcat
06/12/01 14:10:44: Thread ID : 1 : Name = EcIOAdCGIProgs
ProgramID = 3000001
ApplicationID = 3000001
Site = PVC
DeltaTime = 3600
MajorVersion = 1
MinorVersion = 0
KeyFile = CUSTOM/security/EcIOAdServer.Keyfile
PrincipalName = EcIOAdServer
aclDBName = EcIOAdServerDB
Release = B
DebugLevel = 3
SubSysName = IOS
AppLogLevel = 0
AppLogSize = 200000
```

```
Please enter DCE user name : XXXXXXXX
Please enter DCE password : XXXXXXXX
----- CONTRIBUTION -----
```

- 1 - Insert Advertisement
- 2 - Update Advertisement
- 3 - Delete Advertisement
- 4 - Search Advertisement
- 5 - Search Service
- 6 - Exit

```
Select contribution menu : 3
```

1. Delete by ID
2. Delete ESDT by title (using LIKE :-))(subscribable events included)
3. Delete ESDT by ShortName and VersionID (subscribable events excluded)
0. Exit

> 3

Please enter the ShortName : **AE\_5DSno**

Please enter the VersionID : **001**

MESSAGE : Do you really want to delete the ads related to AE\_5DSno (y/n)? **y**

06/12/01 14:11:17: Thread ID : 1 :

Client Path: ././subsys/ecs/TS1/EcIoAdServer

06/12/01 14:11:18: Thread ID : 1 : EcNsServiceLocClient.C - Next Binding:

4a550896-5ee1-11d5-97d3-c676dc3baa77@ncacn\_ip\_tcp:198.118.220.59[]

06/12/01 14:11:18: Thread ID : 1 : EcNsServiceLocClient.C - Trying binding:

4a550896-5ee1-11d5-97d3-

c676dc3baa77@ncacn\_ip\_tcp:198.118.220.59[58584]

06/12/01 14:11:18: Thread ID : 1 : EcNsServiceLocClient.C - Binding to be

----- **etc., bla bla bla** -----

Adv. 1016990 is successfully deleted.

Deleting 1016991 ...

06/12/01 14:11:25: Thread ID : 1 :

----- **etc., bla bla bla** -----

4a550896-5ee1-11d5-97d3-

c676dc3baa77@ncacn\_ip\_tcp:198.118.220.59[58584]

06/12/01 14:11:45: Thread ID : 1 : EcNsServiceLocClient.C - Binding to be

returned:

4a550896-5ee1-11d5-97d3-

c676dc3baa77@ncacn\_ip\_tcp:198.118.220.59[58584]

06/12/01 14:11:45: Thread ID : 1 :

client: server binding is 4a550896-5ee1-11d5-97d3-

c676dc3baa77@ncacn\_ip\_tcp:198.118.220.59[58584]

06/12/01 14:11:49: cellName = PVC

1. Delete by ID
2. Delete ESDT by title (using LIKE :-))(subscribable events included)
3. Delete ESDT by ShortName and VersionID (subscribable events excluded)
0. Exit

> 0

----- CONTRIBUTION -----

- 1 - Insert Advertisement
- 2 - Update Advertisement
- 3 - Delete Advertisement
- 4 - Search Advertisement
- 5 - Search Service
- 6 - Exit

Select contribution menu : **6**

ContributionDriverStart[49]: 28943 Killed

### Removal 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. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins02, for the GDAAC - g0ins02.
- 2 At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/dbms/DMS`, press **Return** .
  - This subdirectory contains the script to be run - **DmDbCleanCollection**.
3. Prior to running the script, four environmental parameters need to be assigned proper values. These parameters are DSQUERY, DBNAME, DBUSERNAME, and DBPASSWD. 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. DBPASSWD is the Data Dictionary login password. To set these environmental parameters in C shell, follow the sample steps that follow. Values are for the PVC in mode TS1. DBUSERNAME and DBPASSWD values won't be displayed here for security reasons.
  - At the UNIX prompt , type `setenv DSQUERY p0ins02_srvr`, press **Return** .
  - At the UNIX prompt , type `setenv DBNAME EcDmDictService_TS1`, press **Return** .
  - At the UNIX prompt, type `setenv DBUSERNAME *****`, press **Return** .
  - At the UNIX prompt, type `setenv DBPASSWD *****`, press **Return** .
4. The script to be run requires two arguments - an ESDT shortname and the version number for said shortname. The form is **DmDbCleanCollection ShortName Version** . A PVC example would be:
  - At the UNIX prompt, type **DmDbCleanCollection AE\_5Dsno 001**, press **Return**

### Removal From the Subscription Database

- 1 Log into the (CSS/IDG) Subscription Server host. The file **.sitemap** under `/usr/ecs/<MODE>/CUSTOM` contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins01, for the GDAAC - g0ins01.
- 2 At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/utilities`, press **Return** .
  - This subdirectory contains the script to be run - **dbDeleteEvents.csh**.
3. This script requires five arguments. This will be of the form **dbDeleteEvents.csh mode ShortName Version ID UserName PassWd** .  
The arguments mean:  
**mode**            The DAAC mode to be affected by the script

**ShortName** The ShortName of the ESDT to be removed  
**VersionID** The version of ShortName to be removed  
**UserName** The Subscription database login username  
**PassWd** 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 \*\*\*\*\* \*\*\*\*\***, press **Return**.  
**Note:** for security reasons, the actual database login ID and Password are not shown.

### Removal 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. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0acs03, for the GDAAC - g0acs03.
- 2 At the UNIX prompt on said host , type **cd /usr/ecs/<MODE>/CUSTOM/utilities**, press **Return** .
  - This subdirectory contains the script to be run - **EcDsSrRmesdt** .

3. This script requires at least two arguments. This will be of the form **EcDsSrRmesdt mode DescripFileName1 DescripFileName2 ....** .

The arguments mean:

**mode** The DAAC mode to be affected by the script  
**DescripFileName1** The name of an ESDT decriptor file that corresponds to the ESDT to be removed.  
**DescripFileNameN** 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 on the PVC, at the UNIX prompt , type **EcDsSrRmesdt TS1 DsESDTAmAE\_5Dsno.001.desc**, press **Return**.  
**Note01:** 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/DsESDTDesc` .  
**Note02:** The ESDT version ID number is incorporated in the descriptor file name as exemplified by the location of ".001" in **DsESDTAmAE\_5Dsno.001.desc** .

**Table 18.2.1-1. ESDT Removal Using ContributionDriverStart - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
<b>Removal From Advertising Server DB</b>		
1	Logon to (IOS) Advertising Server platform	Log onto IOS host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</b>	Press <b>Return</b>
3	type <b>ContributionDriverStart &lt;MODE&gt;</b>	Press <b>Return</b>
4a	At "Please enter DCE user name : " prompt, type <i>username</i>	Press <b>Return</b>
4b	At "Please enter DCE password : " prompt, type <i>password</i>	Press <b>Return</b>
5	At "Select contribution menu:" prompt, type <b>3</b>	Press <b>Return</b>
6	At ">" prompt, type <b>3</b>	Press <b>Return</b>
7a	At "Please enter the ShortName : " prompt, type <i>ShortName</i>	Press <b>Return</b>
7b	At "Please enter the VersionID : " prompt, type <i>VersionID</i>	Press <b>Return</b>
8	At "MESSAGE : Do you really want to delete the ads related to <i>ShortName</i> (y/n)?" prompt, type <b>y</b>	Press <b>Return</b>
<b>Removal From Data Dictionary Server DB</b>		
1	Logon to (DMS) Advertising Server platform	Log onto DMS host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/dbms/DMS</b>	Press <b>Return</b>
3a	type <b>setenv DSQUERY sybase_ddict_srvr</b>	Press <b>Return</b>
3b	type <b>setenv DBNAME EcDmDictService_dbname</b>	Press <b>Return</b>
3c	type <b>setenv DBUSERNAME *****</b>	Press <b>Return</b>
3d	type <b>setenv DBPASSWD *****</b>	Press <b>Return</b>
4	type <b>DmDbCleanCollection ShortName VersionID</b>	Press <b>Return</b>
<b>Removal From Subscription Server DB</b>		
1	Logon to (CSS/IDG) SubscriptionServer platform	Log onto CSS/IDG host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</b>	Press <b>Return</b>
3	type <b>dbDeleteEvents.csh mode ShortName VersionID UserName Passwd</b>	Press <b>Return</b>
<b>Removal From Science Data Server DB</b>		
1	Logon to (DSS) Science Data Server platform	Log onto DSS/SDSRV host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</b>	Press <b>Return</b>
3	type <b>EcDsSrRmesdt mode DescriptFileName1 ...</b>	Press <b>Return</b>

## 18.2.2 ESDT Removal Scenario 2 - Using EcloDbDeleteCollection

Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase servers for the four involved databases are working properly. E.G. for the PVC:  
p0acg05\_srvr ---- supports the (DSS) Science Data Server databases  
p0ins01\_srvr ---- supports the (IDG/CSS) Subscription Server databases  
p0ins02\_srvr ---- supports the (IOS) Advertising and (DMS) Data Dictionary databases
3. The user knows the various ids, passwords and parameters required by the four scripts that will be used.

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

### Removal From the Advertising Database

- 1 Log into the (IOS) Advertising Server host. The file **.sitemap** under `/usr/ecs/<MODE>/CUSTOM` contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins02, for the GDAAC - g0ins02.
- 2 At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/dbms/IOS`, press **Return** .
  2. This subdirectory contains the script to be run - **EcIoDbDeleteCollection**.
3. EcIoDbDeleteCollection requires seven arguments. These are  
**mode** The mode to be affected by the ESDT removal  
**USER** The Advertising database login username  
**PASSWORD** The Advertising database login password  
**SERVER** The Sybase server that contains the Advertising database  
**DBNAME** The name of the Advertising database from which the ESDT will be removed  
**ShortName** The shortname of the ESDT to be removed  
**VersionID** The version ID of the ESDT to be removed

As an example, to remove the ESDT with shortname AE\_5Dsno, versioID 001, from the mode TS1 Advertising database in the PVC

- At the UNIX prompt, type `EcIoDbDeleteCollection TS1 **** * p0ins02_srvr IoAdAdvService_TS1 AE_5Dsno 001`, press **Return**  
**Note:** The database login username and password are not shown for security reasons.

### Removal 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. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins02, for the GDAAC - g0ins02.
- 2 At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/dbms/DMS`, press **Return** .
  - This subdirectory contains the script to be run - **DmDbCleanCollection**.
3. Prior to running the script, four environmental parameters need to be assigned proper values. These parameters are DSQUERY, DBNAME, DBUSERNAME, and DBPASSWD. 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. DBPASSWD is the Data Dictionary login password. To set these environmental parameters in C shell, follow the sample steps that follow. Values are for the PVC in mode TS1. DBUSERNAME and DBPASSWD values won't be displayed here for security reasons.

- At the UNIX prompt , type **setenv DSQUERY p0ins02\_srvr**, press **Return** .
  - At the UNIX prompt , type **setenv DBNAME EcDmDictService\_TS1**, press **Return** .
  - At the UNIX prompt, type **setenv DBUSERNAME \*\*\*\*\***, press **Return** .
  - At the UNIX prompt, type **setenv DBPASSWD \*\*\*\*\***, press **Return** .
4. The script to be run requires two arguments - an ESDT shortname and the version number for said shortname. The form is **DmDbCleanCollection ShortName Version** . A PVC example would be:
- At the UNIX prompt, type **DmDbCleanCollection AE\_5Dsno 001**, press **Return**

### Removal From the Subscription Database

- 1 Log into the (CSS/IDG) Subscription Server host. The file **.sitemap** under **/usr/ecs/<MODE>/CUSTOM** contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0ins01, for the GDAAC - g0ins01.
- 2 At the UNIX prompt on said host , type **cd /usr/ecs/<MODE>/CUSTOM/utilities**, press **Return** .
  - This subdirectory contains the script to be run - **dbDeleteEvents.csh**.
3. This script requires five arguments. This will be of the form **dbDeleteEvents.csh mode ShortName Version ID UserName PassWd** .  
The arguments mean:
 

<b>mode</b>	The DAAC mode to be affected by the script
<b>ShortName</b>	The ShortName of the ESDT to be removed
<b>VersionID</b>	The version of ShortName to be removed
<b>UserName</b>	The Subscription database login username
<b>PassWd</b>	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 \*\*\*\*\* \*\*\*\*\***, press **Return**.  
**Note:** for security reasons, the actual database login ID and Password are not shown.

### Removal 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. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0acs03, for the GDAAC - g0acs03.

2 At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/utilities`, press **Return** .

- This subdirectory contains the script to be run - `EcDsSrRmesdt` .

3. This script requires at least two arguments. This will be of the form `EcDsSrRmesdt mode DescripFileName1 DescripFileName2 ....` .

The arguments mean:

**mode** The DAAC mode to be affected by the script

**DescripFileName1** The name of an ESDT decriptor file that corresponds to the ESDT to be removed.

**DescripFileNameN** 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 on the PVC, at the UNIX prompt , type `EcDsSrRmesdt TS1 DsESDTAmAE_5Dsno.001.desc`, press **Return**.

**Note01:** 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/DsESDTDesc` .

**Note02:** The ESDT version ID number is incorporated in the descriptor file name as exemplified by the location of ".001" in `DsESDTAmAE_5Dsno.001.desc` .

**Table 18.2.2-1. ESDT Removal Using EcldbDeleteCollection - Quick-Step Procedures (1 of 2)**

Step	What to Enter or Select	Action to Take
<b>Removal From Advertising Server DB</b>		
1	Logon to (IOS) Advertising Server platform	Log onto IOS host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/dbms/IOS</code>	Press <b>Return</b>
3	type <code>EcldbDeleteCollection mode USERNAME PASSWORD SERVER DATABASE ShortName VersionID</code>	Press <b>Return</b>
<b>Removal From Data Dictionary Server DB</b>		
1	Logon to (DMS) Advertising Server platform	Log onto DMS host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/dbms/DMS</code>	Press <b>Return</b>
3a	type <code>setenv DSQUERY sybase_ddict_srvr</code>	Press <b>Return</b>
3b	type <code>setenv DBNAME EcDmDictService_dbname</code>	Press <b>Return</b>
3c	type <code>setenv DBUSERNAME *****</code>	Press <b>Return</b>
3d	type <code>setenv DBPASSWD *****</code>	Press <b>Return</b>
4	type <code>DmDbCleanCollection ShortName VersionID</code>	Press <b>Return</b>

**Table 18.2.2-1. ESDT Removal Using EcloDbDeleteCollection - Quick-Step Procedures (2 of 2)**

Step	What to Enter or Select	Action to Take
<b>Removal From Subscription Server DB</b>		
1	Logon to (CSS/IDG) SubscriptionServer platform	Log onto CSS/IDG host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</code>	Press <b>Return</b>
3	type <code>dbDeleteEvents.csh mode ShortName VersionID UserName Passwd</code>	Press <b>Return</b>
<b>Removal From Science Data Server DB</b>		
1	Logon to (DSS) Science Data Server platform	Log onto DSS/SDSRV host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</code>	Press <b>Return</b>
3	type <code>EcDsSrRmesdt mode DescriptFileName1 ...</code>	Press <b>Return</b>

## 18.3 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.

### 18.3.1 Adding an ESDT Using the Science Data Server GUI

Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase servers for the four involved databases are working properly. E.G. for the PVC:  
p0acg05\_srvr ---- supports the (DSS) Science Data Server databases  
p0ins01\_srvr ---- supports the (IDG/CSS) Subscription Server databases  
p0ins02\_srvr ---- supports the (IOS) Advertising and (DMS) Data Dictionary databases
3. The subsystem servers associated with the four involved databases are running. E.G., for the PVC, mode TS1:

<u>Platform</u>	<u>Server</u>
p0acs03	/usr/ecs/TS1/CUSTOM/bin/DSS/EcDsScienceDataServer
p0ins01	/usr/ecs/TS1/CUSTOM/bin/CSS/EcSbSubServer
p0ins02	/usr/ecs/TS1/CUSTOM/bin/IOS/EcIoAdServer
p0ins02	/usr/ecs/TS1/CUSTOM/bin/DMS/EcDmDictServer

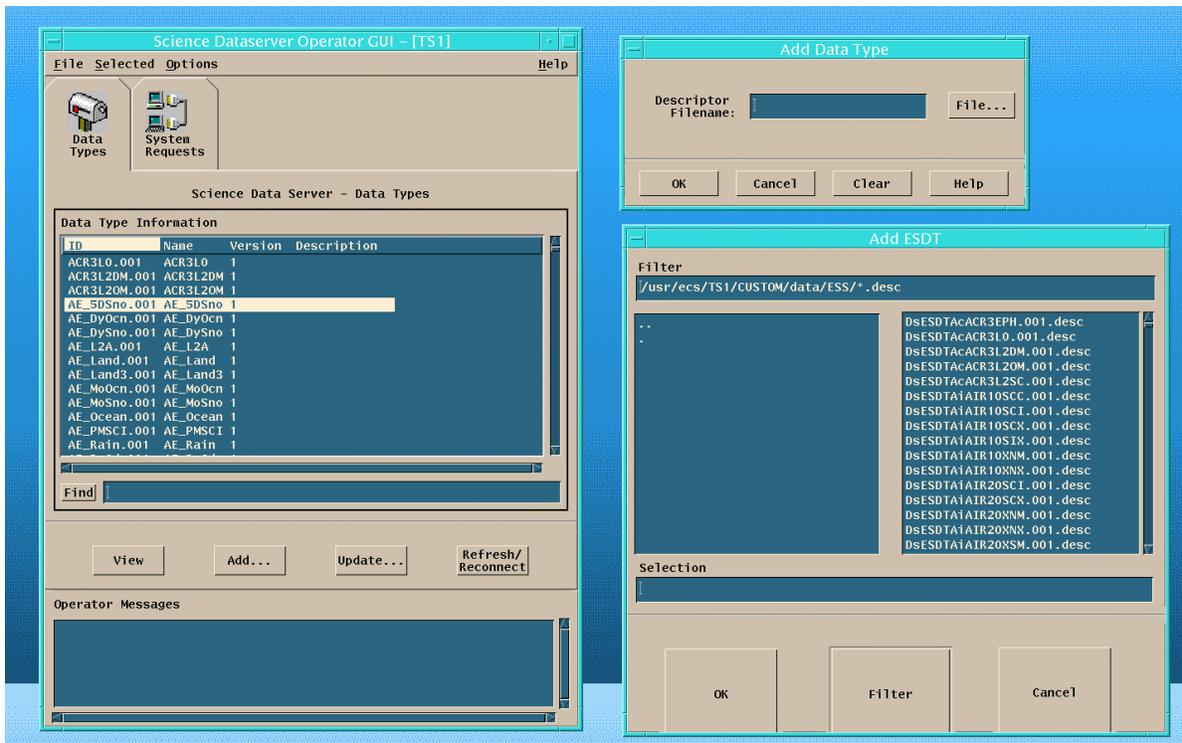
4. If the ESDT to be added already exists, it will be removed from the four involved databases before the Add operation takes place.

To Add an ESDT to the ECS for a given mode, execute the steps that follow:

1. Log into the (DSS) Science Data Server Server host. The file **.sitemap** under /usr/ecs/<MODE>/CUSTOM contains this information. Following established ECS

DAAC host naming conventions, some examples are: for the PVC - p0acs03, for the GDAAC - g0acs03.

2. At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/utilities`, press **Return** .
3. type `EcDsSdSrvGuiStart mode`, press **Return**
5. This brings up the Science Operator Dataserver Gui GUI. Refer to figure 18.3.1-1



**Figure 18.3.1-1. Adding an ESDT - The GUIs Involved**

4. Click the "Add..." button on the Science Dataserver Operator Gui GUI.
  - This brings up the "Add Data Type" GUI. Refer to figure 18.3.1-1
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. Refer to figure 18.3.1-1
6. The "Add ESDT" GUI displays a list of the eligible descriptor files that can be used to add an ESDT. One can tailor this list 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 Dataserver Operator Gui" log.
  - If everything went well, the selected ESDTs have been added.

**Table 18.3.1-1. Adding ESDTs With Science Data Server GUI**

Step	What to Enter or Select	Action to Take
1	Logon to Science Data Server platform	Log onto SDSRV host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</code>	Press <b>Return</b>
3	type <code>EcDsSdSrvGuiStart &lt;MODE&gt;</code>	Press <b>Return</b>
	SDSRV GUI appears	none
4	Click the "Add..." button in the SDSRV GUI	none
5	Click the "File..." button in the "Add Data Type" GUI	none
6	Select descriptor file names in "Add ESDT" GUI	none
7	Click <b>OK</b> in the "Add ESDT" GUI	none
8	Click <b>OK</b> in the "Add Data Type" GUI	none

## 18.4 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 ECS'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.

1. Add optional Collection level metadata
2. Add optional Inventory level metadata (including Product Specific Attributes (PSAs))
3. Add additional services
4. Add additional events
5. Add new parameters to existing services

6. Add qualifiers to existing events
7. Add additional valid values to Inventory level metadata attributes
8. 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)
9. Change a mandatory attribute to optional
10. Modify parameters in existing services

An ESDT Update cannot add mandatory attributes.

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

### 18.4.1 Updating an ESDT Using the Science Data Server GUI

Assumptions:

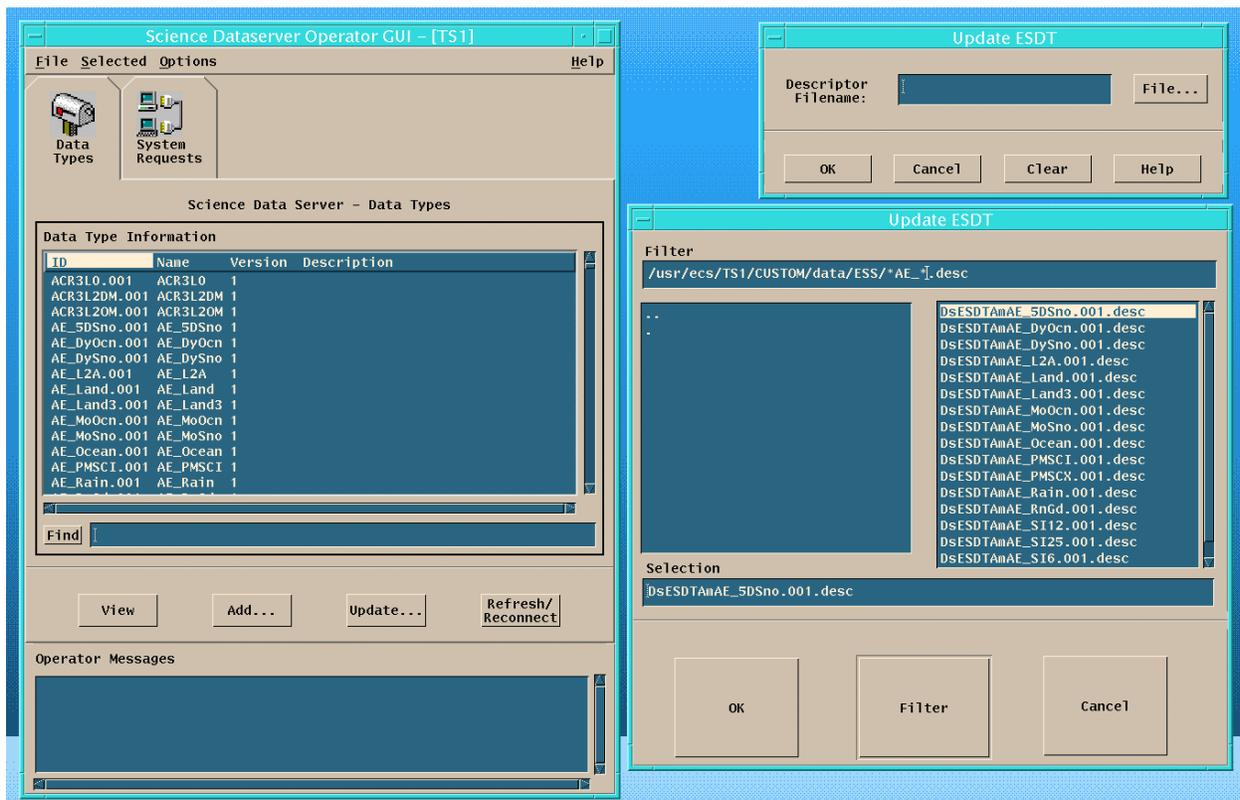
1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase servers for the four involved databases are working properly. E.G. for the PVC:  
p0acg05\_srvr ---- supports the (DSS) Science Data Server databases  
p0ins01\_srvr ---- supports the (IDG/CSS) Subscription Server databases  
p0ins02\_srvr ---- supports the (IOS) Advertising and (DMS) Data Dictionary databases
3. The subsystem servers associated with the four involved databases are running. E.G., for the PVC, mode TS1:

<u>Platform</u>	<u>Server</u>
p0acs03	/usr/ecs/TS1/CUSTOM/bin/DSS/EcDsScienceDataServer
p0ins01	/usr/ecs/TS1/CUSTOM/bin/CSS/EcSbSubServer
p0ins02	/usr/ecs/TS1/CUSTOM/bin/IOS/EcIoAdServer
p0ins02	/usr/ecs/TS1/CUSTOM/bin/DMS/EcDmDictServer

To Update an ESDT to the ECS for a given mode, execute the steps that follow:

1. Log into the (DSS) Science Data Server host. The file **.sitemap** under /usr/ecs/<MODE>/CUSTOM contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0acs03, for the GDAAC - g0acs03.
2. At the UNIX prompt on said host , type **cd /usr/ecs/<MODE>/CUSTOM/utilities**, press **Return** .
3. type **EcDsSdSrvGuiStart mode**, press **Return**

- This brings up the Science Operator Dataserver Gui GUI. Refer to figure 18.4.1-1.



**Figure 18.4.1-1. Updating An ESDT - The GUIs Involved**

4. Click the "Update..." button on the Science Dataserver Operator Gui GUI.
  - This brings up the small "Update ESDT" GUI. Refer to figure 18.4.1-1
5. The user can type in the descriptor file name for the ESDT to be updated. However, it is usually easier and less prone to error, to click the "File..." button.
  - This brings up the large "Update ESDT" GUI. Refer to figure 18.4.1-1
6. The large "Update ESDT" GUI displays a list of the eligible descriptor files that can be used to update an ESDT. One can tailor this list 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 Dataserver Operator Gui" log.
- If everything went well, the selected ESDTs have been Updated.

**Table 18.4.1-1. Updating ESDTs With Science Data Server GUI**

Step	What to Enter or Select	Action to Take
1	Logon to Science Data Server platform	Log onto SDSRV host
2	type <code>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</code>	Press <b>Return</b>
3	type <code>EcDsSdSrvGuiStart &lt;MODE&gt;</code>	Press <b>Return</b>
	SDSRV GUI appears	none
4	Click the "Update..." button in the SDSRV GUI	none
5	Click the "File..." button in the large "Update ESDT" GUI	none
6	Select descriptor file names in large "Update ESDT" GUI	none
7	Click <b>OK</b> in the large "Update ESDT" GUI	none
8	Click <b>OK</b> in the small "Update ESDT " GUI	none

## 18.5 ESDT Volume Group Configuration

Once an ESDT is installed into the ECS, the system knows how to deal with the collections and granules associated with that ESDT - up to a point. The Storage Management 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 .

For the official baseline document that describes ESDTs and includes the configured Volume Group information, access <http://pete.hitc.com/baseline/index.html> , click on Technical Documents and then pick the row that looks like "019 ESDT Baseline 910-TDA-019-Revxx.xls"

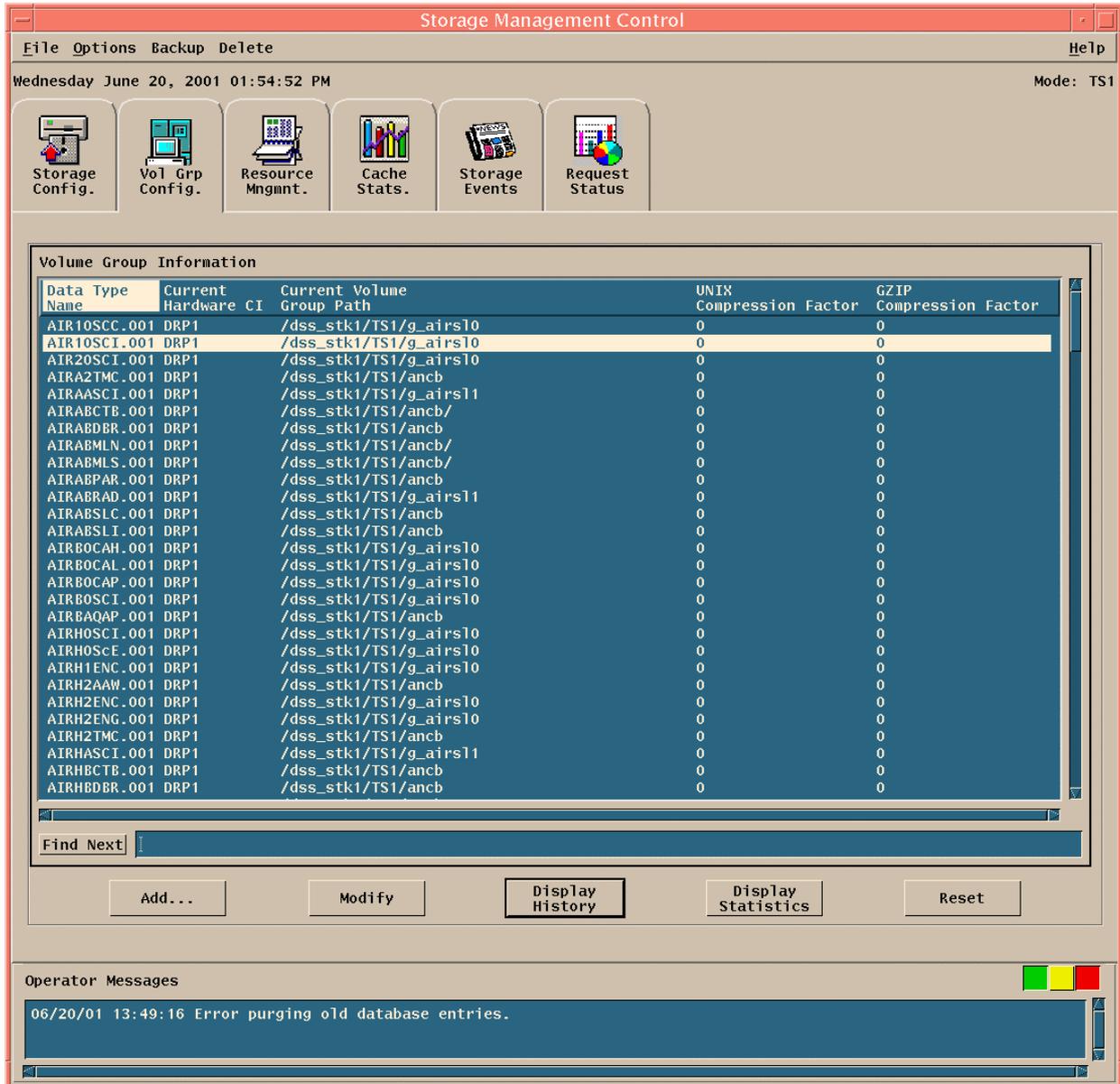
### 18.5.1 Modifying an ESDT's Volume Group Information

Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase server for the Storage Management Subsystem databases is working properly. E.G. for the PVC, p0acg05\_srvr .

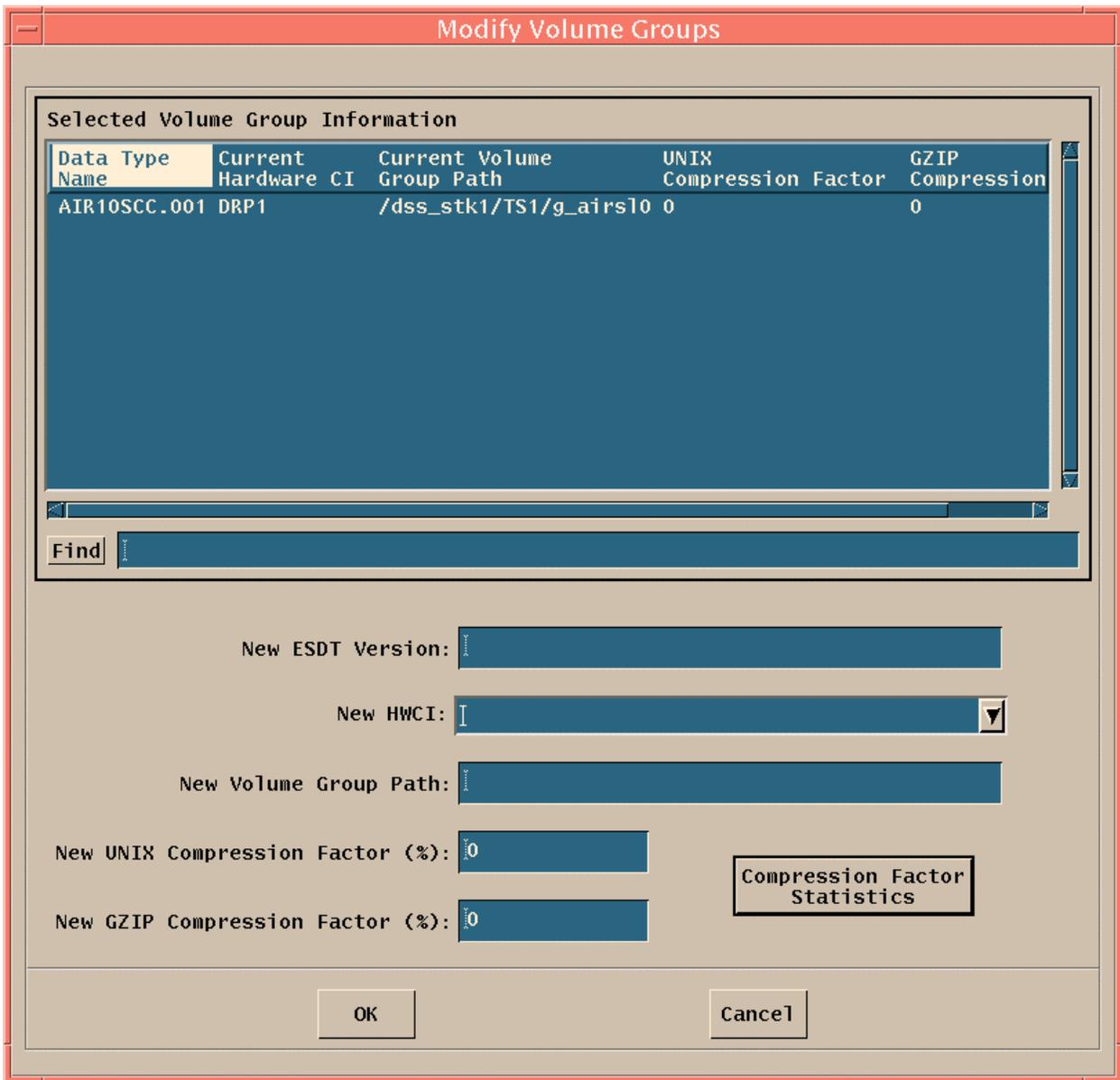
To modify a given ESDT's Volume Group information for a given mode, execute the steps that follow:

1. Log into the (DSS) Data Distribution Server host. The file **.sitemap** under `/usr/ecs/<MODE>/CUSTOM` contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0dis02, for the GDAAC - g0dis02.
2. At the UNIX prompt on said host , type `cd /usr/ecs/<MODE>/CUSTOM/utilities`, press **Return** .
3. Type `EcDsStmgtGuiStart mode`, press **Return**
  - This brings up the Storage Management Control GUI. Refer to figure 18.5.1-1



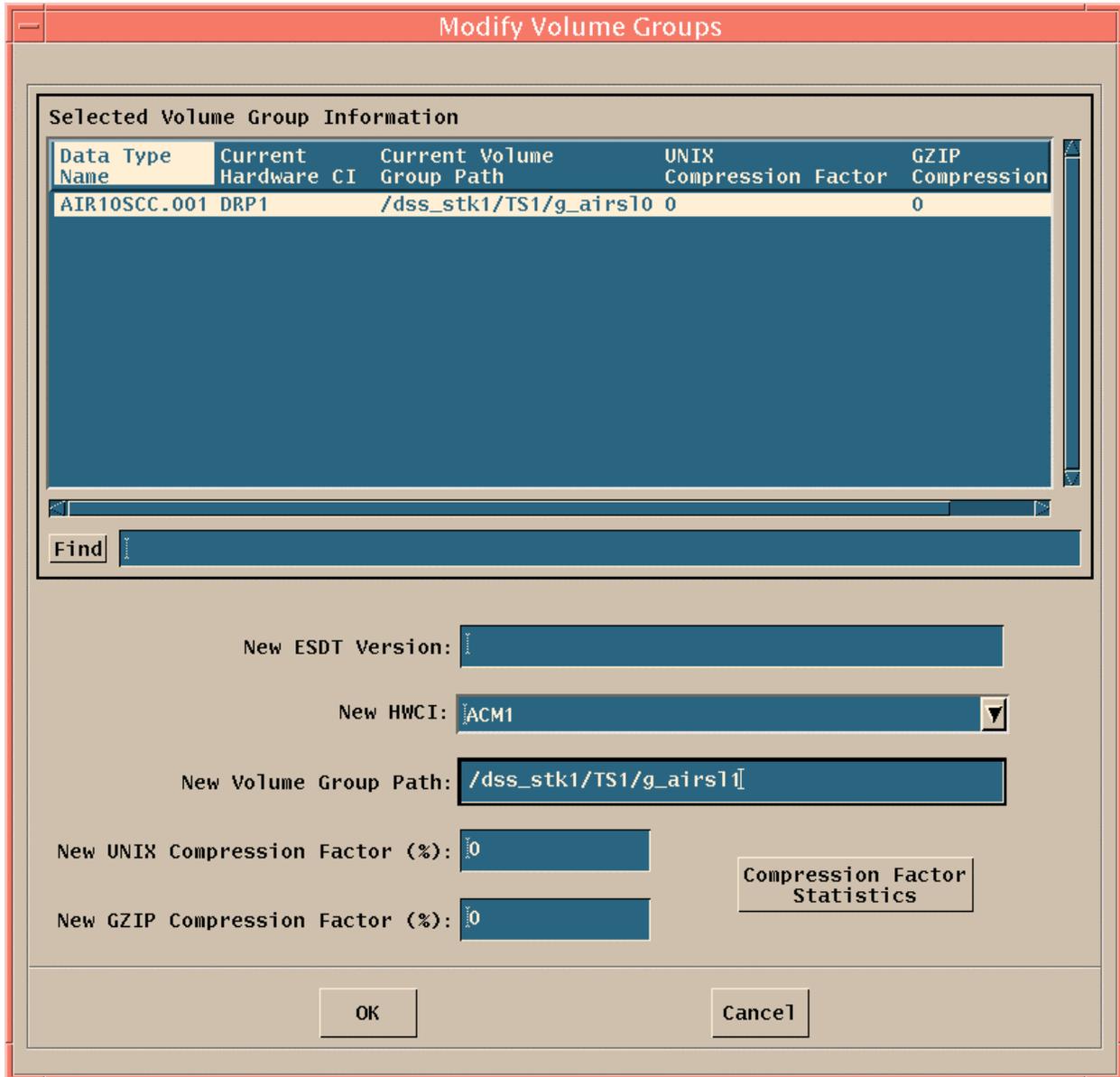
**Figure 18.5.1-1. Storage Management GUI**

4. Click the "Vol Grp Config" tab on the GUI
  - This brings up the "Volume Group Information" pane which is what figure 18.5.1-1 actually illustrates.
5. Select an ESDT to be modified by scrolling through the Volume Group Information pane and clicking on the ShortName.VersionID desired.
6. Click the Modify button.
  - The Modify Volume Groups GUI will appear. See figure 18.5.1-2



**Figure 18.5.1-2. Modify Volume Groups GUI**

7. Type in only the information that needs to be changed.
  - The HWCI information must be selected from a list which is brought up by clicking on the New HWCI selection arrow.
  - A view of the Modify Volume Groups GUI with user entries made is illustrated with figure 18.5.1-3 .



**Figure 18.5.1-3. Modify Volume Groups GUI with entries**

8. Click the "OK" box when satisfied with the modifications entered.
  - The Update ESDT process will start.
  - The "Operator Messages" will display the status of the install process.

**Table 18.5.1-1. Changing ESDT Volume Group Information**

Step	What to Enter or Select	Action to Take
1	Logon to (DSS) Data Distribution Server host	Log onto DDIST host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</b>	Press <b>Return</b>
3	type <b>EcDsStmgtGuiStart &lt;MODE&gt;</b>	Press <b>Return</b>
	Stmgt Control GUI appears	none
4	Click the " <b>Vol Grp Config</b> " tab	none
5	Click on the <b>ShortName.VersionID</b> to be modified	none
6	Click the <b>Modify</b> button	none
	The Modify Volume Groups GUI will appear	
7	Type in the new information	none
8	Click <b>OK</b> button	none

## 18.5.2 Adding an ESDT's Volume Group Information

Assumptions:

1. The required environment variables (e.g., **DISPLAY**) have been set properly.
2. The Sybase server for the Storage Management Subsystem databases is working properly. E.G. for the PVC, p0acg05\_svr .

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:

1. Log into the (DSS) Data Distribution Server host. The file **.sitemap** under /usr/ecs/<MODE>/CUSTOM contains this information. Following established ECS DAAC host naming conventions, some examples are: for the PVC - p0dis02, for the GDAAC - g0dis02.
2. At the UNIX prompt on said host, type **cd /usr/ecs/<MODE>/CUSTOM/utilities**, press **Return** .
3. Type **EcDsStmgtGuiStart mode**, press **Return**
  - This brings up the Storage Management Control GUI. Refer to figure 18.5.
4. Click the "Vol Grp Config" tab on the GUI
  - This brings up the "Volume Group Information" pane which is what figure 18.5.1-1 actually illustrates.
5. Click the "Add..." button.
  - The Add Volume Group GUI will appear

See figure 18.5.2-1

The screenshot shows a dialog box titled "Add Volume Group". It contains the following fields and controls:

- Data Type.Version:** A text input field containing the character "I".
- HWCI:** A dropdown menu with a downward-pointing arrow.
- Volume Group Path:** A text input field.
- UNIX Compression Factor (%):** A text input field containing the number "0".
- GZIP Compression Factor (%):** A text input field containing the number "0".
- Volume Group Type:** A group box containing three radio buttons:  PRIMARY,  BACKUP, and  OFFSITE.
- Buttons:** "OK" and "Cancel" buttons at the bottom.

**Figure 18.5.2-1. Add Volume Group GUI**

6. Type in all the required information.
  - The HWCI information must be selected from a list which is brought up by clicking on the HWCI selection arrow.
7. Click the "OK" box when satisfied with the information entered.
  - The Add Volume Group process will start.
  - The "Operator Messages" will display the status of the install process.

**Table 18.5.2-1. Adding ESDT Volume Group Information**

<b>Step</b>	<b>What to Enter or Select</b>	<b>Action to Take</b>
1	Logon to (DSS) Data Distribution Server host	Log onto DDIST host
2	type <b>cd /usr/ecs/&lt;MODE&gt;/CUSTOM/utilities</b>	Press <b>Return</b>
3	type <b>EcDsStmgtGuiStart &lt;MODE&gt;</b>	Press <b>Return</b>
4	Click the " <b>Vol Grp Config</b> " tab	none
5	Click the " <b>Add...</b> " button	none
6	Enter all the requisite information	none

This page intentionally left blank.

# Appendix A. Troubleshooting and General Investigation

---

This section is primarily meant to serve as a reference to SSIT personnel for use in diagnosing problems encountered during testing of PGEs in the ECS system. This section is divided into six main categories of failure, each section will discuss the handling those failures.

- ESDT Installation failure
- File Insert Failure
- File Acquire Failure
- DPR Generation Failure
- DPR Scheduling Failure
- PGE Execution Failure

## A.1 ESDT Installation Failure

Section 18.3 explains how to install an ESDT. The following sections are general guidelines for a user to handle an ESDT installation failure.

A.1.1.1 Check the status of the servers: When you install an ESDT, there are certain servers which have to be brought up properly. If any of the necessary servers are down or haven't been brought up properly, ESDT installation fails. Please refer to section 18.3 for more information about the servers, which are involved when installing an ESDT.

A.1.1.2 Check the ESDT descriptor file and the DLL file: Inside Science Data Server configuration file, which is EcDsScienceDataServer.CFG (at /usr/ecs/<mode>/CUSTOM/cfg), there are entries which indicate where to find ESDT descriptor files and DLLs that are being installed. Those entries are DSSDESCINPUTDIR and DSSDLLDIR. In the ESDT descriptor file, there is an attribute name that is DLLName. The DLL file should be resided at DSSDLLDIR. If the DLL file name indicated in ESDT descriptor file can not be found under the DSSDLLDIR, a failure of installation will occur.

A.1.1.3 Remove the ESDT descriptor file properly before installing an updated version of the same ESDT : You don't have to worry about removing the descriptor file before installing an ESDT, when you are installing a totally new ESDT. But when you install an updated version of an ESDT, which has already been installed, you need to remove it first. Please refer to section 18.2 for procedure of removing an ESDT. Sometimes ESDT doesn't get removed properly, and then if you try to install it, the installation fails.

A.1.1.4 Check the Science Data Server ALOG File: During ESDT installation the Science Data Server writes status messages to the following log files, EcDsScienceDataServer.ALOG and EcDsScienceDataServerDebug.log. (These files are located at /usr/ecs/<mode>/CUSTOM/logs)

Entries to the ALOG file should include the ShortName of the ESDT. If the ShortName does not appear with a time stamp which reflects the time of the attempted installation, then the request of installation was not communicated to SDSRV. This might be the case if the SDSRV subsystem is not running. These logs files can be inspected further for other errors, if the ESDT ShortName does appear.

**A.1.1.5 Check the IOS Server ALOG file:** During ESDT installation, the Science Data Server notifies the Advertising Subsystem that the ESDT needs to be advertised for later use. The ALOG file (located at /usr/ecs/<mode>/CUSTOM/logs) EcIoAdServer.ALOG should include the ShortName of the ESDT. If the ShortName does not appear with a time stamp which reflects the time of the attempted installation, then Science Data Server ALOG file should indicate failure to advertise the ESDT. This could be caused by IOS Server is not up running properly while the ESDT is being installed.

## **A.2 Insert File Failure**

Files may be inserted into the Science Data Server in a variety of ways, depending on the type of file. In general, files must be accompanied by a descriptive metadata (.met) file in order for the Science Data Server to process them successfully. The types of files to be inserted include DAP (Delivered Algorithm Package) files, input and output science data granules, and PGE executable tar files (PGEEXE.tar). The Science Data Server provides a test driver to insert a file to SDSRV. The SSIT Manager GUI also provides various tools to insert a file, depending on its type.

For inserting files using the SDSRV test driver or the SSIT Manager, the user first needs to prepare a metadata file to go with the file to insert into Science Data Server. Output file insertions during destaging use metadata files generated by the PGE.

Files that are to be inserted to the Science Data Server using the SSIT Manager, in general, must be made known to the PDPS database in advance. This includes dynamic and static data files, and PGE executable tar file. This prerequisite is fulfilled by registering a PGE with the PDPS database.

If the Science Data Server returns a message indicating that the insertion has failed, or in the case of a Destaging failure, a number of things can be done to diagnose the problem causing the failure.

**A.2.1.1 Check for ESDT in SDSRV Database:** An ESDT corresponding to the file type to be inserted has to be installed properly. File insertion will fail if the ESDT is not installed prior to the insertion. Check the Science Data Server for the ESDT corresponding to the file type to be inserted. This is done in either of two ways. The first is to enter a few SQL commands on the UNIX command line. The second utilizes a database viewer GUI. With either method, the aim is to verify that the ESDT required is listed in the Science Data Server data base in a table called DsMdCollections.

The table below lists the steps required to view the SDSRV database using SQL commands.

**Table A.2.2.1-1. Viewing the SDSRV Database Using SQL Commands - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	Login to AIT computer which supports SDSRV using a generic SSIT account.	
2	isql -U <sdsrvusername> -P<password> -S<sdsrvserver>	press Return
3	use <databasename>	press Return
4	Go	press Return
5	Select ShortName from DsMdCollections where ShortName = <ESDTshortname>	press Return
6	Go	press Return

The table below lists the steps required to view the SDSRV database using a database viewer.

**Table A.2.2.1-2. Viewing the SDSRV Database Using the Database Viewer GUI - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports SDSRV using a generic SSIT account.	
2	(if necessary) setenv DISPLAY <machinename>:0.0	press Return
3	Wisqlite <MODE> &	press Return
4	enter into GUI login: DBUSERNAME: <sdsrvusername> DBPASSWD: <password>, DSQUERY: <sdsrvserver>	
5	select database by clicking DATABASE button	
6	type command as in 5 of above table	Click "execute"

For both methods, the aim is to locate the ESDT corresponding to the file to be installed. In the DsMdCollections table, the ESDT is located by the ShortName. If the ShortName is not listed, then the ESDT must be inserted to the Science Data Server before the file insertion occurs.

A.2.1.2 Check for ESDT in the Advertising Database: When an ESDT is installed into the Science Data Server data base, the system also makes entries in the Advertising (IOS) database. The number and types of entries depends on the contents of the ESDT descriptor file. File insertion failures may also be caused by missing or incomplete IOS database entries for the ESDT. Therefore, it is useful to check IOS to make sure the ESDT corresponding to the file type to be inserted has been properly advertised. This is done by checking the advertising database IoAdAdvService\_mode in a table called IoAdAdvMaster for the ShortName in question. This table, for each ESDT ShortName, should show several entries, the number depending on the descriptor file contents. An example of such a listing is given below.

**Table A.2.2.2-1. Sample Listing of ESDT Entries in Advertising Database**

<b>loAdAdvMaster</b>
MOD03.001
MOD03.001:ACQUIRE
MOD03.001:INSERT
MOD03.001:UPDATEMETADATA
MOD03.001:BROWSE
MOD03.001:GETQUERYABLEPARAMETERS
MOD03.001:INSPECT
MOD03.001:INSPECTCL
MOD03.001:DELETE
Subscribable Event:ID:##: MOD03.001:DELETE
Subscribable Event:ID:##: MOD03.001:INSERT
Subscribable Event:ID:##: MOD03.001:UPDATEMETADATA

The table below lists the steps required to view the Advertising database using SQL commands.

**Table A.2.2.2-2. Viewing the Advertising Database Using SQL Commands - Quick-Step Procedures**

<b>Step</b>	<b>What to Enter or Select</b>	<b>Action to Take</b>
1	login to AIT computer which supports the IOS subsystem using a generic SSIT account.	
2	isql -U<iosusername > -P<password > -S<ioserver>	press Return
3	use loAdAdvService_<MODE>	press Return
4	Go	press Return
5	select title from loAdAdvMaster where title like "<ESDTshortname>%"	press Return
6	Go	press Return

The table below lists the steps required to view the Advertising database using a database viewer.

**Table A.2.2.2-3. Viewing the Advertising Database Using the Data Base Viewer GUI - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the IOS subsystem using a generic SSIT account.	
2	(if necessary) setenv DISPLAY <machinename>:0.0	press Return
3	Wisqlite <MODE> &	press Return
4	enter into GUI login: DBUSERNAME: <iosusername> DBPASSWORD: <password>, DSQUERY: <ioserver>	
5	select database by clicking DATABASE button	
6	type command as in 5 of above table	Click "execute"

For both methods, the aim is to list all entries corresponding to the ESDT of the file to be installed. In the IoAdAdvMaster table, the ESDT is located by the title, which includes the shortname. If the shortname is not listed, then the ESDT must be inserted to the Science Data Server before the file insertion could succeed (see Section 5.3). If the listing does not include all required entries, then the ESDT must be removed from the Advertising , Science Data Server, and Data Dictionary databases. Section 5.3.2 describes ESDT removal from Science Data Server, Advertising and Data Dictionary while reinstallation is covered in Section 5.3.1.

A.2.1.3 Check for the volume group: File insertion fails if a volume group for the ESDT has not been set up properly. Please refer to section 18.5 for more information on how to install a volume group.

A.2.1.4 Check if the .met file is correct : Sometimes metadata in the .met file (which you insert along with the datafile) doesn't match with the metadata in the descriptor file of the ESDT. In this case file insertion fails. You either need to install a proper descriptor file or change the .met file to match it.

A.2.1.5 Check the directory where the files to be inserted are stored: This directory should be visible from the server from where you are inserting the files. In this case you will have to copy the files to a directory which is visible to the server.

A.2.1.6 Check SDSRV ALOG File: During any operation involving the Science Data Server, useful information reflecting SDSRV activities is written to two log files. These are EcDsScienceDataServer.ALOG and EcDsScienceDataServerDebug.log. Entries to the ALOG file should include the ShortName of the file data type. Timestamps, which appear throughout the logs files, should be checked to make sure any entries found for a shortname correspond to the time of the attempted insertion. If the ShortName does not appear, then the file insertion request was not communicated to SDSRV. This might be the case if the SDSRV subsystem is not running.

**Table A.2.2.3-1. Viewing the EcDsScienceDataServer.ALOG file - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the SDSRV subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE> /CUSTOM/logs	press Return
3	vi EcDsScienceDataServer.ALOG	press Return

If the shortname does appear in the ALOG file, then the ALOG may be investigated further to determine whether the metadata has been successfully validated. Successful metadata validation is indicated when “End Metadata Validation. ( Metadata is valid).” appears in the ALOG file. If the metadata is not valid, then the metadata validation section of the ALOG can be scanned to find what metadata errors have been identified by SDSRV.

A.2.1.7 Verify that the Servers are Running: File insertion requires not only that SDSRV be running, but also others like IOS, subscription server, subscription manager and Archive servers. This may be done in either of two ways. In the first, each subsystem host machine is checked for the status of its resident subsystems. In the second, the ECSAssist Gui is used to view the operation of all subsystems (section 5.2).

The short procedure below outlines how to check that a subsystem is running, using a command-line technique. This method requires that the user know the names of the subsystem components.

**Table A.2.2.4-1. Checking That Subsystems are Running Using a Command Line Approach - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the subsystem to check.	
2	ps -ef   grep mode	press Return

A.2.1.8 Check DCE Login Validity: A dce login is necessary for proper operation of the system by a user. Users should always login to dce at the start of a session, and in each xterm used. To check that the login is still valid, or that it was done at all, type “klist” at a UNIX prompt. If this command returns “No DCE identity available” then a fresh dce login is necessary.

## A.3 Acquire Failure

### A.3.1 Description

Files are acquired from the Science Data Server either through the SSIT Manager, DAP acquire tool (it can be used for any type of file acquire), or using a test driver from SDSRV.

Additionally, files are acquired by the system during the setup and execution of a PGE. Failure of an acquire is similar to insertion failure, and the methods to diagnose and resolve the failure also resemble those for insertions.

### A.3.2 Handling an Acquire failure:

Diagnosing an acquire failure involves inspecting various system log files and checking in directories involved with the process.

A.3.2.1 Check Science Data Server Log Files: The EcDsScienceDataServer.ALOG file should contain entries regarding the acquire activity and identify the file to be acquired by its ShortName. If the ShortName does not appear in the ALOG file, with a timestamp corresponding to the time of the attempted acquire, then SDSRV may not be running, or may not be communicating with other servers. If the ALOG file does contain entries for that ShortName, and indicates that two files (the file and its associated metadata file) are being distributed, the message in the ALOG file looks like following:

```
Msg: File 1 to be distributed: :SC:MOD03.001:1369:1.HDF-EOS
Priority: 0 Time : 07/29/98 12:35:42
PID : 24279:MsgLink :1684108385 meaningfulname
:DsSrWorkingCollectionDistributeOneDistributFile
```

**Msg: File 2 to be distributed: SCMOD03.0011369.met**

then SDSRV has completed its role in the acquire successfully.

**Table A.3.2.1-1. Viewing the EcDsScienceDataServer.ALOG file - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the SDSRV subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE> /CUSTOM/logs	press Return
3	vi EcDsScienceDataServer.ALOG	press Return

If the ALOG contains the ShortName, and also contains an error showing that the data file time stamp does not match the time stamp required by the acquire, then the data file needs to be removed from the Science Data Server and reinserted. This is usually done using a script called DsDbCleanSingleGranule.

The procedures of removing a data granule from Science Data Server are as follows:

In directory /usr/ecs/<MODE>/CUSTOM/dnms/DSS/ of the SDSRV host, set appropriate environment variables for DBUSERNAME, DBPASSWD, DSQUERY and DBNAME, then type command

```
DsDbCleanSingleGranule SC:<ShortName.version>:<dbID>
```

e.g. DsDbCleanSingleGranule SC:MOD000.001:1234. Then press return.

A.3.2.2 Check the Archive Server ALOG File: Acquire success from the Sciece Data Server is only part of the acquire process. Since any file entered into SDSRV is stored in the archive, the Archive Server must be involved during an acquire. Thus, it may be useful to inspect the Archive Server ALOG file ( EcDsStArchiveServer.ALOG ) to check for error messages associated with the ShortName of the file type.

**Table A.3.2.2-1. Viewing the EcDsStArchiveServer.ALOG file - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the Archive Server subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE> /CUSTOM/logs	press Return
3	vi EcDsStArchiveServer.ALOG	press Return

A.3.2.3 Check Staging Disk: During an acquire, files are copied to a staging area as an intermediate step before distributing them to their destination. As part of diagnosing an acquire failure it is useful to check the staging area to ascertain whether the files have completed part of their journey. Both the file and a subdirectory containing metadata information should be written to the staging area.

**Table A.3.2.3-1. Viewing the Staging Area - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the Archive Server subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE>/CUSTOM/drp/<archivehost>/data/staging/ user<#>	press Return
3	Ls -lrt	press Return

A.3.2.4 Check Staging Server ALOG: If a failure occurs in copying the files to the staging area, then the Staging log files (EcDsStStagingDiskServer.ALOG or EcDsStStagingMonitorServer.ALOG) may reveal the cause.

**Table A.3.2.4-1. Viewing the EcDsStagingServer.ALOG file - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the Archive Server subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE> /CUSTOM/logs	press Return
3	vi EcDsStStagingDiskServer.ALOG or EcDsStStagingMonitorServer.ALOG	press Return

A.3.2.5 Check the Space Available in the Staging Area: Failure can also be caused by a lack of space in the staging area.

**Table A.3.2.5-1. Checking the Space Available in the Staging Area - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports the Archive Server subsystem using a generic SSIT account.	
2	cd /usr/ecs/<MODE> /CUSTOM/drp/<archivehost>/data/staging/user<#>	press Return
3	df -k .	press Return

## A.4 Failure During DPR Generation

### A.4.1 Description

The creation of a Data Processing Request is an essential part of the SSIT process. There are many reasons for DPR creation failure to occur. During DPR generation, the DPR generation executable will turn subscriptionFlag from zero to non-zero, which needs subscription Server up running and the executable will also query Science Data Server for input granules. The cause of failure to generate a DPR could come from following errors: 1) Incorrect information in PGE ODL and ESDT ODL files, which generally references to the PGE registration incorrectly. 2) Not all the required Servers up running properly, which generally reference to the System problem. 3) ESDTs required for the PGE were not properly installed. 4) Database queries failure, which generally reference to Database errors.

### A.4.2 Handling DPR Generation Failures:

To find out why a DPR generation fails, a user needs to look for the Production Request Editor ALOG file, which is EcPIPREditor.ALOG residing in the directory

/usr/ecs/<MODE>/CUSTOM/logs of PLS host. The ALOG file needs to be inspected for evidence of the source of a failure. In addition, that ALOG may indicate that the ALOG files for other subsystems, such as SDSRV or IOS, may contain entries describing the errors. Another useful resource for troubleshooting the failure is to look for PDPS database. All the PGE

registration information is kept in different tables. 1) Inside the table PIDataTypeMaster, there is a column called subscriptionFlag, during a DPR generation, the DPR executable will turn the subscriptionFlag for all the ESDTs needed for the PGE from zero to non-zero. If the Flag for the dataTypeId (corresponding to ESDT) did not turn to non-zero, that indicates subscription trouble. 2) Inside the table PIDataTypeMaster, there is a column called dataServUrString, during a DPR generation, the DPR executable will turn the dataServUrString for all the ESDTs needed for the PGE from NULL to UR value for Science Data Server. 3) If the PGE contains static files, the URs for the static files have to be included in PIDataGranuleShort table before a DPR can be successfully generated. 4) For dynamic granules, the UR values will become available in the PIDataGranuleShort table if the granules are available during the DPR generation, if the dynamic granules are not available during a DPR generation, it will not cause a failure to generate a DPR.

## A.5 Failure Scheduling a DPR

### A.5.1 Description

Problems scheduling a PGE for execution in the system occur when a DPR, scheduled through the Planning Workbench, does not get passed to Autosys.

### A.5.2 Handling a DPR Scheduling failure

There is ALOG file called EcPIWb.ALOG resided at /usr/ecs/<mode>/CUSTOM/utilities of PLS host, which should be inspected first.

A.5.2.1 Check the PDPS Data Base: PIDataProcessingRequest: During scheduling, the PDPS data base is updated to reflect a change in the state of the DPR. In the PDPS data base the PIDataProcessingRequest table will show a value of “NULL” in the completionState field if the DPR did not get passed to Autosys.

The table below lists the steps required to view the PDPS data base using a database viewer.

**Table A.5.2.1-1. Viewing the PDPS Data Base Using the Data Base Viewer GUI - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports PDPS using a generic SSIT account.	
2	(if necessary) setenv DISPLAY <machinename>:0.0	press Return
3	Dbrowse	press Return
4	enter into GUI login: <PDPSservername, e.g. p0pls02_svr>, <PDPSusername, e.g. pdps_role>, <PDPSpassword, e.g. welcome>	
5	select pdps_<MODE>	
6	select “sample data” under “view” menu	
7	select PIDataProcessingRequest	

The table below lists the steps required to view the PDPS database using SQL commands.

**Table A.5.2.1-2. Viewing the PDPS Database Using SQL Commands - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	login to AIT computer which supports PDPS using a generic SSIT account.	
2	isql -U<pdpsusername> -P<password> -S<pdpsserver>	press Return
3	use pdps_<MODE>	press Return
4	Go	press Return
5	select * from PIDataProcessingRequest where dprld = "<dprID>"	press Return
6	Go	press Return

A.5.2.2 Check the PDPS Database: PIDataGranuleShort: Input data granules which are ready for use by a PGE running in the system will have entries with full URLs in the PDPS database table PIDataGranuleShort, under universalReference. Standalone PGE, those not running as part of a PGE chain, needs to have the full URL entered. If the PGE is running as part of a PGE chain, then the input granules are products of preceding PGEs in the chain. If, in the ESDT ODL files prepared for those input granules, the dynamic flag is set to "external" ("E") instead of "internal" ("I"), then the DPR will go into Autosys as soon as the input granules become available. If the dynamic flag is set as "internal" then the DPR should go into Autosys regardless of the availability of the input granules. PGE execution will commence, and the color of the display in JobScope within Autosys will change, as soon as the input granules are available.

Inspection of the PDPS database follows the same procedures as outlined in above two tables, with PIDataGranuleShort as the table, and universalReference as the field. The individual granules can be identified by their data type (dataTypeId).

## A.6 Failures During Execution

### A.6.1 Description

PGEs scheduled for execution in the system follow three stages of processing. Each has its own types and causes of failure. Its always better to put the next stage on hold while executing the previous. Once you execute one stage successfully with Exit Code 0, then take the hold off for the next. Its easier to investigate if there are any failures. For example, when your PGE starts executing the first stage put second and third stage on hold. When your first stage executes successfully, then take the hold off from the second stage while third stage is still on hold. Again, if that goes though successfully then you can take the hold off from the third stage.

After execution of each stage, click on the stage and then click on Job Console button on the left side at the bottom to see the Exit Code of it. If it is 0 that indicates successful execution but if you get non-zero exit code that indicates failure.

A.6.1.1 Resource Allocation, staging and preprocessing: The first stage of PGE processing in Autosys involves Resource allocation, staging and preprocessing. If this fails, the ALOG file of the Data Processing host can be checked to see whether the PGEEXE.tar file was successfully acquired. If there is an acquire failure, further evaluation proceeds as outlined in A.3. Acquire Failure, above.

The log files DPR#.ALOG and DPR#.err are stored in /usr/ecs/<MODE>/CUSTOM/logs directory on PLS host, can be inspected to find out the cause of the failure.

Preprocessing rarely completely fails. It may not generate the system PCF file correctly. The system PCF can be inspected to see whether it matches expectations.

To view the system PCF, execute the following steps:

1. Login to the AIT computer which supports Data Processing.
2. At the UNIX prompt, type **cd /usr/ecs/<MODE>/CUSTOM/pdps/<hostname>/data/DpPrRm/<hostname>\_disk/<pgeId>/<dprId\_<hostname>**.
3. Open the system generated PCF file.

A.6.1.2 PGE Execution: This stage involves PGE execution.

Failures during PGE execution can be investigated using the Toolkit LogStatus file, the system PCF file, and by running the PGE from the command line using the environment set in the PGE profile file and PGS\_PC\_INFO\_FILE points to the system \*.Pcf. All of these are found in the runtime directory.

To inspect the LogStatus file, execute the following steps:

1. Login to the SGI computer which supports Data Processing.
2. At the UNIX prompt, type **cd /usr/ecs/<MODE>/CUSTOM/pdps/<hostname>/data/DpPrRm/<hostname>\_disk/<pgeId>/<dprId\_<hostname>**.

For execution of this particular stage, logs get written on the SGI in the /usr/ecs/<MODE>/CUSTOM/logs directory in a file DPR#.err. Find out the Exit Code of the PGE for this stage from this file. It will also give you some other messages regarding the failure which are useful for further investigation. Exit Code 252 indicates toolkit problem. That means either proper toolkit hasn't been linked or some files from toolkit are missing. Check auto.profile file to determine if PGSHOME variable is set to proper path. Exit Code 1 indicates problem in the inputs. For this type of error, check the system generated PCF file. Sometimes inputs and outputs doesn't get written properly with a correct logical ID in the system generated PCF. Compare system generated PCF with the PCF, which you used to run the command line test. If system generated PCF is not correct, then you need to edit the template PCF which system uses to generate the PCF file.

A.6.1.3 Postprocessing, destaging and deallocation: Postprocessing does not often fail, but it may show as a failure in Autosys if the Execution stage has failed. Should this occur, inspect the DPRID#.err file in the /usr/ecs/<MODE>/CUSTOM/logs on p0pls01.

Destaging involves the insertion of output data granules into SDSRV. Thus, section A.2, Insertions of Files to the Science Data Server, should be consulted for this case. Volume group has to be set up for all the inputs as well as outputs of the PGE in order to execute this stage successfully. If either or all volume groups are missing then this stage fails.

Rarely are there cases of failure in Deallocation. When it does occur, inspect the Data Processing ERR file following the same procedures described above.

This page intentionally left blank.

# Appendix B. Historical Background of ECS Release B.0 (Architectural Differences from Pre-Release B Testbed)

---

The process of SSI&T or integration of EOS Instrument Science Software into the ECS was developed and refined over three distinct iterations of ECS. IR1, the Pre-Release B Testbed and Release B.0. ECS Release B.0 became the at-launch baseline system supporting EOS-TERRA instruments. Numerous releases of ECS have been made since TERRA's launch and will continue to be made in support of AQUA (PM-1) and AURA (CHEM-1) satellites. Every attempt is made to keep integration procedures for science software consistent through succeeding releases of ECS; however, some changes in ECS releases do affect the SSI&T procedures and are thus documented herein for the latest ECS release.

Basic architectural differences were introduced with Release B.0 which led to significant variances in SSI&T for the Version 2 science algorithms from that of the Version 1 science algorithms into the Pre-Release B Testbed. This section originally served to alert the extended SSI&T staff of the impact of Release B.0 on SSI&T procedures with which they were familiar. Because of the long duration that Release B.0 has been operational, the variances are no longer of much relevance. None-the-less, for historical purposes, the material in this Appendix is included.

This section provides a synopsis of the architecture of the Pre-Release B Testbed and of the Release B ECS in Sections B.1 and B.2, respectively. A list of major SSI&T procedural variations is given in Section B.3.

## B.1 Pre-Release B Testbed Architecture: Overview

The Pre-Release B Testbed was made operational at GSFC, EDC and NSDIC DAACS primarily for the purpose of conducting SSI&T for EOS instrument team Version 1 software. Several software components were included, in order to smooth the progression to Version 2 integration into ECS:

- A Planning and Data Processing System (PDPS) provided SSI&T support for registering PGEs, setting up subscriptions for Planning, entering production requests, developing and activating a production plan, scheduling and monitoring jobs with the AutoSys COTS, managing PGE execution, archiving to data storage, and QA monitoring of products including use of EOSVIEW for visualization.
- A flat file data base, a.k.a. the Integrated Metastore Factory or IMF, which was interfaced with Planning and Data Processing components. This provided a 'data-server-like' interface, without which, Testbed integration procedures would bear little resemblance to later B.0 procedures. The IMF supported metadata validation, supported subscription registration and notification, and allowed ESDT additions via a GUI interface.
- An "out-of-the-box" version of HP OpenView was provided for monitoring hardware failures.

From the viewpoint of SSI&T, the most critical cause of procedural differences between the Testbed and Release B.0 is the IMF. The IMF is a UNIX file system-based archive simulator that provides basic science data management functionality. The system is not a server or a persistent daemon process. Rather, it was designed as a client-only solution with a standard B.0 ECS Data Server/Client interface. The system supported interfaces for inserting and retrieving science data and any associated ancillary products. Temporal based query and metadata inspect/update operations were also supported. The system also had a simplified file-based asynchronous notification mechanism for data insert events.

The IMF archive was based on a UNIX file system. Each ESDT consists of a directory and a descriptor file that describes the ESDT. Data granules were stored in each ESDT directory at lower level in the structure. Each directory then, represented a collection. Also present in the archive was a data dictionary file that contained a superset of all valid ESDT attributes. The descriptor and the data dictionary were processed during data insert operation for metadata validation purposes. For asynchronous notification of data insert events, a data handle otherwise known as the Universal Reference (UR) file was written to a built-in repository directory.

The implementation of the IMF as a flat file system, while allowing ECS SSI&T procedures to be conducted on Science PGEs, implies major procedural differences in SSI&T for Release B.0. These differences are listed in Section B.3.

## **B.2 Release B.0 Architecture: Overview**

The Release B.0 architecture can be grouped into the following four categories:

- Data storage and management is provided by the Data Server Subsystem (DSS), with the functions needed to archive science data, search for and retrieve archived data, manage the archives, and stage data resources needed as input to science software or resulting as output from their execution. The Data Server Subsystem provides access to earth science data in an integrated fashion through an Application Programming Interface (API) that is common to all layers.
- Information search and data retrieval is provided by the science user interface functions in the Client Subsystem (CLS), by information search support functions in the Data Management Subsystem (DMS), and by capabilities in the Interoperability Subsystem (IOS) which assist users in locating services and data. These subsystems comprise what is referred to as the “Pull” side of ECS.
- Data processing is provided by the Data Processing Subsystem (DPS) for the science software, and by capabilities for long and short term planning of science data processing, as well as by management of the production environment provided by the Planning Subsystem (PLS). Routine data processing and re-processing will occur in accordance with the established production plans. In addition ECS will provide "on-demand processing", where higher level products are produced only when there is explicit demand for their creation.
- Data ingest is provided by the Ingest Subsystem (INS), which interfaces with external applications and provides data staging capabilities and storage for an approximately 1-year buffer of Level 0 data (so that reprocessing can be serviced from local storage). The INS is

designed to accommodate a potentially large number of external interfaces. It is also designed to provide very diverse functions, such as high-volume ingest of level 0 data and low-volume ingest of data from field campaigns. Data Ingest and data processing comprise what is known as the “Push” side of ECS.

Figure B.1 provides a context diagram for ECS B.0 subsystems.



### **B.2.2 Interoperability Subsystem (IOS)**

The Interoperability subsystem provides an advertising service. It maintains a database of information about the services and data offered by ECS, and provides interfaces for searching this database and for browsing through related information items. For example, ESDTs, are made visible through the advertising service. ESDTs are objects which contain metadata describing data granules and the services which the system can apply to those granules. The Client Subsystem provides the user interface which enables access to the IOS.

### **B.2.3 Data Management Subsystem (DMS)**

The Data Management subsystem provides three main functions:

- Provide end-users with a consolidated logical view of a distributed set of data repositories.
- Allow end-users to obtain descriptions for the data offered by these repositories. This also includes descriptions of attributes about the data and the valid values for those attributes.
- Provide data search and access gateways between ECS and external information systems.

### **B.2.4 Data Server Subsystem (DSS)**

The Data Server subsystem provides the management, cataloging, access, physical storage, distribution functions for the ECS earth science data repositories, consisting of science data and their documentation. The Data Server provides interfaces for other ECS subsystems which require access to data server services. The Data Server Subsystem consists of the following principal design components:

- Database Management System - The Data Server subsystem will use database technology to manage its catalog of earth science data, and for the persistence of its system administrative and operational data.
- Data Type Libraries - The Data Server will use custom dynamic linked libraries (DLLs) to provide an means of implementing the variety of ECS earth science data types and services, and will provide a interface for use by other ECS subsystems requiring access to those services and data.
- File Storage Management System - This component provides archival and staging storage for data.
- Distribution System - The Data Server provides the capabilities needed to distribute bulk data via electronic file transfer or physical media.

### **B.2.5 Ingest Subsystem (INS)**

This subsystem handles the initial reception of all data received at an ECS facility and triggers subsequent archiving and processing of the data. The ingest subsystem is organized into a collection of software components (e.g., ingest management software, translation tools, media handling software) from which those required in a specific situation can be readily configured.

The resultant configuration is called an ingest client. Ingest clients can operate on a continuous basis to serve a routine external interface; or they may exist only for the duration of a specific ad-hoc ingest task. The ingest subsystem also standardizes on a number of possible application protocols for negotiating an ingest operation, either in response to an external notification, or by polling known data locations for requests and data.

### **B.2.6 Data Processing Subsystem (DPS)**

The main components of the data processing subsystem - the science algorithms or Product Generation Executives (PGEs) - will be provided by the science teams. The data processing subsystem provides the necessary hardware resources, as well as a software environment for queuing, dispatching and managing the execution of these algorithms. The processing environment will be highly distributed and will consist of heterogeneous computing platforms. The AutoSys COTS tool is used as the scheduling engine. This DPS-encapsulated COTS is designed to manage production in a distributed UNIX environment. The DPS also interacts with the DSS to cause the staging and de-staging of data resources in synchronization with processing requirements.

### **B.2.7 Planning Subsystem (PLS)**

The Planning Subsystem provides the functions needed to plan routine data processing, schedule on-demand processing, and dispatch and manage processing requests. The subsystem provides access to the data production schedules at each site, and provides management functions for handling deviations from the schedule to operations and science users. The Planning subsystem provides several functions to account for:

- A processing environment which will be highly distributed and consist of heterogeneous computing platforms.
- Existence of inter-site and external data dependencies.
- Dynamic nature of the data and processing requirements of science algorithms.
- High system availability.
- Provision of a resource scheduling function, which can accommodate hardware technology upgrades.
- Support for on-demand processing (as an alternative to predominantly routine processing).
- Ability to provide longer-term (e.g., monthly) processing predictions as well as short term (e.g., daily) planning and scheduling

### **B.2.8 Communications Subsystem (CSS)**

The CSS helps manage the operation of distributed objects in ECS, by providing a communications environment. The environment allows software objects to communicate with each other reliably, synchronously as well as asynchronously, via interfaces that make the

location of a software object and the specifics of the communications mechanisms transparent to the application.

In addition, CSS provides the infrastructure services for the distributed object environment. They are based on the Distributed Computing Environment (DCE) from the Open Software Foundation (OSF). DCE includes a number of basic services needed to develop distributed applications, such as remote procedure calls (rpc), distributed file services (DFS), directory and naming services, security services, and time services.

Finally, CSS provides a set of common facilities, which include legacy communications services required within the ECS infrastructure and at the external interfaces for file transfer, electronic mail, bulletin board and remote terminal support. The Object Services support all ECS applications with interprocess communication and specialized infrastructure services such as security, directory, time, asynchronous message passing, event logging, lifecycle service, transaction processing and World Wide Web (WWW) service.

### **B.2.9 Management Subsystem (MSS)**

The Management Subsystem (MSS) provides enterprise management (network and system management) for all ECS resources: commercial hardware (including computers, peripherals, and network routing devices), commercial software, and custom applications. With few exceptions, the management services are decentralized, such that no single point of failure exists.

MSS provides two levels of an ECS management view: the local (site/DAAC specific) view, provided by Local System Management (LSM), and the enterprise view, provided by the Enterprise Monitoring and Coordination (EMC) at the SMC. Enterprise management relies on the collection of information about the managed resources, and the ability to send notifications to those resources. For network devices, computing platforms, and some commercial off-the-shelf software, MSS relies on software called "agents" which are usually located on the same device/platform and interact with the device's or platform's control and application software, or the commercial software product. However, a large portion of the ECS applications software is custom developed, and some of this software - the science software - is externally supplied. For these components, MSS provides a set of interfaces via which these components can provide information to MSS (e.g., about events which are of interest to system management such as the receipt of a user request or the detection of a software failure). These interfaces also allow applications to accept commands from MSS, provided to MSS from M&O consoles (e.g., an instruction to shut down a particular component). Applications which do not interact with MSS directly will be monitored by software which acts as their "proxies". For example, the Data Processing Subsystem (DPS) acts as the proxy for the science software it executes. DPS notifies MSS of events such as the dispatching or completion of a PGE, or its abnormal termination.

MSS uses HP OpenView as its system management tool. The information collected via the MSS interfaces from the various ECS resources is consolidated into an event history database, some on a near real-time basis, some on a regular polling basis (every 15-to 30 minutes), as well as on demand, when necessitated by an operator inquiry. The database is managed by Sybase, and Sybase query and report writing capabilities are used to extract regular and ad-hoc reports from

it. Extracts and summaries of this information will be further consolidated on a system wide basis by forwarding it to the SMC (also on a regular basis).

MSS provides fault and performance management and other general system management functions such as security management (providing administration of identifications, passwords, and profiles); configuration management for ECS software, hardware, and documents; Billing and Accounting; report generation; trending; request tracking; and mode management (operational, test, simulation, etc.).

### **B.2.10 Internetworking Subsystem (ISS)**

The ISS provides local area networking (LAN) services at ECS installations to interconnect and transport data among ECS resources. The ISS includes all components associated with LAN services including routing, switching, and cabling as well as network interface units and communications protocols within ECS resources.

The ISS also provides access services to link the ECS LAN services to Government-furnished wide-area networks (WANs), point-to-point links and institutional network services. Examples include the NASA Science Internet (NSI), Program Support Communications Network (PSCN), and various campus networks "adjoining" ECS installations.

## **B.3 Implications for SSI&T Procedures**

The Pre-Release B Testbed was used as an integration environment, for EOS instrument team Version 1 software, from May, 1997 through November, 1997. The Release B.0 system was used after January, 1998 to integrate Version 2, or launch-ready instrument processing software for EOS TERRA (AM-1). Architectural differences between the two ECS releases will cause certain SSI&T procedural differences. The major differences are due to the presence of Ingest and Data Server Subsystems in B.0. Table B.1 list the major differences affecting procedures, which will be encountered by SSI&T staff, in integrating Science Version 1 and Version 2 software.

**Table B.1. Major SSI&T Procedural Differences: Testbed to B.0 Releases**

<b>Function</b>	<b>Pre-Release B Testbed</b>	<b>Release B.0</b>
System Operation	Run custom binaries, COTS (AutoSys); no servers. non-DCE	All servers must run and communicate with each other; bring up manually, or use ECSAssistant tool
Ingest Ancillary Data Granules	SSI&T manager tool, EDSTs visible in IMF	Ingest GUI, EDSTs must be visible to IOS server
ESDT Insert	SSI&T manager tool	Use Ingest tool
ESDT Verification	View of flat file database	Verify through IOS
DAP, SSAP Insert	Direct copy	Use Ingest
PDPS Database Population	Basic and advanced temporal production rules	More attributes, additional production rules
PGE Operation	When all data available in IMF, DPR Activated in AutoSys Automatic reprocessing facility for failed PGE. Simple chaining per Production Plan	When all data is available in SDSRV; DPR activated. No automatic reprocessing  Complex chaining through additional production rules.
File Access	Copy from IMF, single site	Verify presence through IOS; ftp from SDSRV; access to multiple sites
Multi-file Granule Support	Files inserted separately, accessed as a single granule	Files inserted together, accessed as a single granule
Subscription Management	Subscription Reaper - non-DCE version	Subscription Manager

This page intentionally left blank.

# Appendix C. Examples of the Various ODL Files Used by Each Instrument Team

---

Section 12 deals, in part, with the use of ODL files in SSI&T activities. This section serves as a supplement and reference for that section. Useful examples of ODL files will follow. ODL Template files, from which specific examples were created, will be listed first. Then, examples of specific ODL files will be listed by instrument (ASTER, MISR, MODIS and AIRS). *Please note that in many of the examples that follow, much of the instrument/ECS provided comments have been deleted in order to keep this document reasonably short.*

## C.1 Template ODL Files

There are five Template ODL files listed here. The specific or tailored ODL files listed in Sections C.2 through C.5 were derived from these templates by appropriate editing and filling-in of values (*\*NOTE: while the TILE ODL file is currently not being used by any of the instrument teams mentioned above, the template is included here for completeness*). The five ODL Template files listed reside, on the AIT Sun host, at /usr/ecs/<mode>/CUSTOM/data/DPS. They are:

- PGE\_ODL.template
- ESDT\_ODL.template
- ORBIT\_ODL.template
- TILE\_ODL.template\*
- PATHMAP\_ODL.template

### C.1.1 PGE\_ODL.template

```
/*
/*****
/*
/*          TEMPLATE PGE SCIENCE METADATA ODL FILE          */
/*
/*
/*
/*          */
/* The SSIT operator's responsibility is to copy this file over and */
/* edit it to add all necessary PDPS metadata values.          */
/*          */
/* All PGE ODL files must reside in directory $DPAT_PGE_SCIENCE_MD. */
/* This directory is now set through the Process Framework CFG files. */
/*          */
/* The operator must add a value to the right of the "=" for each */
/* parameter.          */
/*          */
/*          */
/* Normally, a template version of this file (without the comments) */
/* will be generated by the SSIT operator from the PCF delivered to */
/* SSIT. This file is meant to show the SSIT personnel and the */
/* Instrument teams the information that is needed for a PGE to be */
/* planned and executed by the Planning and Data Processing system of */
/* ECS.          */
/*          */
/*          */
/* CHANGE LOG          */
/* -- Added new schedule type for Data Scheduled PGEs. 02/18/98 */
```

```

/*          Changed QUERY_DELAY to be optional for all PGEs.          */
/*          Changed SPATIAL_KEY_INPUT to KEY_INPUT                    */
/* -- Fixed description for Begin/End Period Offsets.      03/10/98  */
/* -- Added The Distinct Value definition.                 03/26/98  */
/* -- Fixed length of CATEGORY.                            03/27/98  */
/* -- Fixed length of FILETYPE_NAME.                       03/31/98  */
/* -- Added PATHMAP_NAME                                   04/13/98  */
/* -- Changed how WAITFOR is supposed to be set.          05/06/98  */
/*          Added entries for ASSOCIATED_SCIENCE_DATA to handle */
/*          BROWSE and QA products.                        */
/* -- Added START_OF_MINUTE to PROCESSING_BOUNDARY.       06/24/98  */
/*          Updated DATA DAY values for PGE_PARAMETER_DYNAMIC_VALUE, */
/*          and DATABASE_QUERY                             */
/* -- Added KEY_PARAMATER_NAME and KEY_PARAMETER_VALUE    07/05/98  */
/*          for Metadata Checks and Metadata Queries.     */
/* -- Updated description for KEY_PARAMETER_NAME and      07/11/98  */
/*          KEY_PARAMETER_VALUE.                           */
/* -- Updated lengths for PLATFORM and INSTRUMENT.       08/13/98  */
/* -- Updated explanation for "Already Created Tile"     08/18/98  */
/*          for QUERY_TYPE.                                 */
/* -- Added CHECK_FOR_OUTPUT flag.                        08/24/98  */
/* -- Added MOST_RECENT_QUERY_OFFSET and MOST_RECENT_    09/02/98  */
/*          QUERY_RETRIES parameters for the Most Recent Granule */
/*          Production Rule.                               */
/* -- Added AUXILIARY_LOGICAL_ID object for handling     09/23/98  */
/*          multiple L0 granules.                          */
/*          Removed older change commentary.              */
/* -- Added COMPOUND_PGE parameter for handling          10/23/98  */
/*          PGEs with multiple executables. Also deleted */
/*          old change history.                            */
/* -- Updated description for ALTERNATE INPUT TIMER      11/07/98  */
/*          to say that it has not affect for Dynamic    */
/*          Internal ESDTs.                               */
/* -- Added ALIGN_DPR_TIME_WITH_INPUT_TIME parameter    12/20/98  */
/* -- Increased the number of Profile Ids from 99 to    07/12/99  */
/*          999.                                          */
/*          Removed restriction on ALTERNATE_INPUT_TIMER with respect */
/*          to Internal Dynamic ESDTs.                    */
/*          Added PGE_DEFAULT_PROFILE parameter.         */
/*          Added PROFILE_SELECTOR_PGE_PARAMETER.       */
/* -- Added Closest Granule values:                      08/19/99  */
/*          CLOSEST_QUERY_OFFSET, CLOSEST_QUERY_RETRIES, and */
/*          CLOSEST_QUERY_DIRECTION.                     */
/* -- Added "Metadata" to the query type.               12/13/99  */
/* -- Updated Toolkit logical Ids that SSIT allows      02/03/00  */
/*          in ODL.                                       */
/*          Removed old change history                    */
/*          */
/*****

```

```

/*****
/*          PGE name                                          */
/*          -- Must be a string, max len 10 characters      */
/*          -- PGE name inside ODL file must be identical to */
/*          PGE name used as part of ODL filename         */
/*          Example                                          */
/*          PGE_NAME = "ssit"                               */
/*****

```

PGE\_NAME = " "

```

/*****
/*      PGE version                                */
/*      -- Must be a string, max len 5 characters */
/*      -- PGE version inside ODL file must be identical to */
/*      PGE version used as part of ODL filename */
/*      Example                                    */
/*      PGE_VERSION = "1.0"                        */
*****/

```

PGE\_VERSION = ""

```

/*****
/*      PGE Profile ID                            */
/*      -- Must be an integer                    */
/*      -- Must be >= 0 and <= 999              */
/*      Example                                    */
/*      PROFILE_ID = 99                          */
*****/

```

PROFILE\_ID =

```

/*****
/*      PGE Profile Description                    */
/*      -- Must be a string, max length 255 characters */
/*      Example                                    */
/*      PROFILE_DESCRIPTION = "Improved performance numbers" */
*****/

```

PROFILE\_DESCRIPTION = ""

```

/*****
/*      PGE On-Demand Profile Default              */
/*      -- Must be a string, set to "Y" or "N".   */
/*      -- If NOT Present, defaults to "N".       */
/*      -- Marks a particular for this PGE (PGE Name + */
/*      PGE version) as the default for On Demand Processing */
/*      Requests.                                    */
/*      -- If more than 1 PGE (PGE Name + PGE Version) has this */
/*      value set, an error will be returned.     */
/*      Example                                    */
/*      PGE_DEFAULT_PROFILE = "N"                 */
*****/

```

PGE\_DEFAULT\_PROFILE = ""

```

/*****
/*      Spacecraft platform name                  */
/*      -- Must be a string, max len 25 characters */
/*      Example                                    */
/*      PLATFORM = "TRMM"                          */
*****/

```

PLATFORM = ""

```

/*****
/*      Instrument name                            */
/*      -- Must be a string, max len 20 characters */
/*      Example                                    */
/*      INSTRUMENT = "CERES"                       */
*****/

```

```

INSTRUMENT = ""

/*****
/*      Minimum Number of Outputs                                */
/*      (used for QA purposes)                                  */
/*      -- Must be a integer, maxium 3 digits.                  */
/*      Example                                                 */
/*      MINIMUM_OUTPUTS = 0                                     */
*****/

```

```

MINIMUM_OUTPUTS =

/*****
/*      Type of PGE Scheduling                                  */
/*      -- Must be a string with one of the following values:  */
/*      "Time" = TimeScheduled (PGE is scheduled based on the  */
/*      boundary/period and the arrival of data).              */
/*      "Data" = DataScheduled (PGE is scheduled based on the  */
/*      avialability of data produced by other                 */
/*      PGEs).                                                 */
/*      "Tile" = TileScheduled (PGE is scheduled based on the  */
/*      the definition of Tiles). Note that                    */
/*      TILE_SCHEME_NAME must have a value for Tile           */
/*      Scheduled PGEs.                                        */
/*      "Orbit" = OrbitScheduled (PGE is scheduled based      */
/*      the orbit of the spacecraft. Note that then           */
/*      PROCESSING_PERIOD must = "ORBITS=1" and               */
/*      PROCESSING_BOUNDARY must =                             */
/*      "START_OF_ORBIT". Also, A file of named                */
/*      ORBIT_<platform>.odl must be present.                  */
/*      Also if you want a Pathmap it needs to be              */
/*      specified under PATHMAP_NAME.                          */
/*      "Snapshot" = SnapshotScheduled (PGE is scheduled      */
/*      based on a single date/time entered                    */
/*      entered when the production request is                 */
/*      submitted.                                             */
/*      Example                                                 */
/*      SCHEDULE_TYPE = "Tile"                                  */
*****/

```

```

SCHEDULE_TYPE = ""

/*****
/*      Nominal time interval between start of PGE runs        */
/*      -- NOT needed for PGEs where SCHEDULE_TYPE = "Snapshot" */
/*      or SCHEDULE_TYPE = "Data".                              */
/*      -- Must contain a single P=V string, where             */
/*      P is one of { YEARS, MONTHS, THIRDS WEEKS, DAYS,      */
/*      HOURS, MINS, SECS, ORBITS}                             */
/*      -- NOTE that ORBITS must be used for PGEs based on an */
/*      Orbit Model. Note that PROCESSING_BOUNDARY must be    */
/*      set to "START_OF_ORBIT".                                */
/*      Example                                                 */
/*      PROCESSING_PERIOD = "DAYS=1"                            */
*****/

```

```

PROCESSING_PERIOD = ""

/*****
/*      Nominal time boundary on which PGE processing begins  */
*****/

```

```

/*          -- NOT needed for PGEs where SCHEDULE_TYPE = "Snapshot" */
/*          or SCHEDULE_TYPE = "Data".                               */
/*          -- Must contain a one of                               */
/*          { START_OF_MINUTE, START_OF_HOUR, START_OF_6HOUR,      */
/*            START_OF_DAY, START_OF_WEEK,                        */
/*            START_OF_ONE_THIRD_MONTH,                          */
/*            START_OF_MONTH, START_OF_YEAR, START_DATE,         */
/*            START_OF_ORBIT };                                   */
/*          also, "+<n>" or "-<n>" may be added to any of these,  */
/*          where <n> specifies integer seconds.                  */
/*          For START_DATE an "=" can be added followed by the   */
/*          start date.                                           */
/*          -- NOTE that START_OF_ORBIT must be used for PGEs based */
/*          on an Orbit Model. A file of named                     */
/*          ORBIT_<platform>.odl must be present.                 */
/*          Example                                               */
/*          PROCESSING_BOUNDARY = "START_OF_HOUR"                 */
/*****

```

PROCESSING\_BOUNDARY = ""

```

/*****
/*          Software version                                     */
/*          -- Must be a string, max 5 len characters             */
/*          -- If Ssw version is not the same as PGE version,   */
/*          SswId ("<PGE Name>#<Ssw Version>") must already      */
/*          be defined in the database;                           */
/*          That is, the only allowed values of the              */
/*          software version are either this PGE version         */
/*          or a previous PGE version for this PGE name         */
/*          Example                                               */
/*          PGE_SSW_VERSION = "1.0"                               */
/*****

```

PGE\_SSW\_VERSION = ""

```

/*****
/*          Delay for query                                     */
/*          -- Optional for types of PGEs.                       */
/*          -- The amount of time (in SECONDS) that the query for */
/*          input data should be delayed. This value is added   */
/*          onto the Stop Time of any DPR generated with this   */
/*          PGE.                                                 */
/*          -- Used for Tiling or Metadata Query inputs.        */
/*          -- OPTIONAL Parameter. If not specified it is set to 0. */
/*          -- Must be an integer value >= 0.                   */
/*          Example                                               */
/*          QUERY_DELAY = 360 (1 hour)                           */
/*          */
/*****

```

QUERY\_DELAY = 0

```

/*****
/*          Name of the Tiling Scheme used                       */
/*          -- Must be a string of at most 20 characters.       */
/*          -- There can be NO spaces in the string.            */
/*          -- A file that defines the Tiling Scheme must       */
/*          be created with the name TILE_<tiling scheme>.odl   */
/*          Example                                               */
/*          TILE_SCHEME_NAME = "Earth_Squared"                   */
/*****

```

```

/*
/* NOTE that this is only needed for PGEs of Schedule Type = "Tile".
/* It can be deleted for all other types of PGEs.
/*****

TITLE_SCHEME_NAME = ""

/*****
/* Name of Pathmap used
/* -- Must be a string of at most 25 characters.
/* -- There can be NO spaces in the string.
/* -- A file that defines the Pathmap must
/* be created with the name PATHMAP_<Pathmap_Name>.odl
/* Example
/* PATHMAP_NAME = "Some_Name"
/*
/* NOTE that this is only needed for PGEs of Schedule Type = "Orbit".
/* It can be deleted for all other types of PGEs.
/*****

PATHMAP_NAME = ""

/*****
/* OPTIONAL PARAMETER
/* Check For Outputs
/* -- Must be a character value of either "Y" (YES)
/* or "N" (NO).
/* -- Defaults to "N" if not specified.
/* -- When set to "Y", this means that a DPR of the PGE
/* will ONLY be scheduled if the output of that PGE has
/* NOT been produced. This is currently planned for use
/* in ASTER Routine Processing.
/* -- Note that creating a DPR (in the Production Request
/* Editor) with Reprocessing set will override this
/* flag.
/* Example
/* CHECK_FOR_OUTPUTS = "N"
/*****

CHECK_FOR_OUTPUTS = "N"

/*****
/* OPTIONAL PARAMETER
/* Compound Pge Flag
/* -- Must be a character value of either "Y" (YES)
/* or "N" (NO).
/* -- Defaults to "N" (Not Compound PGE) if not specified.
/* -- When set to "Y", this means that this PGE is made up
/* of multiple executables AND that the output of one
/* of these executables is the input of another
/* executable within the PGE.
/* -- Note that setting this flag will hurt the performance
/* of the Destaging step during PGE execution. It is
/* best to only set it to "Y" if both conditions
/* mentioned above are true.
/* Example
/* COMPOUND_PGE = "N"
/*****

COMPOUND_PGE = "N"

```

```

/*****
/* Exit message object */
/*
/* Defines a possible PGE exit code, and associates a message with it. */
/*
/* This object is optional and can be deleted if no EXIT MESSAGEs are */
/* desired. */
/*
/* Replicate the object as needed to define EXIT MESSAGEs for multiple */
/* EXIT CODEs. */
/*
/* See "Establishing Science Software Exit Conditions for the */
/* Production Environment" white paper (420-WP-006-002) for the */
/* definitions and of exit code values and their uses. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

OBJECT = EXIT_MESSAGE

/*****
/*      Class (object counter, used only to distinguish objects) */
/*      -- Must be an integer */
/*      -- Must be unique in this file for this type of object */
/*      -- Must be greater than 0. */
/*      Example */
/*      CLASS = 1 */
/*****

CLASS= 1

/*****
/*      Exit code for this PGE */
/*      -- Must be an integer */
/*      -- Must be 0 or between 200 and 239 */
/*      Example */
/*      EXIT_CODE = 200 */
/*****

EXIT_CODE = 0

/*****
/*      Message corresponding to this exit code */
/*      -- Must be a string, max len 240 characters */
/*      Example */
/*      EXIT_MESSAGE = "PGE successfully completed" */
/*****

EXIT_MESSAGE = ""

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

END_OBJECT = EXIT_MESSAGE

/*****
/* Exit dependency object */
/*
/* Defines names, exit codes and conditions of PGEs on which this */
/* PGE depends. */

```

```

/*                                                    */
/* This object is optional and can be deleted if no EXIT DEPENDANCY(s) */
/* exist for this PGE.                                                    */
/*                                                    */
/* Replicate this object as needed to define multiple EXIT                */
/* DEPENDANCies for the PGE.                                              */
/*                                                    */
/* See "Establishing Science Software Exit Conditions for the              */
/* Production Environment" white paper (420-WP-006-002) for the          */
/* definitions and of exit code values and their uses.                    */
/*                                                    */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present              */
/*****
OBJECT = EXIT_DEPENDENCY

/*****
/*          Class (object counter, used only to distinguish objects)      */
/*          -- Must be an integer                                          */
/*          -- Must be unique in this file for this type of object       */
/*          -- Must be greater than 0.                                     */
/*          Example                                                         */
/*          CLASS = 1                                                       */
/*****

CLASS= 1

/*****
/*          Name of PGE upon which this PGE depends                      */
/*          -- Must be a string, max len 10 characters                    */
/*          -- SswId ("#<Ssw version>") must be different      */
/*          than this SswID (PGE cannot depend on itself)                */
/*          -- SswId must already exist in the database                   */
/*          Example: This CERES PGE depends on the exit code of          */
/*          a MODIS PGE: execute the CERES PGE only if the                */
/*          MODIS PGE had exit code = 0                                   */
/*          DEPENDENCY_PGE_NAME = "MODIS"                                  */
/*****

DEPENDENCY_PGE_NAME = ""

/*****
/*          Version of Ssw upon which this Ssw depends                    */
/*          -- Must be a string, max len 5 characters                      */
/*          -- SswId ("#<Ssw version>") must be different      */
/*          than this SswID (PGE cannot depend on itself)                */
/*          Example                                                         */
/*          DEPENDENCY_SSW_VERSION = "x"                                    */
/*****

DEPENDENCY_SSW_VERSION = ""

/*****
/*          Operator for exit code dependency condition                    */
/*          -- Must be one of { >, <, >=, <=, =, != }                      */
/*          Example                                                         */
/*          EXIT_OPERATION = "="                                           */
/*****

EXIT_OPERATION = ""

```

```

/*****/
/*      Exit code for PGE upon which this PGE depends      */
/*      -- Must be an integer                               */
/*      -- Must be 0 or between 200 and 239                */
/*      -- Must already exist in the database as a valid   */
/*      exit code for the PGE upon which this PGE depends */
/*      Example                                             */
/*      EXIT_CODE = 0                                       */
/*****/

EXIT_CODE = 0

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****/

END_OBJECT = EXIT_DEPENDENCY

/*****/
/* PCF entry object                                         */
/*                                                         */
/* The program DpAtCreateOdlTemplate (run at SSIT) generates one of */
/* these object for each file entry in the PCF. Only generic Toolkit */
/* Logical IDs are ignored during Template Creation.           */
/*                                                         */
/* The operator needs to fill in values for the parameters as described */
/* in the comments for each parameter. Note that some parameters */
/* must be filled for each PCF entry, while others are optional or only */
/* needed based on the values of other parameters.           */
/*                                                         */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED                   */
/*****/

OBJECT = PCF_ENTRY

/*****/
/*      Class (object counter, used only to distinguish objects) */
/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is normally not modified          */
/*      -- Must be an integer                               */
/*      -- Must be unique in this file for this type of object */
/*      -- Must be greater than 0.                         */
/*      Example                                             */
/*      CLASS = 1                                           */
/*****/

CLASS = 1

/*****/
/*      PCF logical ID                                         */
/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is normally not modified.          */
/*      -- Must be a positive integer.                       */
/*      -- Most values between 10000 and 10999 (Toolkit specific */
/*      Logical IDs) are ignored except for the following: */
/*      Data Dictionary Logical ID (10251)                  */
/*      Attitude Data Logical ID (10501)                   */
/*      Ephmerous Data Logical ID (10502)                  */
/*      Math Constant Logical ID (10999)                   */
/*      Index Data File Logical ID (10900)                 */
/*      DEM Logical Ids (10649 - 10655)                    */

```

```

/*          Ascii Dump Logical ID (10255)          */
/*          Disable Status Level RTI Logical ID (10117)      */
/*          Disable Seed RTI Logical ID (10118)          */
/*          Disable Status code RIT Logical ID (10119)      */
/*          Example                                          */
/*          LOGICAL_ID = 100                                */
/*****/

LOGICAL_ID = 100

/*****/
/*          PCF file type                                  */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is normally not modified      */
/*          -- Must be an integer between 1 and 8 inclusive */
/*          =1, PRODUCT INPUT FILES                        */
/*          =2, PRODUCT OUTPUT FILES                      */
/*          =3, SUPPORT INPUT FILES                       */
/*          =4, SUPPORT OUTPUT FILES                     */
/*          =5, USER DEFINED RUNTIME PARAMETERS          */
/*          =6, INTERIM/INTERMEDIATE INPUT FILES         */
/*          =7, INTERIM/INTERMEDIATE OUTPUT FILES        */
/*          =8, TEMPORARY I/O                            */
/*          Example                                          */
/*          PCF_FILE_TYPE = 1                                */
/*****/

PCF_FILE_TYPE = 1

/*****/
/*          Data Type Name -- same as Data Server ESDT Short Name */
/*          -- Must be a string, max len 8 characters          */
/*          -- Required for all PCF ENTRY objects, except those with */
/*          PCF_FILE_TYPE = 5 or 8                            */
/*          -- An ESDT ODL file for this name must exist in      */
/*          in directory $DPAT_ESDT_SCIENCE_MD, and have a name */
/*          of the form                                          */
/*          "ESDT_<Data Type Name#Data Type Version>.odl" */
/*          -- An ESDT of this Short Name must already be defined */
/*          at the Data Server                                  */
/*          Example                                          */
/*          DATA_TYPE_NAME = "TRWp1182"                    */
/*          implies file $DPAT_ESDT_SCIENCE_MD/ESDT_TRWpca1182.odl */
/*          already exists in the SSIT environment, and that    */
/*          ESDT Short Name "TRWp1182" already exists in the   */
/*          Data Server                                          */
/*****/

DATA_TYPE_NAME = ""

/*****/
/*          Data Type Version                              */
/*          -- Must be a string, max len 5 characters        */
/*          -- Required for all PCF ENTRY objects, except those with */
/*          PCF_FILE_TYPE = 5 or 8                            */
/*          -- An ESDT ODL file for this name must exist in      */
/*          in directory $DPAT_ESDT_SCIENCE_MD, and have a name */
/*          of the form                                          */
/*          "ESDT_<Data Type Name#Data Type Version>.odl" */
/*          -- An ESDT of this Short Name and Version must already */
/*          be defined at the Data Server                      */

```

```

/*          Example                                     */
/*          DATA_TYPE_VERSION = "3.5.1"             */
/*****

DATA_TYPE_VERSION = ""

/*****
/*          Minimum number of input granules for this logical ID          */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is only modified if the PGE can          */
/*          execute successfully with fewer granules than in the          */
/*          PCF from which the template was generated.                  */
/*          -- Used to support "Minimum Number of Granules"          */
/*          Production Rule.                                           */
/*          -- Required for all PCF ENTRY objects                      */
/*          PCF_FILE_TYPE = 1, 3, 6 (ignored otherwise).              */
/*          -- Must be a >= 0.                                         */
/*          -- Note that for number of files within a granule          */
/*          greater than one, the FILE TYPE object for this entry      */
/*          must be changed to specify the various file types and      */
/*          maximum number of files.                                    */
/*          Example                                                    */
/*          MIN_GRANULES_REQUIRED = 1                                  */
/*****

MIN_GRANULES_REQUIRED = 1

/*****
/*          Maximum number of input granules for this logical ID          */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is only modified if the PGE can          */
/*          execute successfully with more granules than in the          */
/*          PCF from which the template was generated.                  */
/*          -- Used to support "Minimum Number of Granules"          */
/*          Production Rule.                                           */
/*          -- Required for all PCF ENTRY objects                      */
/*          PCF_FILE_TYPE = 1, 3, 6 (ignored otherwise)              */
/*          -- Must be a positive integer                              */
/*          -- Note that for number of files within a granule          */
/*          greater than one, the FILE TYPE object for this entry      */
/*          must be changed to specify the various file types and      */
/*          maximum number of files.                                    */
/*          Example                                                    */
/*          MAX_GRANULES_REQUIRED = 1                                  */
/*****

MAX_GRANULES_REQUIRED = 1

/*****
/*          Begin Period Offset.                                         */
/*          -- Only needed if data for this PCF entry is to be          */
/*          selected BEFORE (-) or AFTER (+) the period defined          */
/*          for the ESDT (stated in the corresponding ESDT              */
/*          ODL file).                                                  */
/*          -- Defaulted to 0.                                          */
/*          -- If set, must be an integer number of seconds.          */
/*          A positive value indicates that the value is BEFORE          */
/*          the Period of the ESDT. A Negative value is added to          */
/*          the Period so that the data will be found after the          */
/*          start of the period specified for the ESDT.                */
/*          Example                                                    */

```

```

/*          BEGIN_PERIOD_OFFSET = "7200" (2 hours)          */
/*****/

    BEGIN_PERIOD_OFFSET = 0

/*****/
/*          End Period Offset.          */
/*          -- Only needed if data for this PCF entry is to be          */
/*          selected BEFORE (-) or AFTER (+) the period defined          */
/*          for the ESDT (stated in the corresponding ESDT              */
/*          ODL file).          */
/*          -- Defaulted to 0.          */
/*          -- If set, must be an integer number of seconds.          */
/*          A positive value indicates that the value is AFTER          */
/*          the Period of the ESDT. A Negative value is                */
/*          subtracted from the end of the period to find data          */
/*          starting within the period specified for the ESDT.          */
/*          Example          */
/*          END_PERIOD_OFFSET = "-7200" (2 hours)          */
/*****/

    END_PERIOD_OFFSET = 0

/*****/
/*          Input file group ID          */
/*          -- Required for all PCF ENTRY objects with          */
/*          PCF_FILE_TYPE = 1, 3, 6 (ignored otherwise).          */
/*          -- Only used when input is defined as Static in ESDT          */
/*          ODL.          */
/*          -- Must be a string          */
/*          -- 1st character must be one of {C,L,D,O}          */
/*          C -- Coefficient file          */
/*          L -- Lookup file          */
/*          D -- Database file          */
/*          O -- Other Type file          */
/*          -- Rest of string must resolve to a          */
/*          positive integer < 10000          */
/*          Example          */
/*          SCIENCE_GROUP = "C1"          */
/*****/

    SCIENCE_GROUP = ""

/*****/
/*          Type of Input          */
/*          -- Required for all PCF ENTRY objects with          */
/*          PCF_FILE_TYPE = 1,3,6 (ignored otherwise)          */
/*          -- Must be a string with one of the following values:          */
/*          "Required" = Required input/no alternates          */
/*          "Primary" = Primary input/alternates defined          */
/*          Alternate_Input object defined for this          */
/*          PCF Entry.          */
/*          "Optional" = Optional input, PGE can run without it.          */
/*          An Optional_Input object must be defined          */
/*          for this PCF Entry.          */
/*          "Alternate" = Alternate input/there will be an          */
/*          Alternate_Input object defined for this          */
/*          PCF Entry.          */
/*          Example          */
/*          INPUT_TYPE = "Required"          */
/*****/

```

```

INPUT_TYPE = ""

/*****
/*      Align DPR Time with Input                                     */
/*      -- Specifies that the time of the DPR will be shifted      */
/*      to match the real time of input for this Logical Id.      */
/*      -- May only be set for one input per PGE Profile.         */
/*      -- Valid values are "Y" or "N".                           */
/*      -- If not specified, it is set to "N".                     */
/*      Example                                                    */
/*      ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y"                       */
*****/

ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"

/*****
/*      Number of Alternate Inputs needed.                           */
/*      -- Required for all PCF ENTRY objects with                 */
/*      PCF_FILE_TYPE = 1,3,6 that have                             */
/*      INPUT_TYPE = "Primary" (ignored otherwise)                 */
/*      -- Must be either 0 or 1.                                   */
/*      Example                                                    */
/*      NUMBER_NEEDED = 1                                          */
/*      (This means that only 1 of the alternate inputs is        */
/*      required to execute the PGE)                               */
*****/

NUMBER_NEEDED =

/*****
/*      Distinct Value for the input.                                 */
/*      -- Optional entry for PCF ENTRY objects with               */
/*      PCF_FILE_TYPE = 1,3,6. Set to null if not provided.       */
/*      -- A string value, max length 80 characters.              */
/*      -- A value that will allow unique naming of granules     */
/*      input by a PGE.                                           */
/*      -- Must be the name of a metadata parameter defined in   */
/*      a METADATA_DEFINITION object. If a parameter is          */
/*      is specified for which no METADATA_DEFINITION object     */
/*      exists an error will be raised during ODL parsing.       */
/*      -- Supports what are called Multi-Granule ESDTs. These   */
/*      are ESDTs that have multiple granules for the same      */
/*      time period where the only difference between the        */
/*      granules is metadata parameters.                          */
/*      Example                                                    */
/*      DISTINCT_VALUE = "CAMERA_DF"                               */
*****/

DISTINCT_VALUE = ""

/*****
/*      Query Type for the input.                                    */
/*      -- Optional entry for PCF ENTRY objects with               */
/*      PCF_FILE_TYPE = 1,3,6.                                     */
/*      -- Must be one of                                         */
/*      "Temporal" -- Data is retrieved by time.                  */
/*      "Spatial"  -- Data is retrieved by spatial location      */
/*                  of 'key' data type.                           */
/*      "Tile"     -- Data is retrieved by spatial location      */
/*                  of the tile.                                  */
*****/

```

```

/*          "Already Created Tile"                                */
/*          -- Data is retrieved by query of tiles                */
/*          already produced (used for cases when                */
/*          one PGE needs the tile output of one or              */
/*          more other PGEs).                                     */
/*          "Metadata" -- Data is retrieved via temporal query and*/
//          a metadata query                                     */
/*          -- NOTE that if "Already Created Tile" is used, then */
/*          a Metadata Query is expected to query on the TileId  */
/*          parameter in the metadata. "Already Created Tile"    */
/*          will NOT work without a metadata parameter that holds */
/*          the TileId.                                          */
/*          -- The default is "Temporal" (if not specified).     */
/*          Example                                              */
/*          QUERY_TYPE = "Temporal"                              */
/*****/

QUERY_TYPE = ""

/*****/
/*          Spatial Time Delta.                                  */
/*          -- Required for PCF ENTRY objects with                */
/*          PCF_FILE_TYPE = 1,3,6 that have QUERY_TYPE =        */
/*          "Spatial".                                           */
/*          -- An Integer that allows for some time differential */
/*          when querying for input data on spatial constraints. */
/*          It is added to the Start/Stop times of the DPR.     */
/*          -- Time is specified in seconds                       */
/*          Example                                              */
/*          SPATIAL_TIME_DELTA = 100                             */
/*****/

SPATIAL_TIME_DELTA =

/*****/
/*          Spatial Pad                                         */
/*          -- Required for PCF ENTRY objects with                */
/*          PCF_FILE_TYPE = 1,3,6 that have QUERY_TYPE =        */
/*          "Temporal".                                           */
/*          -- A real number (float) value equal to 0.0 or 1000.0. */
/*          Or, a value between those endpoints. The units of    */
/*          measure is kilometers. INTEGERS are not valid!      */
/*          (i.e. 10, 500)                                       */
/*          -- This pad will be applied to the KEY INPUT granule */
/*          Example                                              */
/*          SPATIAL_PAD = 100.0                                   */
/*****/

SPATIAL_PAD =

/*****/
/*          Key Input Data Type.                                  */
/*          -- Optional for PCF ENTRY objects with                */
/*          PCF_FILE_TYPE = 1,3,6 that have QUERY_TYPE =        */
/*          "Temporal" (ignored otherwise).                       */
/*          -- Specifies one of the following:                    */
/*          -- Spatial constraints of this input should be       */
/*          used when acquiring all data with QUERY_TYPE =      */
/*          "Spatial".                                           */
/*          -- The number of granules for the input should      */
/*          determine if a DataScheduled PGE should be          */

```

```

/*          run.          */
/*          -- Must be one of "Y" or "N".          */
/*          -- "YES" should only be set for a single input with a          */
/*          QUERY_TYPE = "Temporal".          */
/*          -- NOTE that the old version of this parameter          */
/*          SPATIAL_KEY_INPUT is still supported and will be          */
/*          treated as having the same meaning.          */
/*          Example          */
/*          KEY_INPUT = "Y"          */
/*****

```

```

KEY_INPUT = " "

```

```

/*****
/* OPTIONAL PARAMETER          */
/* Query Offset for Closest Granule.          */
/*          -- Optional entry for PCF ENTRY objects with          */
/*          PCF_FILE_TYPE = 1,3,6. Set to 0 if not provided.          */
/*          -- Must contain a single P=V string, where          */
/*          P is one of {WEEKS, DAYS, HOURS, MINS, SECS}.          */
/*          Other valid period values are NOT supported for this          */
/*          parameter.          */
/*          -- Used if input is expected to be the "Closest Granule".          */
/*          This means that the data under this PCF_ENTRY will be          */
/*          queried for every CLOSEST_QUERY_OFFSET from the          */
/*          Start Time of the Data Processing Request for the PGE,          */
/*          either forward or backward as indicated by the value          */
/*          of CLOSEST_QUERY_DIRECTION.          */
/*          -- Closest Granule supercedes Most Recent Granule          */
/*          Example          */
/*          CLOSEST_QUERY_OFFSET = "DAYS=1"          */
/*****

```

```

CLOSEST_QUERY_OFFSET =

```

```

/*****
/* Closest Granule Direction.          */
/*          -- Required for PCF ENTRY objects with          */
/*          PCF_FILE_TYPE = 1,3,6 that have specified          */
/*          CLOSEST_QUERY_OFFSET.          */
/*          -- A string that indicates the direction of a search          */
/*          for a desired granule. Must be either:          */
/*          "Forward" or "Backward"          */
/*          -- CLOSEST_QUERY_DIRECTION determines the direction          */
/*          of search (timewise) to query for a suitable granule          */
/*          from the Start Time of the Data Processing Request          */
/*          for the PGE, either forward or backward.          */
/*          -- Closest Granule supercedes Most Recent Granule          */
/*          Examples          */
/*          CLOSEST_QUERY_DIRECTION = "Forward"          */
/*          CLOSEST_QUERY_DIRECTION = "Backward"          */
/*****

```

```

CLOSEST_QUERY_DIRECTION =

```

```

/*****
/* Closest Granule Maximum Number of Retries.          */
/*          -- Required for PCF ENTRY objects with          */
/*          PCF_FILE_TYPE = 1,3,6 that have specified          */
/*          CLOSEST_QUERY_OFFSET.          */
/*          -- An Integer that allows a number of retries on the          */

```

```

/*          inputs where the "Closest Granule" is expected.          */
/*          -- The Query Offset set in the above parameter          */
/*          (CLOSEST_QUERY_OFFSET) is used to repeat the          */
/*          the query for the data for for time periods of          */
/*          Query Offset starting from the Start Time of the          */
/*          Data Processing Request for the PGE either forward or    */
/*          backward as indicated by the value                      */
/*          of CLOSEST_QUERY_DIRECTION.                            */
/*          -- Closest Granule supercedes Most Recent Granule      */
/*          Example                                                */
/*          CLOSEST_QUERY_RETRIES = 20                            */
/*****

CLOSEST_QUERY_RETRIES =

/*****
/* File Types Object                                          */
/*                                                         */
/* THIS OBJECT IS REQUIRED for PCF_FILE_TYPES = 1, 2, 3, 4, 5, 6. */
/*                                                         */
/* The default value for FILETYPE_NAME = "Single File Granule" is */
/* usually all that is needed. This means that the input/output  */
/* has one file per granule. Note that this is separate from the  */
/* MIN/MAX_GRANULES_REQUIRED and MIN/MAX_GRANULE_YIELD parameters */
/* tell how many granules are desired for the PCF entry.          */
/*                                                         */
/* If the Data Type defined under this PCF entry can have multiple */
/* files per data granule then this entry must be updated and there */
/* to be a corresponding entry in the ESDT ODL file for this Data  */
/* Type. There needs to be one of these File Type objects for every */
/* File Type associated with this PCF entry. This object defines   */
/* what file type(s) this PGE wants to use for this PCF entry.    */
/*                                                         */
/* Note that for L0 inputs, there should only be 1 File Type (different */
/* than "Single File Granule") that defines the number of files in a */
/* L0 granule.                                                    */
/*                                                         */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present      */
/*****

OBJECT = FILETYPE

/*****
/*          Class (object counter, used only to distinguish objects) */
/*          -- Must be an integer                                     */
/*          -- Must be unique in this file                           */
/*          Example                                                */
/*          CLASS = 1                                              */
/*****

CLASS = 1

/*****
/*          Name of File Type.                                     */
/*          -- Must be a string, max len 40 characters. Should    */
/*          be meaningful in that the name indicates what sort of */
/*          data is stored within this file type.                  */
/*          -- Defines what File Type is associated with this PCF */
/*          entry. It will determine how many entries are         */
/*          created under this logical ID in the PCF.              */
/*          Example                                                */

```

```

/*          FILETYPE_NAME = "Instrument Band 7"          */
/*****/

    FILETYPE_NAME = "Single File Granule"

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****/

END_OBJECT = FILETYPE

/*****/
/* AUXILIARY LOGICAL ID object          */
/*          */
/* Defines auxiliary logical Ids for a particular input.          */
/* This is used when there may be multiple granules for a particular */
/* Logical Id and the PGE wants each granule under a separate logical */
/* Id. The best example of this is the case where a specific L0 */
/* input could have multiple granules satisfying the given time period. */
/* Since only 1 L0 granule is allowed per logical Id, Auxiliary Logical */
/* Ids can be used to spread the subsequent L0 granules among many */
/* Logical IDs.          */
/*          */
/* When Auxiliary Logical Ids are specified, the first granule that */
/* satisfies the input requirements (time period, metadata checks, */
/* etc.) will be placed under the Logical Id defined under the */
/* PCF_ENTRY. Each subsequent granule will be placed under an */
/* Auxiliary Logical Id. The granules are sorted by time, so the */
/* earliest will go under the PCF_ENTRY Logical Id, with the */
/* Auxiliary Logical Ids filled with later and later granules. */
/*          */
/* There can be more than one AUXILIARY_LOGICAL_ID per PCF_ENTRY, */
/* and if there is one AUXILIARY_LOGICAL_ID object, then there has to */
/* the same number as specified for MAX_GRANULES_REQUIRED.          */
/*          */
/* This object is optional for PCF ENTRY objects with          */
/* PCF_FILE_TYPE = 1, 3 or 6(ignored otherwise). If not needed, this */
/* object should be deleted.          */
/*          */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****/

    OBJECT = AUXILIARY_LOGICAL_ID

/*****/
/*          Class (object counter, used only to distinguish objects) */
/*          -- Must be an integer          */
/*          -- Must be unique in this file for this type of object */
/*          -- Must be greater than 0.          */
/*          Example          */
/*          CLASS = 1          */
/*****/

    CLASS = 1

/*****/
/*          Auxiliary Logical Id          */
/*          -- The Logical Id to place subsequent granules under */
/*          when creating the PCF.          */
/*          -- Must be a positive integer.          */
/*          -- The Ids specified for Toolkit use (10000 to 10999) */

```

```

/*          will not be allowed.          */
/*          Example:                      */
/*          AUX_LOGICAL_ID = 1001        */
/*****

      AUX_LOGICAL_ID =

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

      END_OBJECT = AUXILIARY_LOGICAL_ID

/*****
/* Alternate Input object          */
/*          */
/* Defines parameter names and values for this Data Input to be */
/* designated as an "alternate input." This is defined as an input */
/* that can be substituted for another, already defined input.    */
/*          */
/* Note that the "Primary" or first choice Alternate input is also */
/* designated an Alternate input and thus should have one of these */
/* objects. Order should be set to 1. All subsequent Alternates */
/* should have the same Alternate_Category as the primary and should */
/* have Order > 1.          */
/*          */
/* This object is optional for PCF ENTRY objects with          */
/* PCF_FILE_TYPE = 1, 3 or 6(ignored otherwise). If not needed, this */
/* object should be deleted.          */
/*          */
/* There can only be one of these objects per PCF ENTRY.      */
/*          */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present    */
/*****

      OBJECT = ALTERNATE_INPUT

/*****
/*          Class (object counter, used only to distinguish objects) */
/*          -- Must be an integer          */
/*          -- Must be unique in this file for this type of object */
/*          -- Must be greater than 0.     */
/*          Example          */
/*          CLASS = 1          */
/*****

      CLASS = 1

/*****
/*          Name of Alternate Category          */
/*          -- Must be a string, max len 20 characters          */
/*          -- This is the grouping of Alternates for which this */
/*          entry belongs. The ORDER parameter defines which */
/*          of the alternates is primary, secondary ...          */
/*          -- There should be at least one other entry with the */
/*          category.          */
/*          Example:          */
/*          CATEGORY = "SeaSurfTemp"          */
/*****

```

```

CATEGORY = ""

/*****
/*      Default Order for this Alternate                               */
/*      Indicates the order of preference for alternates             */
/*      within the same category.                                     */
/*      The primary (or first choice alternate) should have         */
/*      ORDER = 1.                                                  */
/*      -- Must be an integer value.                                 */
/*      -- Should be no greater than the maximum number of         */
/*      alternates for the specified CATEGORY.                       */
/*      Example                                                     */
/*      ORDER = 1 (this would be the primary alternate)             */
*****/

ORDER =

/*****
/*      Runtime Parameter Logical Id for this Alternate.           */
/*      Sets up a runtime parameter (defined in the User           */
/*      Defined Runtime Parameters section of the PCF) that will    */
/*      hold the logical ID of the chosen Alternate.               */
/*      -- Must be a positive integer value.                       */
/*      -- Must NOT be a Toolkit specific logical ID              */
/*      (10000 and 10999)                                          */
/*      -- Must have a corresponding Runtime Parameter defined     */
/*      in PCF section 5.                                          */
/*      Example                                                     */
/*      RUNTIME_PARM_ID = 11111                                     */
*****/

RUNTIME_PARM_ID =

/*****
/*      Default Timer value to wait for Alternate to be available  */
/*      -- Must contain a single P=V string, where                 */
/*      P is one of { MONTHS, WEEKS, DAYS, HOURS, MINS, SECS}     */
/*      -- NOTE that this is not needed if WAITFOR (next         */
/*      parameter) is set to "Y".                                   */
/*      Example                                                     */
/*      TIMER = "DAYS=1"                                           */
*****/

TIMER = "PV_Time_Value_goes_here"

/*****
/*      Wait For flag                                              */
/*      Informs PDPS to wait for the alternate input (regardless  */
/*      of the timer value). This means that even if the timer    */
/*      expires, PDPS will wait for it before executing the       */
/*      the PGE.                                                  */
/*      -- A character value of either "Y" (YES) or "N" (NO).     */
/*      -- Must be set the same for all Alternates in the        */
/*      specified CATEGORY. If one Alternate in the CATEGORY      */
/*      is set to "Y" then all WAITFOR flags for Alternates      */
/*      in that list also must have WAITFOR set to "Y".          */
/*      Example                                                     */
/*      WAITFOR = "N"                                             */
*****/

WAITFOR = ""

```

```

/*****
/*      Temporal Flag                                */
/*      Indicates if the alternate should be the previous */
/*      incarnation of the Data Product (Y) rather than the */
/*      most current Product (N).                          */
/*      -- A character value of either "Y" (YES) or "N" (NO). */
/*      Example                                           */
/*      TEMPORAL = "N"                                    */
/*****

      TEMPORAL = ""

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

      END_OBJECT = ALTERNATE_INPUT

/*****
/* Optional Input object                                */
/*                                                    */
/* Defines parameter names and values for this Data Input to be */
/* designated as an "optional input." This means that it is an input */
/* that is desired (if available), but that the PGE can process data */
/* successfully without it.                                    */
/*                                                    */
/* Note that Optional Inputs can work like Alternates, in that there */
/* can be a selection to choose from and an order of preference. */
/* In this case the first choice Optional input would be the "Primary" */
/* (ORDER = 1). If multiple Optional inputs are desired, it is best if */
/* they can be grouped as a list of "Primary" and its "Alternates". */
/*                                                    */
/* This object is optional for PCF ENTRY objects with */
/* PCF_FILE_TYPE = 1, 3 or 6 (ignored otherwise). */
/*                                                    */
/* There can only be one of these objects per PCF ENTRY. */
/* An input can either be Alternate or Optional, not both. */
/* If a PCF entry is not Optional, this object should be deleted. */
/*                                                    */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

      OBJECT = OPTIONAL_INPUT

/*****
/*      Class (object counter, used only to distinguish objects) */
/*      -- Must be an integer */
/*      -- Must be unique in this file for this type of object */
/*      -- Must be greater than 0. */
/*      Example                                           */
/*      CLASS = 1 */
/*****

      CLASS = 1

/*****
/*      Name of Optional Category                                */
/*      -- Must be a string, max len 40 characters */
/*      -- This is the grouping of optional inputs (one or more) */

```

```

/*          for which this entry belongs.  The ORDER paramater          */
/*          defines which of the optionals is primary,                    */
/*          secondary ... for the case where there is more than          */
/*          one optional input.                                           */
/*          Example:                                                       */
/*          CATEGORY = "SeaSurfTemp"                                       */
/*****

CATEGORY = ""

/*****
/*          Default Order for this Optional Input                        */
/*          Indicates the order of preference for optionals              */
/*          within the same category (when there is more than 1).        */
/*          The primary (or first choice optional) should have          */
/*          ORDER = 1.                                                    */
/*          -- Must be an integer value.                                  */
/*          -- Should be no greater than the maximum number of          */
/*          optionals for the specified CATEGORY.                          */
/*          Example                                                       */
/*          ORDER = 1 (this would be the primary optional input or      */
/*          for a single optional input)                                   */
/*****

ORDER =

/*****
/*          Runtime Parameter Logical Id for this Optional Input.        */
/*          Sets up a runtime parameter (defined in the User            */
/*          Defined Runtime Parameters section of the PCF) that will      */
/*          hold the logical ID of the chosen Optional input.           */
/*          -- Must be a positive integer value.                          */
/*          -- Must NOT be a Toolkit specific logical ID                 */
/*          (10000 and 10999)                                             */
/*          -- Must have a corresponding Runtime Parameter defined       */
/*          in PCF section 5.                                             */
/*          Example                                                       */
/*          RUNTIME_PARM_ID = 11111                                       */
/*****

RUNTIME_PARM_ID =

/*****
/*          Default Timer value to wait for Alternate to be available    */
/*          -- Must contain a single P=V string, where                    */
/*          P is one of { MONTHS, WEEKS, DAYS, HOURS, MINS, SECS}        */
/*          -- NOTE that this is not needed if WAITFOR (next            */
/*          parameter) is set to "Y".                                     */
/*          Example                                                       */
/*          TIMER = "DAYS=1"                                              */
/*****

TIMER = "PV_Time_Value_goes_here"

/*****
/*          Temporal Flag                                                */
/*          Indicates if the alternate should be the previous            */
/*          incarnation of the Data Product (Y) rather than the          */
/*          most current Product (N)                                     */
/*          -- A character value of either "Y" (YES) or "N" (NO).        */
/*          Example                                                       */

```

```

/*          TEMPORAL = "N"          */
/*****/

TEMPORAL = ""

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****/

END_OBJECT = OPTIONAL_INPUT

/*****/
/* Metadata checks object */
/*          */
/* Defines parameter names, values and conditions for which this PGE */
/* should execute if true for this input file          */
/* PGE depends.          */
/*          */
/* This object is optional for PCF ENTRY objects with */
/* PCF_FILE_TYPE = 1,3 or 6 (ignored otherwise). Delete if not needed. */
/* Replicate object if multiple METADATA_CHECKS are required. */
/*          */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****/

OBJECT = METADATA_CHECKS

/*****/
/*          Class (object counter, used only to distinguish objects) */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is normally not modified */
/*          -- Must be an integer */
/*          -- Must be unique in this file for this type of object */
/*          -- Must be greater than 0. */
/*          Example */
/*          CLASS = 1 */
/*****/

CLASS = 1

/*****/
/*          Name of metadata parameter on which this PGE depends */
/*          -- Must be a string, max len 40 characters. */
/*          -- Must be present in the ESDT ODL file for this ESDT. */
/*          -- Means that the specified metadata parameter must have */
/*          the specified value for the PGE to execute. */
/*          -- For Product Specific Attributes (PSAs), this is the */
/*          name of the attribute in question. The corresponding */
/*          entry the ESDT_ODL file must specify CONTAINER_NAME = */
/*          "AdditionalAttributes". */
/*          Example: */
/*          The PGE depends on the metadata value for the parameter */
/*          called "tbd_parm_name". */
/*          PARM_NAME = "tbd_parm_name" */
/*****/

PARM_NAME = ""

/*****/
/*          Operator for dependency condition */
/*          -- Must be one of { >, <, >=, <=, ==, != } */

```

```

/*          -- This means that the metadata parameter is:          */
/*          ">" -- actual parameter value must be greater than    */
/*          value specified in VALUE.                               */
/*          "<" -- actual parameter value must be less than      */
/*          value specified in VALUE.                               */
/*          ">=" -- actual parameter value must be greater than  */
/*          or equal to value specified in VALUE.                 */
/*          "<=" -- actual parameter value must be less than or  */
/*          equal to value specified in VALUE.                     */
/*          "==" -- actual parameter value must be equal to      */
/*          value specified in VALUE.                               */
/*          "!=" -- actual parameter value must be NOT equal to  */
/*          value specified in VALUE.                               */

```

```

/*          Example                                               */
/*          OPERATOR = "=="                                       */
/*****

```

OPERATOR = ""

```

/*****
/*          Value for metadata parameter upon which this PGE depends */
/*          -- The value for the metadata parameter that is to be   */
/*          checked against.                                         */
/*          -- Computer data type (string, float or long) of the    */
/*          value must correspond to the computer data type */
/*          given in the ESDT ODL file                               */

```

```

/*          Example                                               */
/*          VALUE = 0                                             */
/*          Requires that TYPE = "INT" for the "tbd_parm_name" object */
/*          in ODL file                                           */
/*          $DPAT_ESDT_SCIENCE_MD/ESDT_<ESDTName#Version>.odl      */
/*          VALUE = "Joe"                                         */
/*          Requires that TYPE = "STR" for the "tbd_parm_name" object */
/*          in ODL file                                           */
/*          $DPAT_ESDT_SCIENCE_MD/ESDT_<ESDTName#Version>.odl      */
/*****

```

VALUE = ""

```

/*****
/*          Database query Value                                  */
/*          -- OPTIONAL parameter. Defaults to "NONE".             */
/*          -- Set to define this Metadata Query as having a      */
/*          a VALUE set by PDPS based on the run of the PGE.     */
/*          This Metadata Query will then be performed on the    */
/*          value retrieved from the PDPS database rather than   */
/*          the value specified in the VALUE parameter.          */
/*          -- Must be one of {"NONE", "PATH NUMBER",            */
/*          "ORBIT NUMBER", "TILE ID", "START DATA DAY",        */
/*          "END DATA DAY", "ORBIT IN DAY", "GRANULE IN ORBIT", */
/*          "YEAR OF DATA", "MONTH OF DATA", "DAY OF DATA"}   */
/*          "NONE" -- no dynamic value, use VALUE                */
/*          "PATH NUMBER" -- get the orbital path number          */
/*          "ORBIT NUMBER" -- get the number of the orbit        */
/*          "TILE ID" -- get the id of the tile                   */
/*          "START DATA DAY" -- get the start data day          */
/*          "END DATA DAY" -- get the end data day              */
/*          "ORBIT IN DAY" -- get the orbit number within day   */
/*          "GRANULE IN ORBIT" -- get the granule within the    */
/*          orbit assuming 6 minute                              */
/*          "YEAR OF DATA" -- the year of the data              */

```

```

/*          "MONTH OF DATA" -- the month of the data          */
/*          "DAY OF DATA" -- the day of the data              */
/*          Example                                           */
/*          DATABASE_QUERY = "PATH NUMBER"                    */
/*****/

DATABASE_QUERY = "NONE"

/*****/
/* Optional Parameter. Defaults to empty string if not specified. */
/*                                                                    */
/*          Name of metadata parameter which provides a key into a */
/*          a multi-containered object. Such an object is the */
/*          MeasuredParameters group in the inventory metadata. */
/*          -- Must be a string, max len 40 characters.          */
/*          -- Must be present in the ESDT ODL file for this ESDT. */
/*          -- Is matched with KEY_PARAMETER_VALUE to determine */
/*          the entry in a multi-containered metadata group. */
/*          -- For Product Specific Attributes (PSAs), this entry */
/*          should NOT be specified.                               */
/*          -- Because of Metadata Query limitations, there can only */
/*          be one KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair */
/*          per PGE ODL File. This is because only a single */
/*          Metadata Query is allowed against the */
/*          MeasuredParameters group.                             */
/*          -- For Metadata Queries within the MeasuredParameters */
/*          group this should be set to the metadata field called */
/*          "ParameterName".                                     */
/*          Example:                                           */
/*          KEY_PARAMETER_NAME = "ParameterName"                */
/*****/

KEY_PARAMETER_NAME = ""

/*****/
/* Optional Parameter. Must be preset if KEY_PARAMETER_NAME exists. */
/* Defaults to the empty string if not specified.                    */
/*                                                                    */
/*          Value of metadata parameter which provides a key into a */
/*          a multi-containered object. Such an object is the */
/*          MeasuredParameters group in the inventory metadata. */
/*          -- Must be a string, max len 80 characters.          */
/*          -- Must be present in the ESDT ODL file for this ESDT. */
/*          -- Is matched with KEY_PARAMETER_NAME to determine */
/*          the entry in a multi-containered metadata group. */
/*          -- For Product Specific Attributes (PSAs), this entry */
/*          should NOT be specified.                               */
/*          -- Because of Metadata Query limitations, there can only */
/*          be one KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair */
/*          per PGE ODL File. This is because only a single */
/*          Metadata Query is allowed against the */
/*          MeasuredParameters group.                             */
/*          -- For Metadata Queries within the MeasuredParameters */
/*          group this should be set to the desired value of the */
/*          metadata field called "ParameterName".               */
/*          Example:                                           */
/*          KEY_PARAMETER_VALUE = "LandCoverage"                */
/*****/

KEY_PARAMETER_VALUE = ""

```

```

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

    END_OBJECT = METADATA_CHECKS

/*****
/* Metadata Query Object */
/*
/*
/* Defines parameter names, values and conditions for which this Input */
/* for the PGE should be selected. Only data that matches the */
/* with the specified metadata parameter with the specified value and */
/* condition will be chosen as input to this PGE. Note that if no */
/* matching data if found the PGE will NOT execute. */
/*
/* This object is optional for PCF ENTRY objects with */
/* PCF_FILE_TYPE = 1,3 or 6 (ignored otherwise). Delete if not needed. */
/* Replicate object if multiple METADATA_QUERYs are required. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

    OBJECT = METADATA_QUERY

/*****
/*
/* Class (object counter, used only to distinguish objects) */
/*
/* -- This line is generated by DpAtCreateOdlTemplate */
/* from the PCF and is normally not modified */
/*
/* -- Must be an integer */
/*
/* -- Must be unique in this file for this type of object */
/*
/* -- Must be greater than 0. */
/*
/* Example */
/*
/* CLASS = 1 */
/*****

    CLASS = 1

/*****
/*
/* Name of metadata parameter on which this PGE depends */
/*
/* -- Must be a string, max len 40 characters */
/*
/* -- Must be present in the ESDT ODL file for this ESDT */
/*
/* Example: */
/*
/* This CERES PGE depends on the Q/A value of */
/*
/* this ESDT "TRWpcal182": execute the CERES PGE only */
/*
/* if ESDT "TRWpcal182" had Q/A parameter */
/*
/* "tbd_parm_name" = 0 */
/*
/* PARM_NAME = "tbd_parm_name" */
/*****

    PARM_NAME = "Parm_name_goes_here"

/*****
/*
/* Operator for dependency condition */
/*
/* -- Must be one of { >, <, >=, <=, ==, != } */
/*
/* Example */
/*
/* OPERATOR = "==" */
/*****

    OPERATOR = "Operator_goes_here"

/*****

```

```

/*      Value for ESDT parameter upon which this PGE depends      */
/*      -- Computer data type (string, float or long) of the      */
/*      value must correspond to the computer data type */
/*      given in the ESDT ODL file                               */
/*      Example                                                 */
/*      VALUE = 0                                               */
/*      Requires that TYPE = "INT" for the "tbd_parm_name" object */
/*      in ODL file                                           */
/*      $DPAT_ESDT_SCIENCE_MD/ESDT_<ESDTName#Version>.odl      */
/*      VALUE = "Joe"                                           */
/*      Requires that TYPE = "STR" for the "tbd_parm_name" object */
/*      in ODL file                                           */
/*      $DPAT_ESDT_SCIENCE_MD/ESDT_<ESDTName#Version>.odl      */
/*****/

```

VALUE = "Value\_goes\_here"

```

/*****/
/*      Database query Value                                     */
/*      -- OPTIONAL parameter. Defaults to "NONE".             */
/*      -- Set to define this Metadata Query as having a       */
/*      a VALUE set by PDPS based on the run of the PGE.       */
/*      This Metadata Query will then be performed on the     */
/*      value retrieved from the PDPS database rather than    */
/*      the value specified in the VALUE parameter.           */
/*      -- Must be one of {"NONE", "PATH NUMBER",             */
/*      "ORBIT NUMBER", "TILE ID", "START DATA DAY",         */
/*      "END DATA DAY", "ORBIT IN DAY", "GRANULE IN ORBIT",  */
/*      "YEAR OF DATA", "MONTH OF DATA", "DAY OF DATA"}    */
/*      "NONE" -- no dynamic value, use VALUE                 */
/*      "PATH NUMBER" -- get the orbital path number           */
/*      "ORBIT NUMBER" -- get the number of the orbit         */
/*      "TILE ID" -- get the id of the tile                   */
/*      "START DATA DAY" -- get the start data day           */
/*      "END DATA DAY" -- get the end data day               */
/*      "ORBIT IN DAY" -- get the orbit number within day     */
/*      "GRANULE IN ORBIT" -- get the granule within the     */
/*      orbit assuming 6 minute                               */
/*      "YEAR OF DATA" -- the year of the data               */
/*      "MONTH OF DATA" -- the month of the data             */
/*      Example                                               */
/*      DATABASE_QUERY = "PATH NUMBER"                       */
/*****/

```

DATABASE\_QUERY = "NONE"

```

/*****/
/*      Optional Parameter. Defaults to empty string if not specified. */
/*      */
/*      Name of metadata parameter which provides a key into a */
/*      a multi-containered object. Such an object is the     */
/*      MeasuredParameters group in the inventory metadata.   */
/*      -- Must be a string, max len 40 characters.          */
/*      -- Must be present in the ESDT ODL file for this ESDT. */
/*      -- Is matched with KEY_PARAMETER_VALUE to determine  */
/*      the entry in a multi-containered metadata group.     */
/*      -- For Product Specific Attributes (PSAs), this entry */
/*      should NOT be specified.                               */
/*      -- For Metadata Checks within the MeasuredParameters */
/*      group this should be set to the metadata field called */
/*      "ParameterName".                                       */

```

```

/*          Example:                                     */
/*          KEY_PARAMETER_NAME = "ParameterName"        */
/*****

KEY_PARAMETER_NAME = ""

/*****
/* Optional Parameter. Must be preset if KEY_PARAMETER_NAME exists. */
/* Defaults to the empty string if not specified.                */
/*                                                                */
/*          Value of metadata parameter which provides a key into a */
/*          a multi-containered object. Such an object is the */
/*          MeasuredParameters group in the inventory metadata.    */
/*          -- Must be a string, max len 80 characters.          */
/*          -- Must be present in the ESDT ODL file for this ESDT. */
/*          -- Is matched with KEY_PARAMETER_NAME to determine */
/*          the entry in a multi-containered metadata group. */
/*          -- For Product Specific Attributes (PSAs), this entry */
/*          should NOT be specified.                               */
/*          -- For Metadata Checks within the MeasuredParameters */
/*          group this should be set to the desired value of the */
/*          metadata field called "ParameterName".              */
/*          Example:                                             */
/*          KEY_PARAMETER_VALUE = "LandCoverage"                */
/*****

KEY_PARAMETER_VALUE = ""

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present      */
/*****

END_OBJECT = METADATA_QUERY

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present      */
/*****

END_OBJECT = PCF_ENTRY

/*****
/* After this point, the comments only address unique parameters that */
/* have not been explained above                                     */
/*                                                                */
/*          Note that the order of PCF entries is not really important. These */
/*          have been ordered the same as their order would be in the PGEs PCF. */
/*****

OBJECT = PCF_ENTRY
CLASS = 2
LOGICAL_ID = 3000
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = ""

/*****
/*          Minimum number of output granules for this logical ID */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is only modified if the PGE may */
/*          successfully produce less granules than specified in */

```

```

/*          the PCF used to generate the template.          */
/*          -- Required for all PCF ENTRY objects with      */
/*          PCF_FILE_TYPE = 2, 4, 7 (ignored otherwise).    */
/*          -- Must be a positive integer.                  */
/*          -- Note that for number of files within a granule */
/*          greater than one, the FILE TYPE object for this entry */
/*          must be changed to specify the various file types and */
/*          maximum number of files.                        */
/*          Example                                          */
/*          MIN_GRANULE_YIELD = 1                          */
/*****

MIN_GRANULE_YIELD = 1

/*****
/*          Maximum number of output granules for this logical ID */
/*          -- This line is generated by DpAtCreateOdlTemplate */
/*          from the PCF and is only modified if the PGE may */
/*          successfully produce more granules than specified in */
/*          the PCF used to generate the template.          */
/*          -- Required for all PCF ENTRY objects with      */
/*          PCF_FILE_TYPE = 2, 4, 7 (ignored otherwise).    */
/*          -- Must be a positive integer.                  */
/*          -- Note that for number of files within a granule */
/*          greater than one, the FILE TYPE object for this entry */
/*          must be changed to specify the various file types and */
/*          maximum number of files.                        */
/*          Example                                          */
/*          MAX_GRANULE_YIELD = 1                          */
/*****

MAX_GRANULE_YIELD = 1

/*****
/*          Associated MCF ID                                */
/*          -- The Logical ID of the MCF associated with this input. */
/*          Informs Data Processing as to the logical id which */
/*          the PGE associates the MCF for this output.      */
/*          -- Required for all PCF ENTRY objects with      */
/*          PCF_FILE_TYPE = 2, 4 (if not output by the Toolkit), */
/*          7. (ignored otherwise).                          */
/*          -- Must be a positive integer.                  */
/*          -- NOTE that any input PCF entries that were created */
/*          by CreateOdlTemplate for MCFs should be deleted. The */
/*          information about which Logical IDs are for MCFs is */
/*          is captured by this parameter for each output that */
/*          the MCF is associated with.                      */
/*          Example                                          */
/*          ASSOCIATED_MCF_ID = 3001                        */
/*****

ASSOCIATED_MCF_ID =

/*****
/*          Output file group ID                            */
/*          -- Required for all PCF ENTRY objects with      */
/*          PCF_FILE_TYPE = 2 (ignored otherwise)          */
/*          -- Must be a string                             */
/*          -- 1st character must be one of {S,Q,H,B}      */
/*          S -- Science file                               */
/*          Q -- Q/A file                                   */

```

```

/*          H -- Production history file          */
/*          B -- Browse file                      */
/*          -- Rest of string must resolve to a   */
/*          positive integer < 1000              */
/*          Example                               */
/*          SCIENCE_GROUP = "S1"                  */
/*          Files associated with this science file would have */
/*          SCIENCE_GROUP = "Q1", SCIENCE_GROUP = "B1", etc.   */
/*****/

```

SCIENCE\_GROUP = " "

```

/*****/
/*          Nominal no. of file instances with *different* logical IDs, */
/*          but which are associated with each other                      */
/*          -- Optional for all PCF ENTRY objects with                    */
/*          PCF_FILE_TYPE = 2 (ignored otherwise).                       */
/*          -- If 0, ignore this parameter -- no other logical IDs      */
/*          are associated with it.                                       */
/*          Example                                                       */
/*          INSTANCE = 0                                                  */
/*          Note: This parameter is specifically designed to accomodate  */
/*          the CERES case where 24 standard product files are generated */
/*          per day, each with a *different* logical ID, but are all     */
/*          essentially an instance of a single file format              */
/*          In this case INSTANCE would take values 1, 2, ..., 24       */
/*****/

```

INSTANCE = 0

```

/*****/
/*          Distinct Value for the output.                                */
/*          -- Optional entry for PCF ENTRY objects with                  */
/*          PCF_FILE_TYPE = 2,4,7. Set to null if not provided.          */
/*          -- A string value, max length 80 characters.                  */
/*          -- A value that will allow unique naming of granules         */
/*          produced by a PGE.                                            */
/*          -- Must be the name of a metadata parameter defined in      */
/*          a METADATA_DEFINITION object. If a parameter is              */
/*          is specified for which no METADATA_DEFINITION object        */
/*          exists an error will be raised during ODL parsing.          */
/*          -- Supports what are called Multi-Granule ESDTs. These      */
/*          are ESDTs that have multiple granules for the same          */
/*          time period where the only difference between the            */
/*          granules is metadata parameters.                             */
/*          Example                                                       */
/*          DISTINCT_VALUE = "CAMERA_DF"                                  */
/*****/

```

DISTINCT\_VALUE = " "

```

/*****/
/*          Minimum expected size (in MB) of this output                 */
/*          (used for QA purposes).                                       */
/*          -- Required for all PCF ENTRY objects with                    */
/*          PCF_FILE_TYPE = 2 (ignored otherwise)                         */
/*          -- Must be a positive integer                                  */
/*          Example                                                       */
/*          MINIMUM_SIZE = 120000                                         */
/*****/

```

```

MINIMUM_SIZE = 0

/*****
/*      Maximum expected size (in MB) of this output          */
/*      (used for QA purposes).                               */
/*      -- Required for all PCF ENTRY objects with           */
/*      PCF_FILE_TYPE = 2 (ignored otherwise)                */
/*      -- Must be a positive integer                         */
/*      -- Must be larger than or equal to MINIMUM_SIZE     */
/*      Example                                              */
/*      MAXIMUM_SIZE = 50000000                             */
*****/

MAXIMUM_SIZE = 1

OBJECT = FILETYPE
CLASS = 1
FILETYPE_NAME = "Single File Granule"
END_OBJECT = FILETYPE

/*****
/* Associated Science Data Object                               */
/*                                                              */
/* THIS OBJECT IS REQUIRED for Outputs where the SCIENCE_GROUP */
/* contains 'B' or 'Q' (meaning it is a BROWSE or QA granule). It is */
/* ignored otherwise.                                         */
/*                                                              */
/* BROWSE and QA output granules are linked to the science granules */
/* for which they are produced. This linkage occurs when the produced */
/* BROWSE or QA granules are inserted to the Data Server. This object */
/* defines the linkage so that the correct link can be made after */
/* the PGE completes and its outputs are inserted to the Data Server. */
/*                                                              */
/* If more than one science granule is associated with the BROWSE or */
/* QA output defined by this PCF_ENTRY, then repeat the Associated */
/* Science Data Objects to specify the various Logical Ids that define */
/* those Associated Science Granules.                         */
/*                                                              */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

OBJECT = ASSOCIATED_SCIENCE_DATA

/*****
/*      Class (object counter, used only to distinguish objects) */
/*      -- Must be an integer                                     */
/*      -- Must be unique in this file                           */
/*      Example                                              */
/*      CLASS = 1                                             */
*****/

CLASS = 1

/*****
/*      Associated Science Granule's Logical Id                */
/*      -- Must a positive integer value.                     */
/*      -- Defines which logical Id this BROWSE/QA granules is */
/*      Associated with. This means that when the associated */
/*      science granule is inserted to the Data Server, a */
/*      will be made with the BROWSE/QA granule defined by */
/*      this PCF_ENTRY.                                       */
*****/

```

```

/*          -- A check will be done to verify that the Logical ID          */
/*          has been defined in the ODL file.                               */
/*          Example                                                         */
/*          ASSOCIATED_SCIENCE_LOGICAL_ID = 12345                          */
/*****/

ASSOCIATED_SCIENCE_LOGICAL_ID =

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present                */
/*****/

END_OBJECT = ASSOCIATED_SCIENCE_DATA

END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY

CLASS = 3
LOGICAL_ID = 200

/*****/
/* This is an example for in Support input.                                */
/*****/
PCF_FILE_TYPE = 3

/*****/
/* Support input and output types (if not associated with generic          */
/* Toolkit files) have their own Data Types and Versions.                  */
/*****/

DATA_TYPE_NAME = ""
DATA_TYPE_VERSION = ""

/*****/
/* This is always 1 for non-product inputs                                  */
/*****/
DATA_TYPE_REQUIREMENT = 1

/*****/
/* Support inputs can be any input type.  Though none are                  */
/* shown, they can have Alternate or Optional input objects as well        */
/* Metadata checks objects.                                                 */
/*****/
INPUT_TYPE = ""
NUMBER_NEEDED = 1

OBJECT = FILETYPE
CLASS = 1
FILETYPE_NAME = "Single File Granule"
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY

CLASS = 4
LOGICAL_ID = 4000
PCF_FILE_TYPE = 5

/*****/
/*          Runtime parameter name                                         */

```

```

/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is normally not modified      */
/*      -- Required for all PCF ENTRY objects with      */
/*      PCF_FILE_TYPE = 5 (ignored otherwise)           */
/*      -- Must be a string, max len 50 characters      */
/*      Example                                         */
/*      PGE_PARAMETER_NAME = "Spacecraft_Class"        */
/*****/
PGE_PARAMETER_NAME = " "

/*****/
/*      Runtime parameter default value                */
/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is normally not modified      */
/*      -- Required for all PCF ENTRY objects with      */
/*      PCF_FILE_TYPE = 5 (ignored otherwise)           */
/*      -- Must be a string, max len 200 characters    */
/*      -- If double quotes must be included in the string */
/*      (i.e. the string must read "This is the string") */
/*      then single quotes must be placed around the string. */
/*      Thus "This is the string" would become 'This is the */
/*      string'. Note that this automatically done by */
/*      the CreateODLTemplate tool.                    */
/*      Example                                         */
/*      PGE_PARAMETER_DEFAULT = "TRMM"                 */
/*****/
PGE_PARAMETER_DEFAULT = " "

/*****/
/*      Runtime parameter Dynamic Value                */
/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is set to "NONE".             */
/*      -- Set to define this runtime parameter as having a */
/*      a value set by PDPS based on the run of the PGE.  */
/*      This runtime parameter will then have the value of */
/*      the specified attribute when the PCF is created.  */
/*      -- Required for all PCF ENTRY objects with      */
/*      PCF_FILE_TYPE = 5 (ignored otherwise)           */
/*      -- Must be one of {"NONE", "PATH NUMBER",      */
/*      "ORBIT NUMBER", "TILE ID", "START DATA DAY",  */
/*      "END DATA DAY"}                                */
/*      "NONE" -- no dynamic value, use Default        */
/*      "PATH NUMBER" -- get the orbital path number    */
/*      "ORBIT NUMBER" -- get the number of the orbit  */
/*      "TILE ID" -- get the id of the tile            */
/*      "START DATA DAY" -- get the start data day    */
/*      "END DATA DAY" -- get the end data day        */
/*      Example                                         */
/*      PGE_PARAMETER_DYNAMIC_VALUE = "PATH NUMBER"    */
/*****/
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"

/*****/
/*      Profile Selector Runtime Parameter Flag        */
/*      -- This line is generated by DpAtCreateOdlTemplate */
/*      from the PCF and is set to "N".               */
/*      -- Must be a string, value of either "Y" (for Yes) and */
/*      "N" (for No).                                    */

```

```

/*      -- If not specified, defaults to "N".                                     */
/*      -- Indicates that this Runtime Parameter (along with                    */
/*      others) uniquely identifies a profile of this PGE                       */
/*      (PGE Name + PGE version) based on the PARAMETER_NAME                   */
/*      and DEFAULT_VALUE pair.                                                */
/*      -- If set to "Y" for any Runtime Parameter, then the                  */
/*      RegisterPGE tool will check to make sure that this                    */
/*      Runtime Parameter/Default Value pairs flagged                          */
/*      assures that this PGE Profile is different from all                    */
/*      the rest.                                                                */
/*      Example                                                                  */
/*      PROFILE_SELECTOR_PGE_PARAMETER = "N"                                    */
/*****

PROFILE_SELECTOR_PGE_PARAMETER = ""

END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY

CLASS = 5
LOGICAL_ID = 200

/*****
/* This is an example for an Interim/Intermediate input.                      */
/*****
PCF_FILE_TYPE = 6

/*****
/* Interim/Intermediate input and output types have their own Data           */
/* Types and Versions.                                                         */
/*****

DATA_TYPE_NAME = ""
DATA_TYPE_VERSION = ""

/*****
/* Last PGE to Use Interim Data Type?                                         */
/* This is a "Y" or "N" parameter that defines if this PGE                    */
/* (the one defined by this ODL file) is the last to use this                */
/* Interim Data type.                                                           */
/* -- Must be a string or "Y" or "N".                                          */
/* Example                                                                      */
/* INTERIM_LAST_PGE_TO_USE = "N"                                               */
/*****

INTERIM_LAST_PGE_TO_USE = "N"

DATA_TYPE_REQUIREMENT = 1

/*****
/* Interim/Intermediate inputs can be any input type. Though none are        */
/* are shown, they can have Alternate or Optional input objects as well      */
/* Metadata checks objects.                                                    */
/*****
INPUT_TYPE = ""
NUMBER_NEEDED = 1

OBJECT = FILETYPE
CLASS = 1
FILETYPE_NAME = "Single File Granule"

```

```

END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED */
*****/

END

```

## C.1.2 ESDT\_ODL.template

```

/*****
/*
/*          TEMPLATE ESDT SCIENCE METADATA ODL FILE          */
/*
/*
/*
/* The SSIT operator's responsibility is to copy this file over and */
/* edit it to add all necessary PDPS metadata values.          */
/*
/* Each ESDT used by a PGE must have a corresponding ESDT SCIENCE */
/* metadata ODL file.                                          */
/*
/* All ESDT ODL files must reside in directory $DPAT_ESDT_SCIENCE_MD . */
/*
/* The operator must add a value to the right of the "=" for each */
/* parameter.                                                  */
/*
/* CHANGE LOG
/* -- Added File Type object.                                05/28/97 */
/* -- Added Processing Level.                                06/04/97 */
/* -- Added Orbit types to period/boundary comments.        06/07/97 */
/* -- Updated Archived_By and Processed_By to be             */
/* required for all types but Static.                        06/24/97 */
/* -- Allowed for 0 value in Interim Short Delete.           10/07/97 */
/* -- Added DURATION parameter.                              11/14/97 */
/* -- Removed OVERLAP as a choice for prediction meth.      11/03/97 */
/* -- Changed METADATA_CHECKS to METADATA_DEFINITION.        12/06/97 */
/* Updated description of FILETYPE object.                   */
/* -- Added optional METADATA_CONTAINER parameter.           12/15/97 */
/* -- Added info on Metadata_Definition for                   02/04/98 */
/* Product Specific Attributes.                               */
/* -- Added The Distinct Parameter definition.                03/24/98 */
/* -- Fixed length for PROVIDER, FILETYPE_NAME.              03/31/98 */
/* -- Made CONTAINER no longer optional for METADATA         05/06/98 */
/* DEFINITION object.                                        */
/* -- Updated definition of USE_OBJECT.                       06/25/98 */
/* -- Added KEY_PARAMATER_NAME and KEY_PARAMETER_VALUE       07/05/98 */
/* for Metadata Definition objects.                           */
/* Updated description for DISTINCT_PARAMETER.                */
/* -- Updated lengths for INSTRUMENT and PLATFORM.           08/13/98 */
/* -- Added "NONROUTINE" for PREDICTION_METHOD               08/24/98 */
/* parameter. This is for ASTER Routine Processing.          */
/* -- Added PRODUCTION_CHAIN object                          07/12/99 */
/* -- Added ONDEMAND_DELETION_INTERVAL parameter             */
/*
*****/

/*****
/*
/*          Data Type name          */
*****/

```

```

/*          -- Must be a string, max len 8 characters          */
/*          -- ESDT name inside ODL file must be identical to */
/*          ESDT name used as part of ODL filename,          */
/*          which in turn was generated from the             */
/*          DATA_TYPE_NAME in the PGE ODL file for the PCF   */
/*          entry.                                           */
/*          -- It should be the same as the Short Name used in the */
/*          ESDT defintion at the Data Server.                */
/*          Example                                           */
/*          DATA_TYPE_NAME = "NMC"                          */
/*****

```

DATA\_TYPE\_NAME = ""

```

/*****
/*          Data Type Version                                */
/*          -- Must be a string, max len 5 characters        */
/*          -- ESDT name inside ODL file must be identical to */
/*          ESDT name used as part of ODL filename,          */
/*          which in turn was generated from the             */
/*          DATA_TYPE_VERSION in the PGE ODL file for the PCF */
/*          entry.                                           */
/*          -- It should be the same as the VersionID used in the */
/*          ESDT defintion at the Data Server.                */
/*          -- Note that this is not important for Interim/   */
/*          Intermediate types.                               */
/*          Example                                           */
/*          DATA_TYPE_VERSION = "3.5.1"                      */
/*****

```

DATA\_TYPE\_VERSION = ""

```

/*****
/*          Science instrument name                          */
/*          -- Must be a string, max 20 len characters        */
/*          Example                                           */
/*          INSTRUMENT = "NMC"                                */
/*****

```

INSTRUMENT = ""

```

/*****
/*          Spacecraft platform name                          */
/*          -- Must be a string, max len 25 characters        */
/*          Example                                           */
/*          PLATFORM = "NOAA9"                                */
/*****

```

PLATFORM = ""

```

/*****
/*          ESDT description                                  */
/*          -- Must be a string, max len 60 characters        */
/*          Example                                           */
/*          DATA_TYPE_DESCRIPTION = "NMC 12-hour forecast"   */
/*****

```

DATA\_TYPE\_DESCRIPTION = ""

```

/*****
/*          ESDT data provider (DAAC name to which files are delivered) */

```

```

/*          -- Must be a string, max len 50 characters          */
/*          Example                                           */
/*          PROVIDER = "National Meteorological Center"       */
/*****

```

PROVIDER = ""

```

/*****
/*          Nominal ESDT file size in MB                      */
/*          -- Must be a floating point number (i.e., include a ".") */
/*          -- Must be greater than 0.000001                  */
/*          Example                                           */
/*          NOMINAL_SIZE = 1.5                                */
/*****

```

NOMINAL\_SIZE =

```

/*****
/*          Processing Level                                  */
/*          -- A string defining the level of processing for this */
/*          ESDT.                                           */
/*          -- Must be a string of no more than 6 characters. */
/*          Example                                           */
/*          PROCESSING_LEVEL = "L0"                          */
/*****

```

PROCESSING\_LEVEL = ""

```

/*****
/*          HDF Data Flag                                     */
/*          Informs DPS that the data within this ESDT will be */
/*          HDF data (if set to Y). Needed for DPS to correctly */
/*          set the PCF entries for metadata access.          */
/*          -- A character value of either "Y" (YES) or "N" (NO). */
/*          -- This will tell the Toolkit whether to get the */
/*          metadata information from the HDF file of the ASCII */
/*          metadata file.                                    */
/*          Example                                           */
/*          HDF_DATA = "N"                                    */
/*****

```

HDF\_DATA = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR files in the INPUT sections */
/* of the PCF (PRODUCT, SUPPORT or INTERMEDIATE)                */
/* (ignored for output files, which are always type "I")       */
/*                                                                */
/*          Dynamic flag -- flags whether an ESDT is dynamic   */
/*          -- Allowed values:                                  */
/*          "S" -- Static file                                  */
/*          "I" -- Dynamic internal file                       */
/*          "E" -- Dynamic external file                      */
/*          "T" -- Interim/Intermediate file                   */
/*          Example                                           */
/*          DYNAMIC_FLAG = "E"                                */
/*****

```

DYNAMIC\_FLAG = ""

```

/*****

```

```

/* THIS PARAMETER IS ONLY REQUIRED FOR Interim/Intermediate files */
/* (DYNAMIC_FLAG = "T") */
/*
/*          */
/* Long Duration of Interim/Intermediate files of the */
/* ESDT before they are be deleted (because no longer needed). */
/* -- Must be a positive number (0 is NOT allowed). */
/* -- Time is specified in MINUTES. */
/* -- This value should be long enough such that there is */
/* no chance that the file will be needed at the end of */
/* this duration. */
/* Example */
/* INTERIM_LONG_DURATION = 7200 (5 days) */
/*****

```

INTERIM\_LONG\_DURATION =

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Interim/Intermediate files */
/* (DYNAMIC_FLAG = "T") */
/*
/*          */
/* Short Duration of Interim/Intermediate files of the */
/* ESDT before they are be deleted (because no longer needed). */
/* -- Must be greater than or equal to 0. It should only */
/* 0 if no other PGE uses this Interim file (i.e. an */
/* Interim file that a PGE uses internally between */
/* Processes). */
/* -- Time is specified in MINUTES. */
/* -- This value is a guess at the soonest (after use and */
/* any QA checks) at when the Interim File can be */
/* deleted. */
/* Example */
/* INTERIM_SHORT_DURATION = 1440 (24 Hours) */
/*****

```

INTERIM\_SHORT\_DURATION =

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic Internal files */
/* (DYNAMIC_FLAG = "I") */
/*
/*          */
/* On Demand Deletion Interval. This is the time between */
/* creation of a granule of this ESDT via an On Demand request */
/* and when this granule is deleted (because it has been */
/* distributed to the requestor). */
/* -- Must contain a single P=V string, where */
/* P is one of { YEARS, MONTHS, THIRDS, WEEKS, DAYS, */
/* HOURS, MINS, SECS, ORBITS} */
/* -- Must be greater than or equal to 1 week ("WEEKS=1"). */
/* An error will be returned if the value specified */
/* is less than 1 week. */
/* -- If not specified then the value defaults to 1 week */
/* ("WEEKS=1"). */
/* Example */
/* ONDEMAND_DELETION_DURATION = "WEEKS=1" */
/*****

```

ONDEMAND\_DELETION\_DURATION = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic External files */
/* (DYNAMIC_FLAG = "E") */

```

```

/*                                     */
/*      Data availability prediction method                                     */
/*      -- Must be one of {"ROUTINE", "NONROUTINE"}                         */
/*      -- "ROUTINE" = data is expected at regular intervals.               */
/*      "NONROUTINE" = data comes in sparatically.                          */
/*      No Period, Boundary or Duration is                                   */
/*      required for NONROUTINE data.                                       */
/*      Example                                                                */
/*      PREDICTION_METHOD = "ROUTINE"                                        */
/*****

```

PREDICTION\_METHOD = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic External files               */
/* (DYNAMIC_FLAG = "E")                                                    */
/*                                                                           */
/*      Supplier name                                                         */
/*      -- Must be a string, max len 30 characters                           */
/*      Example                                                                */
/*      SUPPLIER_NAME = "NOAA"                                               */
/*****

```

SUPPLIER\_NAME = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic External files               */
/* (DYNAMIC_FLAG = "E")                                                    */
/*                                                                           */
/*      Nominal collection period within granule                             */
/*      -- Must contain a single P=V string, where                           */
/*      P is one of { YEARS, MONTHS, THIRDS, WEEKS, DAYS,                   */
/*      HOURS, MINS, SECS, ORBITS}                                           */
/*      -- NOTE that if ORBITS are used PROCESSING_BOUNDARY                 */
/*      must be set to "START_OF_ORBIT".                                       */
/*      -- This value is ignored for PREDICTION_METHOD =                     */
/*      "NONROUTINE"                                                         */
/*      Example                                                                */
/*      PERIOD = "HOURS=12"                                                  */
/*****

```

PERIOD = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic External files               */
/* (DYNAMIC_FLAG = "E")                                                    */
/*                                                                           */
/*      Nominal time boundary on which ESDT arrives                           */
/*      -- Must contain 1 or more P=V strings, where P is one of           */
/*      { START_OF_HOUR, START_OF_6HOUR, START_OF_DAY,                       */
/*      START_OF_WEEK, START_OF_ONE_THIRD_MONTH,                             */
/*      START_OF_MONTH, START_OF_YEAR, START_DATE,                           */
/*      START_OF_ORBIT };                                                    */
/*      also, "+<n>" or "-<n>" may be added to any of these,                 */
/*      where <n> specifies integer seconds.                                   */
/*      For START_DATE an "=" can be added followed by the                 */
/*      start date.                                                           */
/*      -- NOTE that START_OF_ORBIT must be used for Data based             */
/*      on an Orbit Model. A file of named                                   */
/*      ORBIT_<platform>.odl must be present.                                */
/*      -- This value is ignored for PREDICTION_METHOD =                     */

```

```

/*          "NONROUTINE"                                */
/*          Example                                     */
/*          BOUNDARY = "START_OF_DAY+10800"             */
/*****

```

BOUNDARY = ""

```

/*****
/* OPTIONAL PARAMETER                                */
/* ONLY USED FOR Dynamic External files              */
/* (DYNAMIC_FLAG = "E")                              */
/*                                                    */
/*          Duration of the data.                     */
/*          -- Defines the length of time covered by the data.      */
/*          -- Only needed if length of time covered by the data    */
/*             differs from the value specified in PERIOD.          */
/*          -- Must contain a single P=V string, where                */
/*             P is one of { YEARS, MONTHS, THIRDS, WEEKS, DAYS,     */
/*                         HOURS, MINS, SECS, ORBITS}                */
/*          -- NOTE that if ORBITS are used PROCESSING_BOUNDARY     */
/*             must be set to "START_OF_ORBIT".                    */
/*          -- This value is ignored for PREDICTION_METHOD =        */
/*             "NONROUTINE"                                         */
/*          Example                                             */
/*          DURATION = "HOURS=12"                                */
/*****

```

DURATION = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR Dynamic External files      */
/* (DYNAMIC_FLAG = "E")                                          */
/*                                                                */
/*          Avg delay between granule collection and arrival, in secs */
/*          -- Must be a positive integer                          */
/*          Example                                             */
/*          DELAY = 43200                                        */
/*****

```

DELAY =

```

/*****
/*          Spatial characteristics of the Data Type.             */
/*          -- Must be a character, "Y" = Yes, spacial            */
/*             characteristics exist, "N" = No, spacial           */
/*             characteristics do not exist.                       */
/*          Example                                             */
/*          SPATIAL_FLAG = "Y"                                   */
/*****

```

SPATIAL\_FLAG = ""

```

/*****
/* OPTIONAL parameter                                */
/*          Distinct Parameter for Granule naming                */
/*          -- A String, max length 80 characters.              */
/*          -- A value that will allow unqiue naming of granules */
/*             produced by a PGE.                                */
/*          -- Must be the name of a metadata parameter defined in */
/*             a METADATA_DEFINITION objected. If a parameter is  */
/*             is specified for which no METADATA_DEFINITION object */

```

```

/*      exists an error will be raised during ODL parsing.  */
/*      -- Supports what are called Multi-Granule ESDTs.  These  */
/*      are ESDTs that have multiple granules for the same  */
/*      time period where the only difference between the  */
/*      granules is metadata parameters.  */
/*      -- NOTE that this parameter must be unqiue without  */
/*      including KEY_PARAMETER_NAME and KEY_PARAMETER_VALUE.  */
/*      If the parameter requires it, then they must still be  */
/*      specified, but the value specified for  */
/*      DISTINCT_PARAMETER cannot need them to be considered  */
/*      unique.  */
/*      Example  */
/*      DISTINCT_PARAMETER = "CAMERA"  */
/*****/

DISTINCT_PARAMETER = ""

/*****/
/* Use object  */
/* Defines the DAAC(s) where the data is used.  */
/* There should be one of these for every DAAC where the data type is  */
/* used. Delete or replicate this object as necessary.  */
/* This object is really only required for data that is used at a DAAC  */
/* other than where it's produced.  */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present  */
/*****/

OBJECT = USE_OBJECT

/*****/
/*      Class (object counter, used only to distinguish objects)  */
/*      -- Must be an integer  */
/*      -- Must be unique in this file  */
/*      Example  */
/*      CLASS = 1  */
/*****/

CLASS = 1

/*****/
/*      DAAC where the Data Type is used.  */
/*      -- Must be a string, max len 4 characters. Use the  */
/*      DAAC abbreviation (i.e. GSFC)  */
/*      -- There should be one of these for every DAAC where  */
/*      the data type is used.  */
/*      Example  */
/*      USED_BY = "GSFC"  */
/*****/

USED_BY = ""

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present  */
/*****/

END_OBJECT = USE_OBJECT

```

```

/*****
/* THIS PARAMETER IS REQUIRED FOR ALL types of file but STATIC (S) */
/* (DYNAMIC_FLAG = "S") */
/*
/* DAAC where the Data Type is archived. */
/* -- Must be a string, max len 4 characters. Use the */
/* DAAC abbreviation (i.e. GSFC) */
/* Example */
/* ARCHIVED_AT = "GSFC" */
*****/

```

ARCHIVED\_AT = ""

```

/*****
/* THIS PARAMETER IS ONLY REQUIRED FOR ALL types of file but STATIC (S) */
/* (DYNAMIC_FLAG = "S") */
/*
/* DAAC where the Data Type is processed. */
/* -- Must be a string, max len 4 characters. Use the */
/* DAAC abbreviation (i.e. GSFC) */
/* Example */
/* PROCESSED_AT = "GSFC" */
*****/

```

PROCESSED\_AT = ""

```

/*****
/* File Types Object */
/*
/* THIS OBJECT IS REQUIRED FOR all ESDTs that can have multiple files */
/* per data granule. It is NOT needed for ESDTs where each file */
/* represents a single granule (those inputs in the PGE ODL file that */
/* have "Single File Granule" for the File Type). */
/*
/* It is up to the PGE writer to determine if multiple files (whether */
/* different types or multiple files for the same type) are */
/* read/written by the PGE. Files and granules differ because a */
/* a granule is the smallest amount of data recognized by the system, */
/* but one granule may be made up of several files. These files */
/* may be of different types, so that only specific information */
/* (specific files) can be requested as input. */
/*
/* Defines the types of files and their maximum numbers that can be */
/* associated with this ESDT. */
/*
/* There should be one of these for every File Type that can be */
/* associated with this ESDT. */
/*
/* Note that this does NOT need to be added for L0 data. Though */
/* such granules are multi-file, they are handled differently by */
/* PDPS. There does not need to be a FILETYPE object in the ESDT ODL */
/* for L0 data. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

```

OBJECT = FILETYPE

```

/*****
/* Class (object counter, used only to distinguish objects) */
/* -- Must be an integer */
*****/

```

```

/*          -- Must be unique in this file          */
/*          Example          */
/*          CLASS = 1          */
/*****/

CLASS = 1

/*****/
/*          Name of File Type.          */
/*          -- Must be a string, max len 40 characters. Should be meaningful in that the name indicates what sort of data is stored within this file type.          */
/*          -- There should be one of these for every File Type that can be associated with this ESDT.          */
/*          Example          */
/*          FILETYPE_NAME = "Instrument Band 7"          */
/*****/

FILETYPE_NAME = ""

/*****/
/*          Maximum Number of Files under this Type.          */
/*          -- Must be an integer.          */
/*          -- Indicates the maximum number of files for the specified File Type.          */
/*          -- Must be less than 1000.          */
/*          Example          */
/*          MAXIMUM_NUM_FILES = 10          */
/*****/

MAXIMUM_NUM_FILES =

/*****/
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present          */
/*****/

END_OBJECT = FILETYPE

/*****/
/* Metadata Definition Object          */
/*          */
/* Metadata Definition objects are required if there are to be Metadata Checks or Metadata Queries on this ESDT. The object defines the metadata parameters and their types on which checks or queries will or can be performed.          */
/*          */
/* The actual values for the checks and/or queries are defined in the PGE ODL file. All that needs to be defined in this ESDT ODL file is the computer data type of the value. NOTE that there can be a Metadata Definition object in the ESDT file and NO corresponding Metadata Checks or Query object in the PGE ODL file. But if there is a Metadata Checks or Query object in the PGE ODL file, there MUST be a corresponding Metadata Definition in the ESDT ODL file.          */
/*          */
/* This object is optional (only needed if there are Metadata Checks or Metadata Query objects in the corresponding PGE ODL file).          */
/* There may be many of these objects per ESDT file.          */
/*          */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present          */
/*****/

```

OBJECT = METADATA\_DEFINITION

```
/*
*****
/*      Class (object counter, used only to distinguish objects)      */
/*      -- Must be an integer                                          */
/*      -- Must be unique in this file                                */
/*      Example                                                         */
/*      CLASS = 1                                                       */
*****

```

CLASS = 1

```
/*
*****
/*      Parameter name for possible check or query                    */
/*      -- Must be a string, max len 40 characters                    */
/*      -- Must be identical to parm name read in PGE ODL file      */
/*      Example                                                         */
/*      PARM_NAME = "tbd_parm_name"                                    */
*****

```

PARM\_NAME = ""

```
/*
*****
/*      Container name above the parameter to be checked/queried     */
/*      -- If not needed, should be set to "NONE".                    */
/*      -- Must be filled in (correctly) if there is a container     */
/*      object or a group surrounding the parameter specified by    */
/*      PARM_NAME. This is because Inspects on granules can only  */
/*      be performed at the highest level object in the metadata   */
/*      tree. -- Must be a string, max len 100 characters            */
/*      -- For Product Specific Attributes (PSAs) this must be     */
/*      set to "AdditionalAttributes"                                 */
/*      Example                                                         */
/*      For metadata that looks as follows:                            */
/*      GROUP = SOME_GROUP_NAME                                         */
/*      OBJECT = OBJECT_CONTAINER                                       */
/*      CLASS = "1"                                                    */
/*      OBJECT = PARAMETER_WE_ARE_QUERYING_ON                          */
/*      NUM_VAL = 1                                                    */
/*      VALUE = "Value_we_want"                                        */
/*      END_OBJECT = PARAMETER_WE_ARE_QUERYING_ON                      */
/*      END_OBJECT = OBJECT_CONTAINER                                  */
/*      END_GROUP = SOME_GROUP_NAME                                    */
/*      This parameter would be set as follows:                        */
/*      CONTAINER_NAME = "SOME_GROUP_NAME"                             */
*****

```

CONTAINER\_NAME = ""

```
/*
*****
/*      Type of parameter for check or query                          */
/*      -- Must be one of {FLOAT,INT,STR}                              */
/*      Example                                                         */
/*      TYPE = "INT"                                                   */
*****

```

TYPE = ""

```

/*****
/*  Optional Parameter.  Defaults to empty string if not specified.  */
/*
/*      Name of metadata parameter which provides a key into a      */
/*      a multi-containered object.  Such an object is the          */
/*      MeasuredParameters group in the inventory metadata.         */
/*      -- Must be a string, max len 40 characters.                  */
/*      -- Must be present in the ESDT ODL file for this ESDT.      */
/*      -- Is matched with KEY_PARAMETER_VALUE to determine         */
/*      the entry in a mult-containered metadata group.             */
/*      -- For Product Specific Attributes (PSAs), this entry       */
/*      should NOT be specified.                                     */
/*      -- Because an ESDT may be used by more than one PGE, it     */
/*      is possible to have more than one                            */
/*      KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair                  */
/*      (in multiple METADATA_DEFINITION objects)                   */
/*      per ESDT ODL File.  Any PGE ODL file may only have         */
/*      a single KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair.      */
/*      -- For Metadata Checks or Qeuries within the                */
/*      MeasuredParameters group this should be set to the         */
/*      metadata field called "ParameterName".                      */
/*      Example:                                                     */
/*      KEY_PARAMETER_NAME = "ParameterName"                        */
*****/

```

KEY\_PARAMETER\_NAME = ""

```

/*****
/*  Optional Parameter.  Must be preset if KEY_PARAMETER_NAME exists.  */
/*  Defaults to the empty string if not specified.                      */
/*
/*      Value of metadata parameter which provides a key into a      */
/*      a multi-containered object.  Such an object is the          */
/*      MeasuredParameters group in the inventory metadata.         */
/*      -- Must be a string, max len 80 characters.                  */
/*      -- Must be present in the ESDT ODL file for this ESDT.      */
/*      -- Is matched with KEY_PARAMETER_NAME to determine         */
/*      the entry in a mult-containered metadata group.             */
/*      -- For Product Specific Attributes (PSAs), this entry       */
/*      should NOT be specified.                                     */
/*      -- Because an ESDT may be used by more than one PGE, it     */
/*      is possible to have more than one                            */
/*      KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair                  */
/*      (in multiple METADATA_DEFINITION objects)                   */
/*      per ESDT ODL File.  Any PGE ODL file may only have         */
/*      a single KEY_PARAMETER_NAME/KEY_PARAMETER_VALUE pair.      */
/*      -- For Metadata Checks or Queries within the                */
/*      MeasuredParameters group this should be set to the         */
/*      desired value of the metadata field called                  */
/*      "ParameterName".                                           */
/*      Example:                                                     */
/*      KEY_PARAMETER_VALUE = "LandCoverage"                        */
*****/

```

KEY\_PARAMETER\_VALUE = ""

```

/*****
/*  THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present          */
*****/

```

END\_OBJECT = METADATA\_DEFINITION

```

/*****
/* Metadata Definition object may be repeated as needed */
/*****

/*****
/* Production Chain Object */
/*
/* THIS OBJECT is only needed for those ESDTs that will be produced */
/* by On Demand Production Requests (Production Requests that are */
/* generated as a result of a request for an On Demand Product). */
/*
/* The Production Chain object surrounds a list (in order) of the */
/* PGEs needed to produce a granule of this ESDT. There may be one */
/* PGE in the list (if that PGE takes in DYNAMIC External data and */
/* produces this ESDT), or a chain of PGEs (if PGE A produces an */
/* ESDT that is input to PGE B which produces THIS ESDT). */
/*
/* The information contained within this object will only be used if */
/* an On Demand Request is for an ESDT which must have another */
/* ESDT produced for the PGE that is to create the Product. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

```

OBJECT = PRODUCTION\_CHAIN

```

/*****
/*      Class (object counter, used only to distinguish objects) */
/*      -- Must be an integer */
/*      -- Must be unique in this file */
/*      Example */
/*      CLASS = 1 */
/*****

```

CLASS = 1

```

/*****
/* PGE In Chain Object */
/*
/* THIS OBJECT defines a PGE that is part of the Production Chain */
/* used to produce this ESDT. */
/*
/* The Production Chain object surrounds a list (in order) of the */
/* PGEs needed to produce a granule of this ESDT. There may be one */
/* PGE in the list (if that PGE takes in DYNAMIC External data and */
/* produces this ESDT), or a chain of PGEs (if PGE A produces an */
/* ESDT that is input to PGE B which produces THIS ESDT). */
/*
/* The PGE_IN_CHAIN objects within the PRODUCTION_CHAIN object define */
/* the PGEs (in order) that need to be run to produce this ESDT. */
/* Only the PGE Name and Version are needed to identify the PGE, the */
/* Profile Id will be the one with the DEFEAUL_PROFILE flag set. */
/*
/* The information contained within this object will only be used if */
/* an On Demand Request is for an ESDT which must have another */
/* ESDT produced for the PGE that is to create the Product. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

```

```

OBJECT = PGE_IN_CHAIN

/*****
/*      Class (object counter)                               */
/*      -- Must be an integer                               */
/*      -- Must be unique in this file                      */
/*      -- Is used in this case to determine the order of the */
/*      PGEs.  CLASS = 1 is the first PGE in the chain.    */
/*      Example                                             */
/*      CLASS = 1                                           */
*****/

CLASS = 1

/*****
/*      PGE name                                           */
/*      -- Must be a string, max len 10 characters         */
/*      -- This is the name of the PGE that makes up one entry */
/*      in the chain of PGEs.                             */
/*      Example                                             */
/*      PGE_NAME = "ssit"                                   */
*****/

PGE_NAME = ""

/*****
/*      PGE version                                         */
/*      -- Must be a string, max len 5 characters         */
/*      -- This is the version of the PGE that makes up one */
/*      entry in the chain of PGEs.                       */
/*      Example                                             */
/*      PGE_VERSION = "1.0"                                */
*****/

PGE_VERSION = ""

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

END_OBJECT = PGE_IN_CHAIN

/*****
/* Repeat PGE_IN_CHAIN objects as needed to make up the Production */
/* chain.                                                         */
*****/

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

END_OBJECT = PRODUCTION_CHAIN

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED                    */
*****/

END

```

### C.1.3 ORBIT\_ODL.template

```

/*****
/*
/*          TEMPLATE ORBIT MODEL METADATA ODL FILE          */
/*
/*          */
/*          */
/* The SSIT operator's responsibility is to copy this file over and */
/* edit it to add all necessary PDPS metadata values.          */
/*          */
/* All ORBIT MODEL ODL files must reside in directory          */
/* $DPAT_RULE_SCIENCE_MD (set in the configuration files).    */
/*          */
/* The operator must add a value to the right of the "=" for each */
/* parameter.          */
/*          */
/* This file is only needed if the PGE has a period/boundary relating */
/* to orbit.          */
/*          */
/* There can be one or more ORBIT_MODEL objects defined in */
/* this file so that multiple orbits can be defined for the same */
/* platform.          */
/*          */
/* CHANGE LOG          */
/* -- Added Orbit_Path_Number.          11/18/97 */
/* -- Changed acceptable Date Format.    01/05/98 */
/* -- Added another acceptable date format. 06/24/98 */
/* -- Updated length of PLATFORM.       08/13/98 */
/* -- Fixed where this file is located in above 10/01/98 */
/* comments.          */
*****/

/*****
/*          Spacecraft platform name for the Orbit Model.          */
/*          -- Must be a string, max len 25 characters          */
/*          Example          */
/*          PLATFORM = "TRMM"          */
*****/

PLATFORM = ""

/*****
/* Orbit Model object          */
/*          */
/* Defines the Orbit Model for a single orbit          */
/*          */
/* Replicate for the defining of multiple orbits for the same platform. */
/*          */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present          */
*****/

OBJECT = ORBIT_MODEL

/*****
/*          Class (object counter, used only to distinguish objects) */
/*          -- Must be an integer          */
/*          -- Must be unique in this file for this type of object */
/*          -- Must be greater than 0.          */
/*          Example          */
/*          CLASS = 1          */
*****/
```

```

CLASS = 1

/*****
/*      Number of the Orbit                               */
/*      -- Must be an integer                             */
/*      -- Must be >= 0                                   */
/*      Example                                           */
/*      ORBIT_NUMBER = 12                                 */
*****/

ORBIT_NUMBER =

/*****
/*      Path Number of the Orbit                           */
/*      -- Must be an integer                             */
/*      -- Must be >= 0 and <= 233                       */
/*      Example                                           */
/*      ORBIT_PATH_NUMBER = 3                             */
*****/

ORBIT_PATH_NUMBER =

/*****
/*      The period of the orbit (a duration).             */
/*      -- Must contain a single P=V string, where       */
/*      P is one of { MONTHS, WEEKS, DAYS, HOURS, MINS, SECS } */
/*      Example                                           */
/*      ORBIT_PERIOD = "HOURS=100"                       */
*****/

ORBIT_PERIOD = " "

/*****
/*      The starting date/time of the orbit.             */
/*      -- Must contain the date and time of the orbit. */
/*      -- The format for the date/time string can be one of the */
/*      following:                                         */
/*      "MMM DD YYYY HH:MM:SS", where                     */
/*      YYYY=4 digit year, MMM=3 character abbreviation for */
/*      Month, DD=2 digit Day, HH=Hours, MM=Minutes,     */
/*      SS=Seconds. The time is accepted as UTC.         */
/*      */
/*      "MM/DD/YYYY HH:MM:SS"                             */
/*      YYYY=4 digit year, MM=2 digit Month,             */
/*      DD=2 digit Day, HH=Hours, MM=Minutes,           */
/*      SS=Seconds. The time is accepted as UTC.         */
/*      -- NOTE that the format for the date of MM/DD/YY will */
/*      no longer be accepted because it did not handle years */
/*      after 1999 correctly.                             */
/*      Example                                           */
/*      ORBIT_START = "Oct 31 1996 22:01:55"             */
*****/

ORBIT_START = " "

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

END_OBJECT = ORBIT_MODEL

```

```

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED */
*****/

```

END

### C.1.4 TILE\_ODL.template

```

/*****
/*
/*          TEMPLATE TILE DEFINITION METADATA ODL FILE          */
/*
/*          */
/* The SSIT operator's responsibility is to copy this file over and */
/* edit it to add all necessary PDPS metadata values.          */
/*          */
/* Each Tile Scheme used by a PGE must have a corresponding TILE */
/* DEFINITION metadata ODL file.                                */
/*          */
/* All TILE DEFINITION ODL files must reside in directory      */
/* $DPAT_RULE_SCIENCE_MD. Each must be named TILE_<tile scheme>.odl */
/*          */
/* For a PGE to use a tile scheme, it must have SCHEDULE_TYPE = */
/* "Tile". TILE_SCHEME_TYPE must equal the tiling scheme to be used. */
/*          */
/* The operator must add a value to the right of the "=" for each */
/* parameter.                                                  */
/*          */
/*          */
/* CHANGE LOG
/* -- Removed the concept of CLUSTERS.                        01/18/98 */
/* -- Added COORDINATE object.                                */
/* -- Updated various descriptions to make them better.      02/04/98 */
/*          */
/*          */
*****/

/*****
/*          Tile Scheme          */
/*          -- Must be a string, max len 20 characters      */
/*          -- There can be NO spaces in the string.        */
/*          -- Tile Scheme must be identical to            */
/*          Tile Scheme used as part of ODL filename,      */
/*          which in turn was generated from the          */
/*          TILE_SCHEME_NAME in the PGE ODL file.          */
/*          Example
/*          TILE_SCHEME_NAME = "Earth_Squared"
*****/

TILE_SCHEME_NAME = " "

/*****
/* Tile object          */
/*          */
/* Defines a tile for the scheme defined by TILE_SCHEME_NAME. */
/* Each tile must be defined seperately, with an ID, and associated */
/* coordinates.          */
/*          */

```

```

/* There should be a Tile object for every tile in the Tiling Scheme. */
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

OBJECT = TILE

/*****
/*      Class (object counter, used only to distinguish objects) */
/*      -- Must be an integer */
/*      -- Must be unique in this file for this type of object */
/*      -- Must be greater than 0. */
/*      Example */
/*      CLASS = 1 */
/*****

CLASS = 1

/*****
/*      ID of Tile */
/*      -- Must be an integer */
/*      -- Must greater than 0 but less than max integer. */
/*      -- Tiles should be listed sequentially (though no */
/*      checking for this is done by software). */
/*      -- This must be unique throughout the system. This */
/*      means that if this Tile Id is defined in other Tile */
/*      Schemes, it must have the same coordinates and */
/*      description. */
/*      Example */
/*      TILE_ID = 12 */
/*****

TILE_ID =

/*****
/*      Description of a Tile */
/*      -- A String of characters, max 255. */
/*      -- Describes what the Tile is for, perhaps its */
/*      geographic location or area that it covers. */
/*      Example */
/*      TILE_DESCRIPTION = "Upper North America" */
/*****

TILE_DESCRIPTION = ""

/*****
/* Tile Coordinate object */
/*
/* Defines a coordinate (Latitude and Longitude) for a tile. */
/*
/* Each tile must have at least 4 TILE_COORDINATE objects defined. More */
/* (than 4) such objects are permitted to better define the tile. */
/*
/* Coordinate objects must follow a clockwise sequence such that if */
/* lines were drawn between the points in the order they are given the */
/* desired shape would be drawn.
/*
/*
/*
/*

```

```

/*
/* For example:
/*
/*      Coordinate 1                      Coordinate 2
/*      o-----o----->o
/*      ^
/*      |
/*      |
/*      o<-----o-----o
/*      Coordinate 4                      Coordinate 3
/*
/*      Or:
/*
/*      Coordinate 2                      Coordinate 3
/*      o-----o----->o
/*      ^
/*      |
/*      |
/*      o<-----o-----o
/*      Coordinate 1                      Coordinate 4
/*
/*
/*
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present
/*
/*****

```

OBJECT = TILE\_COORDINATE

```

/*****
/*      Class (object counter, used only to distinguish objects)
/*      -- Must be an integer
/*      -- Must be unique in this file for this type of object
/*      -- Must be greater than 0.
/*      Example
/*      CLASS = 1
/*****

```

CLASS = 1

```

/*****
/*      Latitude Coordinate
/*      -- Must be one per Coordinate object.
/*      -- Must be an float
/*      Example
/*      LATITUDE = 12.15
/*****

```

LATITUDE =

```

/*****
/*      Longitude Coordinate
/*      -- Must be one per Coordinate object.
/*      -- Must be an float
/*      Example
/*      LONGITUDE = -43.22
/*****

```

LONGITUDE =

```

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present
/*****

```

```

END_OBJECT = TILE_COORDINATE

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
*****/

END_OBJECT = TILE

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED */
*****/

END

```

### C.1.5 PATHMAP\_ODL.template

```

/*****
/*
/*          TEMPLATE PATHMAP DEFINITION METADATA ODL FILE
/*
/*
/* The SSIT operator's responsibility is to copy this file over and
/* edit it to add all necessary PATH MAP metadata values.
/*
/*
/* A PATHMAP defines the mapping between Absolute Path Number
/* a sequential numbering from 1-233 and Mapped Path Number which
/* is the interpreted 1-233 number.
/*
/*
/* If a PGE defines a PATHMAP in the PGE ODL then there must be a
/* matching PATHMAP DEFINITION metadata ODL file and the PGE must have
/* a SCHEDULE_TYPE = "Orbit".
/*
/*
/* All PATHMAP DEFINITION ODL files must reside in directory
/* $DPAT_RULE_SCIENCE_MD. Each must be named
/* PATHMAP_<Pathmap_Name>.odl. Note there can be NO spaces in the
/* Pathmap_Name because it is used as a filename.
/*
/*
/* For a PGE to use a PATHMAP, the PATHMAP_NAME parameter in the PGE
/* ODL file must equal the Pathmap_Name to be used.
/*
/*
/* The operator must add a value to the right of the "=" for each
/* parameter.
/*
/*
/* CHANGE LOG
/*
/*
*****/

/*****
/*          Spacecraft platform name for the Orbit Model.
/*          -- Must be a string, max len 20 characters
/*          Example
/*          PLATFORM = "TRMM"
*****/

PLATFORM = " "

/*****

```

```

/*          Pathmap Name                                     */
/*          -- Must be a string, max len 25 characters      */
/*          -- There can be NO spaces in the string.       */
/*          -- Pathmap Name must be identical to          */
/*          Pathmap Name used as part of ODL filename,    */
/*          which in turn was generated from the          */
/*          PATHMAP_NAME in the PGE ODL file.             */
/*          Example                                         */
/*          PATHMAP_NAME = "Some_Pathmap"                 */
/*****

```

```
PATHMAP_NAME = " "
```

```

/*****
/* Pathmap Entry Object                                     */
/*                                                         */
/* Defines a mapping between Absolute Path Number        */
/* a sequential numbering from 1-233 and Mapped Path Number which */
/* is the interpreted 1-233 number.                       */
/*                                                         */
/* There should be a Pathmap Entry object for each 1-233 Path Number. */
/* An error will be returned if one of the path numbers is missed. */
/*                                                         */
/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****

```

```
OBJECT = PATHMAP_ENTRY
```

```

/*****
/*          Class (object counter, used only to distinguish objects) */
/*          -- Must be an integer                                     */
/*          -- Must be unique in this file for this type of object */
/*          -- Must be greater than 0.                             */
/*          Example                                                 */
/*          CLASS = 1                                               */
/*****

```

```
CLASS = 1
```

```

/*****
/*          Absolute Path Number                                     */
/*          -- Must be an integer                                     */
/*          -- Must be between 1-233.                               */
/*          Example                                               */
/*          ABSOLUTE_PATH = 20                                     */
/*****

```

```
ABSOLUTE_PATH =
```

```

/*****
/*          Mapped Path Number                                     */
/*          -- Must be an integer                                     */
/*          -- Must be between 1-233.                               */
/*          Example                                               */
/*          MAPPED_PATH = 27                                       */
/*****

```

```
MAPPED_PATH = " "
```

```

/*****

```

```

/* THE FOLLOWING LINE MUST NOT BE MODIFIED if it is present */
/*****
END_OBJECT = PATHMAP_ENTRY

/*****
/* THE FOLLOWING LINE MUST NOT BE MODIFIED */
/*****
END

```

## C.2 Typical ASTER PGE & ESDT ODL Files

Listings are provided for the following ASTER ODL files:

C.2.1 ASTER PGE ODL file for PGE\_NAME BTS

C.2.2 ASTER ESDT ODL file for DATA\_TYPE\_NAME AST\_LIB

C.2.3 ASTER ESDT ODL file for DATA\_TYPE\_NAME AST\_ANC

C.2.4 ASTER ESDT ODL file for DATA\_TYPE\_NAME AST\_04

C.2.5 ASTER ESDT ODL file for DATA\_TYPE\_NAME AST\_09T

AST\_LIB, AST\_ANC and AST\_04 are referenced within the PGE .

A typical ASTER PGE will differ from the example here by the PGE\_NAME, the specific input/output files referenced, and runtime parameters. However, the overall structure of a given ASTER PGE ODL file would be similar to the one used here. (N.B. The ODL files shown here are associated with the ASTER version v2.2.34 software)

### C.2.1 ASTER PGE BTS ODL

```

PGE_NAME = "BTS"
PGE_VERSION = "2.2h"
PROFILE_ID = 1
PROFILE_DESCRIPTION = "ASTER Brightness Temp with QA"
PLATFORM = "AM1"
INSTRUMENT = "ASTER"
MINIMUM_OUTPUTS = 1
SCHEDULE_TYPE = "Data"
PROCESSING_PERIOD = "SECS=1"
PROCESSING_BOUNDARY = "START_OF_SEC"
PGE_SSW_VERSION = "2.2h"

OBJECT = PCF_ENTRY
  CLASS = 11
  LOGICAL_ID = 15004
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "AST_L1B"
  DATA_TYPE_VERSION = "001"
  DATA_TYPE_REQUIREMENT = 1
  INPUT_TYPE = "Required"
  KEY_INPUT = "Y"
  NUMBER_NEEDED = 1
/**** Entry needed for all I/O (except for Temporary) ****/
/**** Only modify if multiple files and/or file types for this PCF entry ****/
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1

```

```
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
/*Bright-Temp-LUT-v3.hdf*/
```

```
OBJECT = PCF_ENTRY
CLASS = 12
LOGICAL_ID = 15330
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "AST_ANC"
DATA_TYPE_VERSION = "001"
DATA_TYPE_REQUIREMENT = 1
SCIENCE_GROUP = L1
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
```

```
/***** Entry needed for all I/O (except for Temporary) *****/
```

```
/***** Only modify if multiple files and/or file types for this PCF entry *****/
```

```
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
```

```
/***** "atmcorr-v3-dec.hdf" *****/
```

```
CLASS = 29
LOGICAL_ID = 15152
PCF_FILE_TYPE = 3
DATA_TYPE_NAME = "AST_ANC"
DATA_TYPE_VERSION = "001"
DATA_TYPE_REQUIREMENT = 1
SCIENCE_GROUP = "L2"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
```

```
/***** Entry needed for all I/O (except for Temporary) *****/
```

```
/***** Only modify if multiple files and/or file types for this PCF entry *****/
```

```
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
```

```
/***** "FBA_Filter_File_1.cal" *****/
```

```
CLASS = 30
LOGICAL_ID = 15151
PCF_FILE_TYPE = 3
DATA_TYPE_NAME = "AST_ANC"
DATA_TYPE_VERSION = "001"
DATA_TYPE_REQUIREMENT = 1
SCIENCE_GROUP = "O30"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
```

```
/***** Entry needed for all I/O (except for Temporary) *****/
```

```
/***** Only modify if multiple files and/or file types for this PCF entry *****/
```

```
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```

/* QA2-binning-intervals-v1.cal */
OBJECT = PCF_ENTRY
  CLASS = 13
  LOGICAL_ID = 15913
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "AST_ANC"
  DATA_TYPE_VERSION = "001"
  DATA_TYPE_REQUIREMENT = 1
  SCIENCE_GROUP = 098
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
/**** Entry needed for all I/O (except for Temporary) ****/
/**** Only modify if multiple files and/or file types for this PCF entry ****/
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/**/
/* QA_thresholds.dat */
OBJECT = PCF_ENTRY
  CLASS = 14
  LOGICAL_ID = 15120
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "AST_ANC"
  DATA_TYPE_VERSION = "001"
  DATA_TYPE_REQUIREMENT = 1
  SCIENCE_GROUP = 097
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
/**** Entry needed for all I/O (except for Temporary) ****/
/**** Only modify if multiple files and/or file types for this PCF entry ****/
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*Output Product*/
OBJECT = PCF_ENTRY
  CLASS = 15
  LOGICAL_ID = 15010
  PCF_FILE_TYPE = 2
  DATA_TYPE_NAME = "AST_04"
  DATA_TYPE_VERSION = "001"
  YIELD = 1
  ASSOCIATED_MCF_ID = 15114
  SCIENCE_GROUP = "S1"
  INSTANCE = 0
  MINIMUM_SIZE = 5
  MAXIMUM_SIZE = 10
/**** Entry needed for all I/O (except for Temporary) ****/
/**** Only modify if multiple files and/or file types for this PCF entry ****/
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY

```

```

CLASS = 126
LOGICAL_ID = 15015
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "ASTALGRN"
DATA_TYPE_VERSION = "001"
YIELD = 1
SCIENCE_GROUP = "S3"
ASSOCIATED_MCF_ID = 15119
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
/**** Entry needed for all I/O (except for Temporary) ****/
/**** Only modify if multiple files and/or file types for this PCF entry ****/
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 16
LOGICAL_ID = 15602
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "PGE Major Version"
PGE_PARAMETER_DEFAULT = "2"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 17
LOGICAL_ID = 15603
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "PGE Minor Version"
PGE_PARAMETER_DEFAULT = "2"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 18
LOGICAL_ID = 16200
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "QA PGE Major Version"
PGE_PARAMETER_DEFAULT = "2"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 19
LOGICAL_ID = 16201
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "QA PGE Minor Version"
PGE_PARAMETER_DEFAULT = "2"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 20
LOGICAL_ID = 15604
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Brightness Temperature LUT"
PGE_PARAMETER_DEFAULT = "3"

```

```

    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 139
    LOGICAL_ID = 15167
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "BrTtmp Lookup Table Version"
    PGE_PARAMETER_DEFAULT = "3"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 22
    LOGICAL_ID = 15165
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Atmos Corr. LUT Version"
    PGE_PARAMETER_DEFAULT = "3"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 24
    LOGICAL_ID = 15166
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "FBA Filters File Version"
    PGE_PARAMETER_DEFAULT = "3"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 21
    LOGICAL_ID = 15914
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "QA2 Binning Interval Version"
    PGE_PARAMETER_DEFAULT = "1"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 135
    LOGICAL_ID = 15320
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Alert File indirection"
    PGE_PARAMETER_DEFAULT = "15015:1"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 135
    LOGICAL_ID = 15321
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "PGE Name"
    PGE_PARAMETER_DEFAULT = "Brightness Temperature at the Sensor"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

END

```

### C.2.2 ASTER ESDT AST\_LIB ODL

```
DATA_TYPE_NAME = "AST_L1B"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "ASTER"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "ASTER Level 1B Data Set Registered Radiance at the  
                          Sensor"  
PROVIDER = "EROS Data Center"  
NOMINAL_SIZE = 120.0  
PROCESSING_LEVEL = "L1"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "E"  
PREDICTION_METHOD = "NONROUTINE"  
SUPPLIER_NAME = "EDC"  
PERIOD = "SECS=1"  
DURATION = "SECS=1"  
BOUNDARY = "START_OF_SEC"  
DELAY = 1  
SPATIAL_FLAG = "Y"  
ARCHIVED_AT = "EDC"  
PROCESSED_AT = "EDC"  
  
END
```

### C.2.3 ASTER ESDT AST\_ANC ODL

```
DATA_TYPE_NAME = "AST_ANC"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "ASTER"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "HDF Ancillary data for ASTER"  
PROVIDER = "Goddard Space Flight Center"  
PROCESSING_LEVEL = "L1"  
HDF_DATA = "Y"  
NOMINAL_SIZE = 1.0  
DYNAMIC_FLAG = "S"  
  
END
```

### C.2.4 ASTER ESDT AST\_04 ODL

```
DATA_TYPE_NAME = "AST_04"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "ASTER"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "ASTER Level 2 Brightness Temperature at the Sensor"  
PROVIDER = "Goddard Space Flight Center"  
PROCESSING_LEVEL = "L1"  
HDF_DATA = "Y"  
NOMINAL_SIZE = 4.744895  
DYNAMIC_FLAG = "I"  
ARCHIVED_AT = "EDC"  
PROCESSED_AT = "EDC"  
  
END
```

## **C.2.5 ASTER ESDT AST\_09T ODL**

```
DATA_TYPE_NAME = "AST_09T"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "ASTER"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "ASTER Level 2 Surface Radiance Product (TIR)"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 9.439935  
PROCESSING_LEVEL = "L1"  
HDF_DATA = "Y"  
DYNAMIC_FLAG = "I"  
PREDICTION_METHOD = "ROUTINE"  
SUPPLIER_NAME = "EDC"  
PERIOD = "SECS=1"  
BOUNDARY = "START_OF_SEC"  
DELAY = 1  
SPATIAL_FLAG = "N"  
ARCHIVED_AT = "EDC"  
PROCESSED_AT = "EDC"
```

END

## **C.3 Typical MISR PGE & ESDT ODL Files**

Listings are provided for the following MISR ODL files:

C.3.1 MISR PGE ODL file for PGE\_NAME MPGE1 (M1AN)

C.3.2 MISR ESDT MISANCGM ODL

C.3.3 MISR ESDT MIRCCT ODL

### **C.3.4 MISR ESDT MISLOAN ODL**

C.3.5 MISR ESDT ActSched ODL

C.3.6 MISR ESDT MIANCSSC ODL

C.3.7 MISR ESDT MIANCAGP ODL

C.3.8 MISR ESDT MIANPPAN ODL

C.3.9 MISR ESDT MISL0SY1 ODL

C.3.10 MISR ESDT MISL0SY2 ODL

C.3.11 MISR ESDT MISL0SY3 ODL

C.3.12 MISR ESDT MIRFOIAN ODL

C.3.13 MISR ESDT MIB2GEOP ODL

C.3.14 MISR ESDT MIANCARP ODL (Version# 001)

C.3.15 MISR ESDT MIANCARP ODL (Version# 002)

C.3.16 MISR ESDT MICNFG ODL

C.3.17 MISR ESDT AM1EPHN0 ODL

C.3.18 MISR ESDT AM1ATTNF ODL

C.3.19 MISR ESDT MIANRCCH ODL

C.3.20 MISR ESDT MIL1A ODL

C.3.21 MISR ESDT MISBR ODL

C.3.22 MISR ESDT MISQA ODL

- C.3.23 MISR ESDT MI1B2T ODL
- C.3.24 MISR ESDT MI1B2E ODL
- C.3.25 MISR ESDT MIRCCM ODL
- C.3.26 MISR ESDT MI1B1 ODL
- C.3.27 MISR ESDT MIB1LM ODL

A typical MISR PGE will differ from the example here by the PGE\_NAME, the specific input/output files referenced, and runtime parameters. However, the overall structure of a given MISR PGE ODL file would be similar to the one used here. (N.B. The ODL files shown here are associated with the MISR version v2.1.3 Patch 2 software)

### C.3.1 MISR PGE MPGE1AN ODL (profile #1)

```

PGE_NAME = "M1AN"
PGE_VERSION = "21302"
PROFILE_ID = 1
PROFILE_DESCRIPTION = "MISR PGE 1 AN - Version V21302, SSI&T 17 March 2001"
PLATFORM = "AM1"
INSTRUMENT = "MISR"

/* MISR PGE 1 produces at a minimum 11 output files including QA          */
MINIMUM_OUTPUTS = 11

SCHEDULE_TYPE = "Orbit"
PROCESSING_PERIOD = "ORBITS=1"
PROCESSING_BOUNDARY = "START_OF_ORBIT"
PATHMAP_NAME = "MISR"

/* PGE_SSW_VERSION should match the PGE_VERSION                          */
PGE_SSW_VERSION = "21302"

OBJECT = EXIT_MESSAGE
  CLASS= 1
  EXIT_CODE = 0
  EXIT_MESSAGE = "CODE(0): Successful Completion of MISR PGE 1 AN"
END_OBJECT = EXIT_MESSAGE
OBJECT = EXIT_MESSAGE
  CLASS= 2
  EXIT_CODE = 202
  EXIT_MESSAGE = "CODE(202): Execution Failure of MISR PGE 1 AN"
END_OBJECT = EXIT_MESSAGE

/*****
/*          MISR PGE 1 AN  Inputs          */
/*  Inputs:          */
/*  LID      ESDT.Version      Science Group          */
/*****
/*          MISR PGE 1 AN Inputs          */
/*  Inputs:          */
/*  LID      ESDT.Version      Science Group          */
/*  190     MISANCGM.002      Dynamic External Input          */

```

```

/*      227      MIRCCT.001      L4003      */
/*      243      MIRCCT.001      L9001      */
/*      250      MICNFG.001      C1205      */
/*      252      MICNFG.001      C1305      */
/*      500      MISL0AN.001      Dynamic External Input      */
/*      555      MISL0SY1.001     Dynamic External Input      */
/*      556      MISL0SY2.001     Dynamic External Input      */
/*      557      MISL0SY3.001     Dynamic External Input      */
/*      599      MICNFG.001      C1415      */
/*      1101     MICNFG.001      C1005      */
/*      1120     ActSched.001     Dynamic External Input      */
/*      1301     MIANCSSC.001     C0002      */
/*      1304     MIANCAGP.001     L0002 - Path Dependent Static File      */
/*      1305     MIANPPAN.001     L1001      */
/*      1306     MIRFOIAN.001     L1001      */
/*      1334     MIB2GEOP.001     Dynamic Internal Input      */
/*      1500     MIANCARP.001     C0010      */
/*      1501     MIANCARP.001     C0011      */
/*      1502     MIANCARP.002     Dynamic External Input      */
/*      1503     MIANCARP.001     C0012      */
/*      1984     MICNFG.001      C1105      */
/*      10501    AM1EPHN0.001     Dynamic Internal Input      */
/*      10502    AM1ATTNF.001     Dynamic Internal Input      */
/*
/*****

```

```

/*      PCF Entry for 190:MISANCGM      */
/*      MISR Ancillary Dataset for Camera Model  OBJECT = PCF_ENTRY      */
CLASS = 11
LOGICAL_ID = 190
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MISANCGM"
DATA_TYPE_VERSION = "002"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
CLOSEST_QUERY_OFFSET = "WEEKS=9"
CLOSEST_QUERY_DIRECTION = "Backward"
CLOSEST_QUERY_RETRIES = 6
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 227:MIRCCT      */
/*      MISR RC Thresholds datasetOBJECT = PCF_ENTRY      */
CLASS = 12
LOGICAL_ID = 227
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MIRCCT"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "L4003"

```

```

INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*          PCF Entry for 243:MIRCCT          */
/*          MISR RC Thresholds datasetOBJECT = PCF_ENTRY          */
CLASS = 13
LOGICAL_ID = 243
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MIRCCT"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "L9001"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*          PCF Entry for 500:MISL0AN          */
/*          L0 AN data          */
OBJECT = PCF_ENTRY
CLASS = 14
LOGICAL_ID = 500
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MISL0AN"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 2
INPUT_TYPE = "Required"
/* ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y" */
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
/* 4PY version */
OBJECT = FILETYPE
    FILETYPE_NAME = "Multi-File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
OBJECT = AUXILIARY_LOGICAL_ID
    CLASS = 1
    AUX_LOGICAL_ID = 501
END_OBJECT = AUXILIARY_LOGICAL_ID
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1120:ActSched                      */
/*          Detailed Activity Schedule from EMOS            */
OBJECT = PCF_ENTRY
  CLASS = 16
  LOGICAL_ID = 1120
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "ActSched"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 2
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1301:MIANCSSC                    */
/*          MISR CSSC dataset                              */
OBJECT = PCF_ENTRY
  CLASS = 17
  LOGICAL_ID = 1301
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIANCSSC"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C0002"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1304:MIANCAGP                    */
/*          MISR Ancillary Geographic Product              */
OBJECT = PCF_ENTRY
  CLASS = 18
  LOGICAL_ID = 1304
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIANCAGP"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "L0002"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1

```

```

QUERY_TYPE = "Metadata"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
  FILETYPE_NAME = "Single File Granule"
  CLASS = 1
END_OBJECT = FILETYPE
OBJECT = METADATA_QUERY
  CLASS = 1
  PARM_NAME = "SP_AM_PATH_NO"
  OPERATOR = "=="
  VALUE = "999"
  DATABASE_QUERY = "PATH NUMBER"
END_OBJECT = METADATA_QUERY
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1305:MIANPPAN                      */
/*          MISR Project Parameters (PP) dataset             */
OBJECT = PCF_ENTRY
  CLASS = 19
  LOGICAL_ID = 1305
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIANPPAN"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "L1001"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Metadata"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
  OBJECT = METADATA_QUERY
    CLASS = 1
    PARM_NAME = "SP_AM_PATH_NO"
    OPERATOR = "=="
    VALUE = "999"
    DATABASE_QUERY = "PATH NUMBER"
  END_OBJECT = METADATA_QUERY
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 555:MISL0SY1                      */
/*          L0 Out of Synch data                            */
OBJECT = PCF_ENTRY
  CLASS = 20
  LOGICAL_ID = 555
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MISL0SY1"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 2
  INPUT_TYPE = "Optional"
  NUMBER_NEEDED = 1

```

```

QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Multi-File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
OBJECT = AUXILIARY_LOGICAL_ID
    CLASS = 1
    AUX_LOGICAL_ID = 5551
END_OBJECT = AUXILIARY_LOGICAL_ID
OBJECT = OPTIONAL_INPUT
    CLASS = 1
    CATEGORY = "Out of Sync SY1"
    ORDER = 1
    RUNTIME_PARM_ID = 555
    TIMER = "SECS=10"
    TEMPORAL = "N"
END_OBJECT = OPTIONAL_INPUT
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 556:MISL0SY2      */
/*      L0 Out of Synch data           */
OBJECT = PCF_ENTRY
    CLASS = 21
    LOGICAL_ID = 556
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MISL0SY2"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 2
    INPUT_TYPE = "Optional"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Multi-File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
    OBJECT = AUXILIARY_LOGICAL_ID
        CLASS = 1
        AUX_LOGICAL_ID = 5561
    END_OBJECT = AUXILIARY_LOGICAL_ID
    OBJECT = OPTIONAL_INPUT
        CLASS = 1
        CATEGORY = "Out of Sync SY2"
        ORDER = 1
        RUNTIME_PARM_ID = 556
        TIMER = "SECS=10"
        TEMPORAL = "N"
    END_OBJECT = OPTIONAL_INPUT
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 557:MISL0SY3      */
/*      L0 Out of Synch data           */
OBJECT = PCF_ENTRY

```

```

CLASS = 22
LOGICAL_ID = 557
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MISL0SY3"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 2
INPUT_TYPE = "Optional"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Multi-File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
OBJECT = AUXILIARY_LOGICAL_ID
    CLASS = 1
    AUX_LOGICAL_ID = 5571
END_OBJECT = AUXILIARY_LOGICAL_ID
OBJECT = OPTIONAL_INPUT
    CLASS = 1
    CATEGORY = "Out of Sync SY3"
    ORDER = 1
    RUNTIME_PARM_ID = 557
    TIMER = "SECS=10"
    TEMPORAL = "N"
END_OBJECT = OPTIONAL_INPUT
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 1306:MIRFOIAN      */
/*      MISR Reference Orbit Imagery     */
OBJECT = PCF_ENTRY
CLASS = 110
LOGICAL_ID = 1306
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MIRFOIAN"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "L1001"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Metadata"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
OBJECT = METADATA_QUERY
    CLASS = 1
    PARM_NAME = "SP_AM_PATH_NO"
    OPERATOR = "=="
    VALUE = "999"
    DATABASE_QUERY = "PATH NUMBER"
END_OBJECT = METADATA_QUERY
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1334:MIB2GEOP          */
/*          MISR Geometric Parameters            */
OBJECT = PCF_ENTRY
  CLASS = 111
  LOGICAL_ID = 1334
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIB2GEOP"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1500:MIANCARP          */
/*          MISR Ancillary Radiometric Product  */
OBJECT = PCF_ENTRY
  CLASS = 112
  LOGICAL_ID = 1500
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIANCARP"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C0010"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1501:MIANCARP          */
/*          MISR Ancillary Radiometric Product  */
OBJECT = PCF_ENTRY
  CLASS = 113
  LOGICAL_ID = 1501
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MIANCARP"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C0011"
  INPUT_TYPE = "Required"

```

```
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1502:MIANCARP      */
/*      MISR Ancillary Radiometric Product      */
OBJECT = PCF_ENTRY
CLASS = 114
LOGICAL_ID = 1502
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MIANCARP"
DATA_TYPE_VERSION = "002"
MIN_GANULES_REQUIRED = 1
MAX_GANULES_REQUIRED = 1
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
CLOSEST_QUERY_OFFSET = "WEEKS=8"
CLOSEST_QUERY_DIRECTION = "Backward"
CLOSEST_QUERY_RETRIES = 10
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1503:MIANCARP      */
/*      MISR Ancillary Radiometric Product      */
OBJECT = PCF_ENTRY
CLASS = 115
LOGICAL_ID = 1503
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MIANCARP"
DATA_TYPE_VERSION = "001"
MIN_GANULES_REQUIRED = 1
MAX_GANULES_REQUIRED = 1
SCIENCE_GROUP = "C0012"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```

/*          PCF Entry for 250:MICNFG          */
/*          MISR RCCM configuration file       */
OBJECT = PCF_ENTRY
  CLASS = 116
  LOGICAL_ID = 250
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MICNFG"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C1205"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 252:MICNFG          */
/*          MISR GRP configuration file       */
OBJECT = PCF_ENTRY
  CLASS = 117
  LOGICAL_ID = 252
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MICNFG"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C1305"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 599:MICNFG          */
/*          MISR RAP configuration file       */
OBJECT = PCF_ENTRY
  CLASS = 118
  LOGICAL_ID = 599
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MICNFG"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C1415"
  INPUT_TYPE = "Required"

```

```

NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 1984:MICNFG      */
/*      MISR RP configuration file      */
OBJECT = PCF_ENTRY
CLASS = 119
LOGICAL_ID = 1984
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MICNFG"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "C1105"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/**** Attitude/Ephemeris/DEM entry. Please delete if not used by PGE. **/
/*      PCF Entry for 10501:AM1EPHN0      */
/*      Ephemeris data generated from DPREP      */
/*      External Data Source      */
OBJECT = PCF_ENTRY
CLASS = 120
LOGICAL_ID = 10501
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "AM1EPHN0"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 2
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/**** Attitude/Ephemeris/DEM entry. Please delete if not used by PGE. **/
/*      PCF Entry for 10502:AM1ATTNF                                */
/*      Attitude data generated by DPREP                          */
/*      External Data Source                                        */
OBJECT = PCF_ENTRY
  CLASS = 121
  LOGICAL_ID = 10502
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "AM1ATTNF"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 2
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 1101:MICNFG                                */
/*      MISR PCS configuration file                                */
OBJECT = PCF_ENTRY
  CLASS = 132
  LOGICAL_ID = 1101
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "MICNFG"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  SCIENCE_GROUP = "C1005"
  INPUT_TYPE = "Required"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*****
/* MISR PGE 1 AN Outputs                                          */
/*      Output:                                                  */
/*      LID      ESDT.Version   Science Group   Associated MCF      */
/*      251     MIANRCCH.002   S4             1136            */
/*      600     MIL1A.001     S5             1130            */
/*      610     MISBR.002    S6             1138            */
/*      611     MISBR.002    S7             1138            */
/*      650     MISQA.002    S10            1137            */
/*      1375    MI1B2T.001   S2             1133            */
/*      1376    MI1B2E.001   S1             1134            */
/*      1377    MIRCCM.001   S3             1135            */
/*      1335    MISQA.002    S11            11371           */
/*      1336    MISQA.002    S12            11372           */

```

```

/*      1337      MISQA.002      S13      11373      */
/*      1976      MI1B1.001      S8      1140      */
/*      1983      MIB1LM.001      S9      1131      */
/*      1985      MISQA.002      S14      11374      */
/*      1986      MISQA.002      S15      11375      */
/*
/*****

```

```

/*      PCF Entry for 251:MIANRCCH      */
/*      MISR RC Histogram file      */
OBJECT = PCF_ENTRY
CLASS = 136
LOGICAL_ID = 251
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MIANRCCH"
DATA_TYPE_VERSION = "002"
MIN GRANULE YIELD = 1
MAX GRANULE YIELD = 1
ASSOCIATED_MCF_ID = 1136
SCIENCE_GROUP = "S4"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 600:MIL1A      */
/*      MISR L1A Product      */
OBJECT = PCF_ENTRY
CLASS = 137
LOGICAL_ID = 600
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MIL1A"
DATA_TYPE_VERSION = "001"
MIN GRANULE YIELD = 1
MAX GRANULE YIELD = 1
ASSOCIATED_MCF_ID = 1130
SCIENCE_GROUP = "S5"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
FILETYPE_NAME = "Single File Granule"
CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 610:MISBR      */
/*      MISR Browse data HDF file      */
OBJECT = PCF_ENTRY
CLASS = 138
LOGICAL_ID = 610
PCF_FILE_TYPE = 2

```

```

DATA_TYPE_NAME = "MISBR"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 1138
SCIENCE_GROUP = "S6"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 611:MISBR      */
/*      MISR Browse data JPEG file  */
OBJECT = PCF_ENTRY
CLASS = 139
LOGICAL_ID = 611
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MISBR"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 1138
SCIENCE_GROUP = "S7"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*      PCF Entry for 650:MISQA      */
/*      MISR L1A QA                  */
OBJECT = PCF_ENTRY
CLASS = 140
LOGICAL_ID = 650
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MISQA"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 1137
SCIENCE_GROUP = "S10"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1375:MI1B2T          */
/*          MISR L1B2 Terrain data             */
OBJECT = PCF_ENTRY
  CLASS = 141
  LOGICAL_ID = 1375
  PCF_FILE_TYPE = 2
  DATA_TYPE_NAME = "MI1B2T"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULE_YIELD = 1
  MAX_GRANULE_YIELD = 1
  ASSOCIATED_MCF_ID = 1133
  SCIENCE_GROUP = "S2"
  INSTANCE = 0
  MINIMUM_SIZE = 0
  MAXIMUM_SIZE = 0
  DISTINCT_VALUE = "AN"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1376:MI1B2E          */
/*          MISR L1B2 Ellipsoid data           */
OBJECT = PCF_ENTRY
  CLASS = 142
  LOGICAL_ID = 1376
  PCF_FILE_TYPE = 2
  DATA_TYPE_NAME = "MI1B2E"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULE_YIELD = 1
  MAX_GRANULE_YIELD = 1
  ASSOCIATED_MCF_ID = 1134
  SCIENCE_GROUP = "S1"
  INSTANCE = 0
  MINIMUM_SIZE = 0
  MAXIMUM_SIZE = 0
  DISTINCT_VALUE = "AN"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1377:MIRCCM          */
/*          MISR L1B2 RCCM data                */
OBJECT = PCF_ENTRY
  CLASS = 143
  LOGICAL_ID = 1377
  PCF_FILE_TYPE = 2
  DATA_TYPE_NAME = "MIRCCM"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULE_YIELD = 1
  MAX_GRANULE_YIELD = 1
  ASSOCIATED_MCF_ID = 1135

```

```
SCIENCE_GROUP = "S3"  
INSTANCE = 0  
MINIMUM_SIZE = 0  
MAXIMUM_SIZE = 0  
DISTINCT_VALUE = "AN"  
OBJECT = FILETYPE  
    FILETYPE_NAME = "Single File Granule"  
    CLASS = 1  
END_OBJECT = FILETYPE  
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1335:MISQA      */  
/*      MISR L1B2 Terrain QA data    */  
OBJECT = PCF_ENTRY  
    CLASS = 144  
    LOGICAL_ID = 1335  
    PCF_FILE_TYPE = 2  
    DATA_TYPE_NAME = "MISQA"  
    DATA_TYPE_VERSION = "002"  
    MIN_GRANULE_YIELD = 0  
    MAX_GRANULE_YIELD = 1  
    ASSOCIATED_MCF_ID = 11371  
    SCIENCE_GROUP = "S11"  
    INSTANCE = 0  
    MINIMUM_SIZE = 0  
    MAXIMUM_SIZE = 0  
    OBJECT = FILETYPE  
        FILETYPE_NAME = "Single File Granule"  
        CLASS = 1  
    END_OBJECT = FILETYPE  
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1336:MISQA      */  
/*      MISR L1B2 Ellipsoid QA data   */  
OBJECT = PCF_ENTRY  
    CLASS = 145  
    LOGICAL_ID = 1336  
    PCF_FILE_TYPE = 2  
    DATA_TYPE_NAME = "MISQA"  
    DATA_TYPE_VERSION = "002"  
    MIN_GRANULE_YIELD = 0  
    MAX_GRANULE_YIELD = 1  
    ASSOCIATED_MCF_ID = 11372  
    SCIENCE_GROUP = "S12"  
    INSTANCE = 0  
    MINIMUM_SIZE = 0  
    MAXIMUM_SIZE = 0  
    OBJECT = FILETYPE  
        FILETYPE_NAME = "Single File Granule"  
        CLASS = 1  
    END_OBJECT = FILETYPE  
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1337:MISQA      */  
/*      MISR L1B2 RCCM QA data       */  
OBJECT = PCF_ENTRY
```

```

CLASS = 146
LOGICAL_ID = 1337
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MISQA"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 11373
SCIENCE_GROUP = "S13"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1976:MI1B1          */
/*          MISR L1B1 Radiometric Product     */
OBJECT = PCF_ENTRY
CLASS = 147
LOGICAL_ID = 1976
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MI1B1"
DATA_TYPE_VERSION = "001"
MIN_GRANULE_YIELD = 1
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 1140
SCIENCE_GROUP = "S8"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

```

```

/*          PCF Entry for 1983:MIB1LM        */
/*          MISR L1B1 Local Mode data         */
OBJECT = PCF_ENTRY
CLASS = 148
LOGICAL_ID = 1983
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MIB1LM"
DATA_TYPE_VERSION = "001"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 6
ASSOCIATED_MCF_ID = 1131
SCIENCE_GROUP = "S9"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1

```

```
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1985:MISQA      */
/*      MISR L1B1 QA data             */
OBJECT = PCF_ENTRY
CLASS = 149
LOGICAL_ID = 1985
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MISQA"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 11374
SCIENCE_GROUP = "S14"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
/*      PCF Entry for 1986:MISQA      */
/*      MISR L1B1 Local Mode QA       */
OBJECT = PCF_ENTRY
CLASS = 150
LOGICAL_ID = 1986
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MISQA"
DATA_TYPE_VERSION = "002"
MIN_GRANULE_YIELD = 0
MAX_GRANULE_YIELD = 6
ASSOCIATED_MCF_ID = 11375
SCIENCE_GROUP = "S15"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
CLASS = 151
LOGICAL_ID = 292
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Product version"
PGE_PARAMETER_DEFAULT = "0007"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
```

```

CLASS = 152
LOGICAL_ID = 295
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Camera"
PGE_PARAMETER_DEFAULT = "An"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 153
LOGICAL_ID = 620
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Ascii met file for HDF browse"
PGE_PARAMETER_DEFAULT = "610:1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 154
LOGICAL_ID = 621
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Ascii met file for JPEG browse"
PGE_PARAMETER_DEFAULT = "611:1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 155
LOGICAL_ID = 1102
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "which pge"
PGE_PARAMETER_DEFAULT = "MISR_PGE01"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 156
LOGICAL_ID = 1104
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Orbit number"
PGE_PARAMETER_DEFAULT = "999999"
PGE_PARAMETER_DYNAMIC_VALUE = "ORBIT NUMBER"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 157
LOGICAL_ID = 1103
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Orbit path"
PGE_PARAMETER_DEFAULT = "999"
PGE_PARAMETER_DYNAMIC_VALUE = "PATH NUMBER"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 158
LOGICAL_ID = 10119

```

```

    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Disabled status code list"
    PGE_PARAMETER_DEFAULT = "35870 163843126 163843127 163842611 163842612
166300169 164662287"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

END

```

### C.3.2 MISR ESDT MISANCGM ODL

```

DATA_TYPE_NAME = "MISANCGM"
DATA_TYPE_VERSION = "002"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "Camera Geometric Model for Level 1B2"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 1.0
PROCESSING_LEVEL = "L1"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "NONROUTINE"
SUPPLIER_NAME = "LARC"
/* BOUNDARY = "START_OF_YEAR" */
/* PERIOD = "YEARS=5" */
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

### C.3.3 MISR ESDT MIRCCT ODL

```

DATA_TYPE_NAME = "MIRCCT"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Radiometric Camera-by-Camera Threshold dataset"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 10.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"

END

```

### C.3.4 MISR ESDT MISLOAN ODL

```
DATA_TYPE_NAME = "MISLOAN"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Level 0 CCD Science Data AN Camera"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 1000.0
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
/* PERIOD = "ORBITS=1" */
PERIOD = "HOURS=2"
/* BOUNDARY = "START_OF_ORBIT" */
BOUNDARY = "START_OF_DAY"
DURATION = "HOURS=2"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = FILETYPE
    CLASS = 2
    FILETYPE_NAME = "Multi-File Granule"
    MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE

END
```

### C.3.5 MISR ESDT ActSched ODL

```
DATA_TYPE_NAME = "ActSched"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "Detailed Activity Schedule"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 1.0
/* Changed by Jim Galasso 10/9/1999 */
/* Processing Level cannot be L0 multiple files using the same LID */
/* Change of Processing level is to support PGE processing when 2 DAS */
/* files are required because the PGE's DPR times span 2 files */
PROCESSING_LEVEL = "SCHED"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
/* Q: Should the supplier of the DAS be identified as EMOS? */
SUPPLIER_NAME = "EMOS"
PERIOD = "DAYS=1"
/* Boundary set for DAS files to be 2000 to 2000 each day */
BOUNDARY = "START_OF_DAY-14400"
DURATION = "HOURS=24"
```

```
DELAY = 3600
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END
```

### **C.3.6 MISR ESDT MIANCSSC ODL**

```
DATA_TYPE_NAME = "MIANCSSC"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Cloud Screening Surface Classification dataset"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 5.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END
```

### **C.3.7 MISR ESDT MIANCAGP ODL**

```
DATA_TYPE_NAME = "MIANCAGP"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Ancillary Geographic Product"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 110.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "Y"
OBJECT = METADATA_DEFINITION
  CLASS = 1
  PARM_NAME = "SP_AM_PATH_NO"
  CONTAINER_NAME = "AdditionalAttributes"
  TYPE = "STR"
END_OBJECT = METADATA_DEFINITION
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT

END
```

### C.3.8 MISR ESDT MIANPPAN ODL

```
DATA_TYPE_NAME = "MIANPPAN"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Projection Parameters Ancillary Dataset, Camera
AN"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 310.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "Y"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
OBJECT = METADATA_DEFINITION
    CLASS = 1
    PARM_NAME = "SP_AM_PATH_NO"
    CONTAINER_NAME = "AdditionalAttributes"
    TYPE = "STR"
END_OBJECT = METADATA_DEFINITION

END
```

### C.3.9 MISR ESDT MISL0SY1 ODL

```
DATA_TYPE_NAME = "MISL0SY1"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Out of Sync L0 CCSDS packets for APID = 373"
PROVIDER = "Langley Research Center"
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
/* Q: NOMINAL_SIZE ???? */
NOMINAL_SIZE = 5.9
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
PERIOD = "ORBITS=1"
/* PERIOD = "HOURS=2" */
BOUNDARY = "START_OF_ORBIT"
/* BOUNDARY = "START_OF_DAY+3600" */
DURATION = "ORBITS=1"
/* DURATION = "HOURS=2" */
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = FILETYPE
```

```

CLASS = 2
FILETYPE_NAME = "Multi-File Granule"
MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE

END

```

### C.3.10 MISR ESDT MISL0SY2 ODL

```

DATA_TYPE_NAME = "MISL0SY2"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Out of Sync L0 CCSDS packets for APID = 374"
PROVIDER = "Langley Research Center"
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
/* Q: NOMINAL_SIZE ????? */
NOMINAL_SIZE = 5.9
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
PERIOD = "ORBITS=1"
/* PERIOD = "HOURS=2" */
BOUNDARY = "START_OF_ORBIT"
/* BOUNDARY = "START_OF_DAY+3600" */
DURATION = "ORBITS=1"
/* DURATION = "HOURS=2" */
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
CLASS = 1
USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = FILETYPE
CLASS = 2
FILETYPE_NAME = "Multi-File Granule"
MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE

END

```

### C.3.11 MISR ESDT MISL0SY3 ODL

```

DATA_TYPE_NAME = "MISL0SY3"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Out of Sync L0 CCSDS packets for APID = 378"
PROVIDER = "Langley Research Center"
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
/* Q: NOMINAL_SIZE ????? */
NOMINAL_SIZE = 5.9
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"

```

```

SUPPLIER_NAME = "LARC"
PERIOD = "ORBITS=1"
/* PERIOD = "HOURS=2" */
BOUNDARY = "START_OF_ORBIT"
/* BOUNDARY = "START_OF_DAY+3600" */
DURATION = "ORBITS=1"
/* DURATION = "HOURS=2" */
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = FILETYPE
  CLASS = 2
  FILETYPE_NAME = "Multi-File Granule"
  MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE

END

```

### C.3.12 MISR ESDT MIRFOIAN ODL

```

DATA_TYPE_NAME = "MIRFOIAN"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Reference Orbit Imagery Ancillary Dataset,
Camera AN"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 280.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
OBJECT = METADATA_DEFINITION
  CLASS = 1
  PARM_NAME = "SP_AM_PATH_NO"
  CONTAINER_NAME = "AdditionalAttributes"
  TYPE = "STR"
END_OBJECT = METADATA_DEFINITION
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

### C.3.13 MISR ESDT MIB2GEOP ODL

```

DATA_TYPE_NAME = "MIB2GEOP"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"

```

```

DATA_TYPE_DESCRIPTION = "MISR Geometric Parameters"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 6.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
SPATIAL_FLAG = "Y"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

### **C.3.14 MISR ESDT MIANCARP ODL (Version# 001)**

```

DATA_TYPE_NAME = "MIANCARP"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Ancillary Radiometric Product (ARP)"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 5.0
PROCESSING_LEVEL = "All"
HDF_DATA = "Y"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT

END

```

### **C.3.15 MISR ESDT MIANCARP ODL (Version# 002)**

```

DATA_TYPE_NAME = "MIANCARP"
DATA_TYPE_VERSION = "002"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Ancillary Radiometric Product (ARP)"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 5.0
PROCESSING_LEVEL = "All"
HDF_DATA = "Y"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "NONROUTINE"
/* PERIOD = "MONTHS=2" */
/* BOUNDARY = "START_OF_MONTH" */
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "LARC"
END_OBJECT = USE_OBJECT

```

```
ARCHIVED_AT = "LARC"  
PROCESSED_AT = "LARC"
```

```
END
```

### **C.3.16 MISR ESDT MICNFG ODL**

```
DATA_TYPE_NAME = "MICNFG"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MISR"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "MISR Configuration File for all PGEs"  
PROVIDER = "Langley Research Center"  
NOMINAL_SIZE = 0.5  
PROCESSING_LEVEL = "All"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "LARC"  
END_OBJECT = USE_OBJECT
```

```
END
```

### **C.3.17 MISR ESDT AM1EPHN0 ODL**

```
DATA_TYPE_NAME = "AM1EPHN0"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "All"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "AM-1 L0/FDD Ephemeris data in Toolkit format"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 2.0  
PROCESSING_LEVEL = "L1"  
DYNAMIC_FLAG = "I"  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "GSFC"  
END_OBJECT = USE_OBJECT  
ARCHIVED_AT = "GSFC"  
PROCESSED_AT = "GSFC"  
HDF_DATA = "N"
```

```
END
```

### **C.3.18 MISR ESDT AM1ATTNF ODL**

```
DATA_TYPE_NAME = "AM1ATTNF"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "All"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "AM-1 FDD Attitude data in Toolkit format"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 2.0
```

```

PROCESSING_LEVEL = "L1"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
HDF_DATA = "N"

END

```

### C.3.19 MISR ESDT MIANRCCH ODL

```

DATA_TYPE_NAME = "MIANRCCH"
DATA_TYPE_VERSION = "002"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Radiometric Camera-by-Camera Histogram Dataset"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 3.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "N"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

### C.3.20 MISR ESDT MIL1A ODL

```

DATA_TYPE_NAME = "MIL1A"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Level 1A CCD Science data, all cameras"
PROVIDER = "Langley Research Center"
/*      Q: Need to find the correct nominal file size for MIL1A          */
/* NOMINAL_SIZE = 498.0 */
/* NOMINAL_SIZE = 12000.0 */
/* NOMINAL_SIZE = 100.0 */
NOMINAL_SIZE = 1500.0
PROCESSING_LEVEL = "L1A"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT

```

```
CLASS = 1
USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
```

END

### C.3.21 MISR ESDT MISBR ODL

```
DATA_TYPE_NAME = "MISBR"
DATA_TYPE_VERSION = "002"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Browse data for use with systematic QA analysis"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 3.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
CLASS = 1
USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
```

END

### C.3.22 MISR ESDT MISQA ODL

```
DATA_TYPE_NAME = "MISQA"
DATA_TYPE_VERSION = "002"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Quality Assurance data"
PROVIDER = "Langley Research Center"
/* Increased to 20.0 from 1.0 by Jim Galasso 10/9/1999 */
NOMINAL_SIZE = 20.0
/* Changed to Processing Level all 10/9/1999 */
PROCESSING_LEVEL = "ALL"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
PERIOD = "ORBITS=1"
BOUNDARY = "START_OF_ORBIT"
DURATION = "HOURS=2"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
CLASS = 1
USED_BY = "LARC"
END_OBJECT = USE_OBJECT
```

```
ARCHIVED_AT = "LARC"  
PROCESSED_AT = "LARC"
```

```
END
```

### C.3.23 MISR ESDT MI1B2T ODL

```
DATA_TYPE_NAME = "MI1B2T"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MISR"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "MISR Level 1B2 Terrain Data"  
PROVIDER = "Langley Research Center"  
NOMINAL_SIZE = 400.0  
PROCESSING_LEVEL = "L1B2"  
HDF_DATA = "Y"  
DYNAMIC_FLAG = "I"  
PREDICTION_METHOD = "ROUTINE"  
SUPPLIER_NAME = "LARC"  
DELAY = 1  
SPATIAL_FLAG = "Y"  
DISTINCT_PARAMETER = "AssociatedSensorShortName"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "LARC"  
END_OBJECT = USE_OBJECT  
ARCHIVED_AT = "LARC"  
PROCESSED_AT = "LARC"  
OBJECT = METADATA_DEFINITION  
    CLASS = 2  
    PARM_NAME = "AssociatedSensorShortName"  
    CONTAINER_NAME = "AssociatedPlatformInstrumentSensor"  
    TYPE = "STR"  
END_OBJECT = METADATA_DEFINITION
```

```
END
```

### C.3.24 MISR ESDT MI1B2E ODL

```
DATA_TYPE_NAME = "MI1B2E"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MISR"  
PLATFORM = "AM1"  
DATA_TYPE_DESCRIPTION = "MISR Level 1B2 Ellipsoid Data"  
PROVIDER = "Langley Research Center"  
NOMINAL_SIZE = 700.0  
PROCESSING_LEVEL = "L1B2"  
HDF_DATA = "Y"  
DYNAMIC_FLAG = "I"  
PREDICTION_METHOD = "ROUTINE"  
SUPPLIER_NAME = "LARC"  
DELAY = 1  
SPATIAL_FLAG = "Y"  
DISTINCT_PARAMETER = "AssociatedSensorShortName"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "LARC"  
END_OBJECT = USE_OBJECT
```

```

ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = METADATA_DEFINITION
  CLASS = 2
  PARM_NAME = "AssociatedSensorShortName"
  CONTAINER_NAME = "AssociatedPlatformInstrumentSensor"
  TYPE = "STR"
END_OBJECT = METADATA_DEFINITION

END

```

### C.3.25 MISR ESDT MIRCCM ODL

```

DATA_TYPE_NAME = "MIRCCM"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR radiometric camera-by-camera Cloud Mask"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 3.0
PROCESSING_LEVEL = "L1B2"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "Y"
DISTINCT_PARAMETER = "AssociatedSensorShortName"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"
OBJECT = METADATA_DEFINITION
  CLASS = 2
  PARM_NAME = "AssociatedSensorShortName"
  CONTAINER_NAME = "AssociatedPlatformInstrumentSensor"
  TYPE = "STR"
END_OBJECT = METADATA_DEFINITION

END

```

### C.3.26 MISR ESDT MI1B1 ODL

```

DATA_TYPE_NAME = "MI1B1"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Level 1B2 Ellipsoid Data"
PROVIDER = "Langley Research Center"
/* NOMINAL_SIZE = 574.0 */
/* changed for FILEWATCHER! */
/* NOMINAL_SIZE = 12000.0 */
/* NOMINAL_SIZE = 100.0 */
NOMINAL_SIZE = 1500.0
PROCESSING_LEVEL = "L1B1"
HDF_DATA = "Y"

```

```

DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

### C.3.27 MISR ESDT MIB1LM ODL

```

DATA_TYPE_NAME = "MIB1LM"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MISR"
PLATFORM = "AM1"
DATA_TYPE_DESCRIPTION = "MISR Level 1B1 Local Mode Radiance Data"
PROVIDER = "Langley Research Center"
NOMINAL_SIZE = 100.0
PROCESSING_LEVEL = "L1"
HDF_DATA = "Y"
DYNAMIC_FLAG = "I"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "LARC"
DELAY = 1
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "LARC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "LARC"
PROCESSED_AT = "LARC"

END

```

```

C.3.28 MISR ORBIT ODLPLATFORM = "AM1"
OBJECT = ORBIT_MODEL
  CLASS = 1

```

```

/* ORBIT_NUMBER = 7472 */
ORBIT_NUMBER = 7496

/* Cross correlate using PATHMAP_MISR.odl file */

/* ORBIT_PATH_NUMBER = 28 */
ORBIT_PATH_NUMBER = 52

/* ORBIT_PERIOD = "SECS=5934" */
ORBIT_PERIOD = "SECS=5933"

/* ORBIT_START = "05/14/2001 10:41:32" */
ORBIT_START = "05/16/2001 02:14:44"

```

END\_OBJECT = ORBIT\_MODEL

END

C.3.29 MISR PATHMAP ODLPLATFORM = "AM1"

PATHMAP\_NAME = "MISR"

OBJECT = PATHMAP\_ENTRY

CLASS = 1

ABSOLUTE\_PATH = 1

MAPPED\_PATH = 1

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 2

ABSOLUTE\_PATH = 2

MAPPED\_PATH = 17

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 3

ABSOLUTE\_PATH = 3

MAPPED\_PATH = 33

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 4

ABSOLUTE\_PATH = 4

MAPPED\_PATH = 49

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 5

ABSOLUTE\_PATH = 5

MAPPED\_PATH = 65

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 6

ABSOLUTE\_PATH = 6

MAPPED\_PATH = 81

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 7

ABSOLUTE\_PATH = 7

MAPPED\_PATH = 97

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 8

ABSOLUTE\_PATH = 8

MAPPED\_PATH = 113

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 9

ABSOLUTE\_PATH = 9

MAPPED\_PATH = 129

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 10

ABSOLUTE\_PATH = 10

MAPPED\_PATH = 145

END\_OBJECT = PATHMAP\_ENTRY

OBJECT = PATHMAP\_ENTRY

CLASS = 11

ABSOLUTE\_PATH = 11

MAPPED\_PATH = 161

```
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 12
  ABSOLUTE_PATH = 12
  MAPPED_PATH = 177
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 13
  ABSOLUTE_PATH = 13
  MAPPED_PATH = 193
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 14
  ABSOLUTE_PATH = 14
  MAPPED_PATH = 209
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 15
  ABSOLUTE_PATH = 15
  MAPPED_PATH = 225
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 16
  ABSOLUTE_PATH = 16
  MAPPED_PATH = 8
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 17
  ABSOLUTE_PATH = 17
  MAPPED_PATH = 24
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 18
  ABSOLUTE_PATH = 18
  MAPPED_PATH = 40
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 19
  ABSOLUTE_PATH = 19
  MAPPED_PATH = 56
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 20
  ABSOLUTE_PATH = 20
  MAPPED_PATH = 72
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 21
  ABSOLUTE_PATH = 21
  MAPPED_PATH = 88
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 22
  ABSOLUTE_PATH = 22
  MAPPED_PATH = 104
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 23
  ABSOLUTE_PATH = 23
  MAPPED_PATH = 120
END_OBJECT = PATHMAP_ENTRY
```

```
OBJECT = PATHMAP_ENTRY
  CLASS = 24
  ABSOLUTE_PATH = 24
  MAPPED_PATH = 136
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 25
  ABSOLUTE_PATH = 25
  MAPPED_PATH = 152
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 26
  ABSOLUTE_PATH = 26
  MAPPED_PATH = 168
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 27
  ABSOLUTE_PATH = 27
  MAPPED_PATH = 184
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 28
  ABSOLUTE_PATH = 28
  MAPPED_PATH = 200
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 29
  ABSOLUTE_PATH = 29
  MAPPED_PATH = 216
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 30
  ABSOLUTE_PATH = 30
  MAPPED_PATH = 232
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 31
  ABSOLUTE_PATH = 31
  MAPPED_PATH = 15
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 32
  ABSOLUTE_PATH = 32
  MAPPED_PATH = 31
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 33
  ABSOLUTE_PATH = 33
  MAPPED_PATH = 47
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 34
  ABSOLUTE_PATH = 34
  MAPPED_PATH = 63
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 35
  ABSOLUTE_PATH = 35
  MAPPED_PATH = 79
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
```

```
CLASS = 36
  ABSOLUTE_PATH = 36
  MAPPED_PATH = 95
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 37
  ABSOLUTE_PATH = 37
  MAPPED_PATH = 111
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 38
  ABSOLUTE_PATH = 38
  MAPPED_PATH = 127
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 39
  ABSOLUTE_PATH = 39
  MAPPED_PATH = 143
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 40
  ABSOLUTE_PATH = 40
  MAPPED_PATH = 159
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 41
  ABSOLUTE_PATH = 41
  MAPPED_PATH = 175
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 42
  ABSOLUTE_PATH = 42
  MAPPED_PATH = 191
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 43
  ABSOLUTE_PATH = 43
  MAPPED_PATH = 207
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 44
  ABSOLUTE_PATH = 44
  MAPPED_PATH = 223
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 45
  ABSOLUTE_PATH = 45
  MAPPED_PATH = 6
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 46
  ABSOLUTE_PATH = 46
  MAPPED_PATH = 22
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 47
  ABSOLUTE_PATH = 47
  MAPPED_PATH = 38
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 48
```

```
    ABSOLUTE_PATH = 48
    MAPPED_PATH = 54
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 49
    ABSOLUTE_PATH = 49
    MAPPED_PATH = 70
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 50
    ABSOLUTE_PATH = 50
    MAPPED_PATH = 86
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 51
    ABSOLUTE_PATH = 51
    MAPPED_PATH = 102
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 52
    ABSOLUTE_PATH = 52
    MAPPED_PATH = 118
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 53
    ABSOLUTE_PATH = 53
    MAPPED_PATH = 134
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 54
    ABSOLUTE_PATH = 54
    MAPPED_PATH = 150
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 55
    ABSOLUTE_PATH = 55
    MAPPED_PATH = 166
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 56
    ABSOLUTE_PATH = 56
    MAPPED_PATH = 182
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 57
    ABSOLUTE_PATH = 57
    MAPPED_PATH = 198
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 58
    ABSOLUTE_PATH = 58
    MAPPED_PATH = 214
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 59
    ABSOLUTE_PATH = 59
    MAPPED_PATH = 230
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 60
    ABSOLUTE_PATH = 60
```

```

    MAPPED_PATH = 13
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 61
    ABSOLUTE_PATH = 61
    MAPPED_PATH = 29
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 62
    ABSOLUTE_PATH = 62
    MAPPED_PATH = 45
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 63
    ABSOLUTE_PATH = 63
    MAPPED_PATH = 61
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 64
    ABSOLUTE_PATH = 64
    MAPPED_PATH = 77
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 65
    ABSOLUTE_PATH = 65
    MAPPED_PATH = 93
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 66
    ABSOLUTE_PATH = 66
    MAPPED_PATH = 109
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 67
    ABSOLUTE_PATH = 67
    MAPPED_PATH = 125
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 68
    ABSOLUTE_PATH = 68
    MAPPED_PATH = 141
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 69
    ABSOLUTE_PATH = 69
    MAPPED_PATH = 157
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 70
    ABSOLUTE_PATH = 70
    MAPPED_PATH = 173
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 71
    ABSOLUTE_PATH = 71
    MAPPED_PATH = 189
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 72
    ABSOLUTE_PATH = 72
    MAPPED_PATH = 205

```

```
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 73
  ABSOLUTE_PATH = 73
  MAPPED_PATH = 221
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 74
  ABSOLUTE_PATH = 74
  MAPPED_PATH = 4
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 75
  ABSOLUTE_PATH = 75
  MAPPED_PATH = 20
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 76
  ABSOLUTE_PATH = 76
  MAPPED_PATH = 36
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 77
  ABSOLUTE_PATH = 77
  MAPPED_PATH = 52
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 78
  ABSOLUTE_PATH = 78
  MAPPED_PATH = 68
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 79
  ABSOLUTE_PATH = 79
  MAPPED_PATH = 84
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 80
  ABSOLUTE_PATH = 80
  MAPPED_PATH = 100
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 81
  ABSOLUTE_PATH = 81
  MAPPED_PATH = 116
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 82
  ABSOLUTE_PATH = 82
  MAPPED_PATH = 132
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 83
  ABSOLUTE_PATH = 83
  MAPPED_PATH = 148
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 84
  ABSOLUTE_PATH = 84
  MAPPED_PATH = 164
END_OBJECT = PATHMAP_ENTRY
```

```
OBJECT = PATHMAP_ENTRY
  CLASS = 85
  ABSOLUTE_PATH = 85
  MAPPED_PATH = 180
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 86
  ABSOLUTE_PATH = 86
  MAPPED_PATH = 196
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 87
  ABSOLUTE_PATH = 87
  MAPPED_PATH = 212
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 88
  ABSOLUTE_PATH = 88
  MAPPED_PATH = 228
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 89
  ABSOLUTE_PATH = 89
  MAPPED_PATH = 11
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 90
  ABSOLUTE_PATH = 90
  MAPPED_PATH = 27
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 91
  ABSOLUTE_PATH = 91
  MAPPED_PATH = 43
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 92
  ABSOLUTE_PATH = 92
  MAPPED_PATH = 59
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 93
  ABSOLUTE_PATH = 93
  MAPPED_PATH = 75
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 94
  ABSOLUTE_PATH = 94
  MAPPED_PATH = 91
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 95
  ABSOLUTE_PATH = 95
  MAPPED_PATH = 107
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 96
  ABSOLUTE_PATH = 96
  MAPPED_PATH = 123
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
```

```
CLASS = 97
  ABSOLUTE_PATH = 97
  MAPPED_PATH = 139
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 98
  ABSOLUTE_PATH = 98
  MAPPED_PATH = 155
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 99
  ABSOLUTE_PATH = 99
  MAPPED_PATH = 171
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 100
  ABSOLUTE_PATH = 100
  MAPPED_PATH = 187
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 101
  ABSOLUTE_PATH = 101
  MAPPED_PATH = 203
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 102
  ABSOLUTE_PATH = 102
  MAPPED_PATH = 219
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 103
  ABSOLUTE_PATH = 103
  MAPPED_PATH = 2
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 104
  ABSOLUTE_PATH = 104
  MAPPED_PATH = 18
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 105
  ABSOLUTE_PATH = 105
  MAPPED_PATH = 34
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 106
  ABSOLUTE_PATH = 106
  MAPPED_PATH = 50
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 107
  ABSOLUTE_PATH = 107
  MAPPED_PATH = 66
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 108
  ABSOLUTE_PATH = 108
  MAPPED_PATH = 82
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 109
```

```
ABSOLUTE_PATH = 109
MAPPED_PATH = 98
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 110
ABSOLUTE_PATH = 110
MAPPED_PATH = 114
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 111
ABSOLUTE_PATH = 111
MAPPED_PATH = 130
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 112
ABSOLUTE_PATH = 112
MAPPED_PATH = 146
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 113
ABSOLUTE_PATH = 113
MAPPED_PATH = 162
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 114
ABSOLUTE_PATH = 114
MAPPED_PATH = 178
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 115
ABSOLUTE_PATH = 115
MAPPED_PATH = 194
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 116
ABSOLUTE_PATH = 116
MAPPED_PATH = 210
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 117
ABSOLUTE_PATH = 117
MAPPED_PATH = 226
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 118
ABSOLUTE_PATH = 118
MAPPED_PATH = 9
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 119
ABSOLUTE_PATH = 119
MAPPED_PATH = 25
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 120
ABSOLUTE_PATH = 120
MAPPED_PATH = 41
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 121
ABSOLUTE_PATH = 121
```

```
MAPPED_PATH = 57
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 122
ABSOLUTE_PATH = 122
MAPPED_PATH = 73
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 123
ABSOLUTE_PATH = 123
MAPPED_PATH = 89
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 124
ABSOLUTE_PATH = 124
MAPPED_PATH = 105
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 125
ABSOLUTE_PATH = 125
MAPPED_PATH = 121
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 126
ABSOLUTE_PATH = 126
MAPPED_PATH = 137
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 127
ABSOLUTE_PATH = 127
MAPPED_PATH = 153
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 128
ABSOLUTE_PATH = 128
MAPPED_PATH = 169
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 129
ABSOLUTE_PATH = 129
MAPPED_PATH = 185
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 130
ABSOLUTE_PATH = 130
MAPPED_PATH = 201
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 131
ABSOLUTE_PATH = 131
MAPPED_PATH = 217
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 132
ABSOLUTE_PATH = 132
MAPPED_PATH = 233
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 133
ABSOLUTE_PATH = 133
MAPPED_PATH = 16
```

```
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 134
  ABSOLUTE_PATH = 134
  MAPPED_PATH = 32
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 135
  ABSOLUTE_PATH = 135
  MAPPED_PATH = 48
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 136
  ABSOLUTE_PATH = 136
  MAPPED_PATH = 64
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 137
  ABSOLUTE_PATH = 137
  MAPPED_PATH = 80
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 138
  ABSOLUTE_PATH = 138
  MAPPED_PATH = 96
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 139
  ABSOLUTE_PATH = 139
  MAPPED_PATH = 112
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 140
  ABSOLUTE_PATH = 140
  MAPPED_PATH = 128
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 141
  ABSOLUTE_PATH = 141
  MAPPED_PATH = 144
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 142
  ABSOLUTE_PATH = 142
  MAPPED_PATH = 160
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 143
  ABSOLUTE_PATH = 143
  MAPPED_PATH = 176
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 144
  ABSOLUTE_PATH = 144
  MAPPED_PATH = 192
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 145
  ABSOLUTE_PATH = 145
  MAPPED_PATH = 208
END_OBJECT = PATHMAP_ENTRY
```

```
OBJECT = PATHMAP_ENTRY
  CLASS = 146
  ABSOLUTE_PATH = 146
  MAPPED_PATH = 224
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 147
  ABSOLUTE_PATH = 147
  MAPPED_PATH = 7
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 148
  ABSOLUTE_PATH = 148
  MAPPED_PATH = 23
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 149
  ABSOLUTE_PATH = 149
  MAPPED_PATH = 39
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 150
  ABSOLUTE_PATH = 150
  MAPPED_PATH = 55
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 151
  ABSOLUTE_PATH = 151
  MAPPED_PATH = 71
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 152
  ABSOLUTE_PATH = 152
  MAPPED_PATH = 87
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 153
  ABSOLUTE_PATH = 153
  MAPPED_PATH = 103
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 154
  ABSOLUTE_PATH = 154
  MAPPED_PATH = 119
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 155
  ABSOLUTE_PATH = 155
  MAPPED_PATH = 135
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 156
  ABSOLUTE_PATH = 156
  MAPPED_PATH = 151
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 157
  ABSOLUTE_PATH = 157
  MAPPED_PATH = 167
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
```

```
CLASS = 158
  ABSOLUTE_PATH = 158
  MAPPED_PATH = 183
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 159
  ABSOLUTE_PATH = 159
  MAPPED_PATH = 199
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 160
  ABSOLUTE_PATH = 160
  MAPPED_PATH = 215
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 161
  ABSOLUTE_PATH = 161
  MAPPED_PATH = 231
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 162
  ABSOLUTE_PATH = 162
  MAPPED_PATH = 14
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 163
  ABSOLUTE_PATH = 163
  MAPPED_PATH = 30
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 164
  ABSOLUTE_PATH = 164
  MAPPED_PATH = 46
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 165
  ABSOLUTE_PATH = 165
  MAPPED_PATH = 62
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 166
  ABSOLUTE_PATH = 166
  MAPPED_PATH = 78
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 167
  ABSOLUTE_PATH = 167
  MAPPED_PATH = 94
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 168
  ABSOLUTE_PATH = 168
  MAPPED_PATH = 110
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 169
  ABSOLUTE_PATH = 169
  MAPPED_PATH = 126
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 170
```

```
ABSOLUTE_PATH = 170
MAPPED_PATH = 142
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 171
ABSOLUTE_PATH = 171
MAPPED_PATH = 158
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 172
ABSOLUTE_PATH = 172
MAPPED_PATH = 174
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 173
ABSOLUTE_PATH = 173
MAPPED_PATH = 190
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 174
ABSOLUTE_PATH = 174
MAPPED_PATH = 206
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 175
ABSOLUTE_PATH = 175
MAPPED_PATH = 222
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 176
ABSOLUTE_PATH = 176
MAPPED_PATH = 5
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 177
ABSOLUTE_PATH = 177
MAPPED_PATH = 21
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 178
ABSOLUTE_PATH = 178
MAPPED_PATH = 37
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 179
ABSOLUTE_PATH = 179
MAPPED_PATH = 53
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 180
ABSOLUTE_PATH = 180
MAPPED_PATH = 69
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 181
ABSOLUTE_PATH = 181
MAPPED_PATH = 85
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 182
ABSOLUTE_PATH = 182
```

```
MAPPED_PATH = 101
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 183
ABSOLUTE_PATH = 183
MAPPED_PATH = 117
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 184
ABSOLUTE_PATH = 184
MAPPED_PATH = 133
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 185
ABSOLUTE_PATH = 185
MAPPED_PATH = 149
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 186
ABSOLUTE_PATH = 186
MAPPED_PATH = 165
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 187
ABSOLUTE_PATH = 187
MAPPED_PATH = 181
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 188
ABSOLUTE_PATH = 188
MAPPED_PATH = 197
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 189
ABSOLUTE_PATH = 189
MAPPED_PATH = 213
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 190
ABSOLUTE_PATH = 190
MAPPED_PATH = 229
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 191
ABSOLUTE_PATH = 191
MAPPED_PATH = 12
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 192
ABSOLUTE_PATH = 192
MAPPED_PATH = 28
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 193
ABSOLUTE_PATH = 193
MAPPED_PATH = 44
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 194
ABSOLUTE_PATH = 194
MAPPED_PATH = 60
```

```
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 195
  ABSOLUTE_PATH = 195
  MAPPED_PATH = 76
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 196
  ABSOLUTE_PATH = 196
  MAPPED_PATH = 92
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 197
  ABSOLUTE_PATH = 197
  MAPPED_PATH = 108
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 198
  ABSOLUTE_PATH = 198
  MAPPED_PATH = 124
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 199
  ABSOLUTE_PATH = 199
  MAPPED_PATH = 140
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 200
  ABSOLUTE_PATH = 200
  MAPPED_PATH = 156
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 201
  ABSOLUTE_PATH = 201
  MAPPED_PATH = 172
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 202
  ABSOLUTE_PATH = 202
  MAPPED_PATH = 188
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 203
  ABSOLUTE_PATH = 203
  MAPPED_PATH = 204
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 204
  ABSOLUTE_PATH = 204
  MAPPED_PATH = 220
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 205
  ABSOLUTE_PATH = 205
  MAPPED_PATH = 3
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 206
  ABSOLUTE_PATH = 206
  MAPPED_PATH = 19
END_OBJECT = PATHMAP_ENTRY
```

```
OBJECT = PATHMAP_ENTRY
  CLASS = 207
  ABSOLUTE_PATH = 207
  MAPPED_PATH = 35
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 208
  ABSOLUTE_PATH = 208
  MAPPED_PATH = 51
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 209
  ABSOLUTE_PATH = 209
  MAPPED_PATH = 67
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 210
  ABSOLUTE_PATH = 210
  MAPPED_PATH = 83
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 211
  ABSOLUTE_PATH = 211
  MAPPED_PATH = 99
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 212
  ABSOLUTE_PATH = 212
  MAPPED_PATH = 115
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 213
  ABSOLUTE_PATH = 213
  MAPPED_PATH = 131
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 214
  ABSOLUTE_PATH = 214
  MAPPED_PATH = 147
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 215
  ABSOLUTE_PATH = 215
  MAPPED_PATH = 163
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 216
  ABSOLUTE_PATH = 216
  MAPPED_PATH = 179
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 217
  ABSOLUTE_PATH = 217
  MAPPED_PATH = 195
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
  CLASS = 218
  ABSOLUTE_PATH = 218
  MAPPED_PATH = 211
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
```

```
CLASS = 219
ABSOLUTE_PATH = 219
MAPPED_PATH = 227
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 220
ABSOLUTE_PATH = 220
MAPPED_PATH = 10
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 221
ABSOLUTE_PATH = 221
MAPPED_PATH = 26
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 222
ABSOLUTE_PATH = 222
MAPPED_PATH = 42
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 223
ABSOLUTE_PATH = 223
MAPPED_PATH = 58
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 224
ABSOLUTE_PATH = 224
MAPPED_PATH = 74
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 225
ABSOLUTE_PATH = 225
MAPPED_PATH = 90
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 226
ABSOLUTE_PATH = 226
MAPPED_PATH = 106
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 227
ABSOLUTE_PATH = 227
MAPPED_PATH = 122
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 228
ABSOLUTE_PATH = 228
MAPPED_PATH = 138
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 229
ABSOLUTE_PATH = 229
MAPPED_PATH = 154
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 230
ABSOLUTE_PATH = 230
MAPPED_PATH = 170
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
CLASS = 231
```

```

    ABSOLUTE_PATH = 231
    MAPPED_PATH = 186
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 232
    ABSOLUTE_PATH = 232
    MAPPED_PATH = 202
END_OBJECT = PATHMAP_ENTRY
OBJECT = PATHMAP_ENTRY
    CLASS = 233
    ABSOLUTE_PATH = 233
    MAPPED_PATH = 218
END_OBJECT = PATHMAP_ENTRY
END

```

## C.4 Typical Terra MODIS PGE & ESDT ODL Files

Listings are provided for the following MODIS ODL files:

C.4.1 MODIS PGE ODL for PGE\_NAME PGE01

C.4.2 MODIS ESDT MOD000 ODL

C.4.3 MODIS ESDT MOD01 ODL

C.4.4 MODIS ESDT MOD01LUT ODL

### C.4.5 MODIS ESDT MOD03 ODL

C.4.6 MODIS ESDT MOD03LUT ODL

C.4.7 MODIS PGE ODL for PGE\_NAME PGE03

C.4.8 GDAS\_0ZF ODL

C.4.9 OZ\_DAILY ODL

C.4.10 REYNSST ODL

C.4.11 SEA\_ICE ODL

C.4.12 NISE ODL

A typical MODIS PGE will differ from the examples here by the PGE\_NAME, the specific input/output files referenced, and runtime parameters. However, the overall structure of a given MODIS PGE ODL file would be similar to the ones used here. (N.B. The ODL files shown here are associated with the MODIS version 2.1 software)

### C.4.1 MODIS PGE PGE01 ODL

```

PGE_NAME = "PGE01"
PGE_VERSION = "2.1"
PROFILE_ID = 1
PROFILE_DESCRIPTION = "The profile for MOD_PR01 and MOD_PR03 "
PLATFORM = "AM1"
INSTRUMENT = "MODIS"
MINIMUM_OUTPUTS = 0
SCHEDULE_TYPE = "Time"
PROCESSING_PERIOD = "MINS=15"
PROCESSING_BOUNDARY = "START_OF_MIN"
PGE_SSW_VERSION = "2.1"

```

QUERY\_DELAY = 0

OBJECT = EXIT\_MESSAGE  
CLASS= 1  
EXIT\_CODE = 0  
EXIT\_MESSAGE = "PGE01 Exit"  
END\_OBJECT = EXIT\_MESSAGE  
OBJECT = EXIT\_DEPENDENCY  
CLASS= 1  
DEPENDENCY\_PGE\_NAME = "none"  
DEPENDENCY\_SSW\_VERSION = "none"  
EXIT\_OPERATION = "="  
EXIT\_CODE = 0  
END\_OBJECT = EXIT\_DEPENDENCY

OBJECT = PCF\_ENTRY  
CLASS = 10  
LOGICAL\_ID = 599001  
PCF\_FILE\_TYPE = 1  
DATA\_TYPE\_NAME = "MOD000"  
DATA\_TYPE\_VERSION = "001"  
MIN\_GRANULES\_REQUIRED = 1  
MAX\_GRANULES\_REQUIRED = 1  
BEGIN\_PERIOD\_OFFSET = -7200  
END\_PERIOD\_OFFSET = -7200  
INPUT\_TYPE = "Optional"  
ALIGN\_DPR\_TIME\_WITH\_INPUT\_TIME = "N"  
NUMBER\_NEEDED = 1  
QUERY\_TYPE = "Temporal"  
SPATIAL\_TIME\_DELTA = 0  
KEY\_INPUT = "N"  
OBJECT = FILETYPE  
FILETYPE\_NAME = "L0 Data Files"  
CLASS = 1  
END\_OBJECT = FILETYPE  
OBJECT = OPTIONAL\_INPUT  
CLASS = 1  
ORDER = 1  
RUNTIME\_PARM\_ID = 51  
TIMER = "HOURS=4"  
TEMPORAL = "N"  
END\_OBJECT = OPTIONAL\_INPUT  
END\_OBJECT = PCF\_ENTRY

OBJECT = PCF\_ENTRY  
CLASS = 11  
LOGICAL\_ID = 599002  
PCF\_FILE\_TYPE = 1  
DATA\_TYPE\_NAME = "MOD000"  
DATA\_TYPE\_VERSION = "001"  
MIN\_GRANULES\_REQUIRED = 1  
MAX\_GRANULES\_REQUIRED = 1  
BEGIN\_PERIOD\_OFFSET = 0  
END\_PERIOD\_OFFSET = 0  
INPUT\_TYPE = "Required"  
ALIGN\_DPR\_TIME\_WITH\_INPUT\_TIME = "N"  
NUMBER\_NEEDED = 1  
QUERY\_TYPE = "Temporal"  
SPATIAL\_TIME\_DELTA = 0  
KEY\_INPUT = "N"  
OBJECT = FILETYPE

```

        FILETYPE_NAME = "L0 Data Files"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 12
    LOGICAL_ID = 599003
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD01LUT"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L1"
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 15
    LOGICAL_ID = 600020
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD01LUT"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L2"
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Two GEO_parameter data files"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 17
    LOGICAL_ID = 10501
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "AM1EPHN0"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0

```

```
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
CLASS = 18
LOGICAL_ID = 10502
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "AM1ATTN0"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
BEGIN_PERIOD_OFFSET = 0
END_PERIOD_OFFSET = 0
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
CLASS = 110
LOGICAL_ID = 500100
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MOD01"
DATA_TYPE_VERSION = "001"
MIN_GRANULE_YIELD = 3
MAX_GRANULE_YIELD = 3
ASSOCIATED_MCF_ID = 500500
SCIENCE_GROUP = "S1"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
```

```
OBJECT = PCF_ENTRY
CLASS = 111
LOGICAL_ID = 600000
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "MOD03"
DATA_TYPE_VERSION = "001"
MIN_GRANULE_YIELD = 3
MAX_GRANULE_YIELD = 3
```

```

ASSOCIATED_MCF_ID = 600111
SCIENCE_GROUP = "S2"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 113
LOGICAL_ID = 503000
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Length of L1A granules in seconds"
PGE_PARAMETER_DEFAULT = "300.000000"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 114
LOGICAL_ID = 504000
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Scan rate for L1A granule"
PGE_PARAMETER_DEFAULT = "1.477"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 115
LOGICAL_ID = 505000
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "PGE version for L1A granule"
PGE_PARAMETER_DEFAULT = "2.1.1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 116
LOGICAL_ID = 800510
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "SatelliteInstrument; AM1M-Terra, PM1M-Aqua"
PGE_PARAMETER_DEFAULT = "AM1M"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 117
LOGICAL_ID = 800500
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "PGE01 Version"
PGE_PARAMETER_DEFAULT = "2.1.1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 118
LOGICAL_ID = 600280
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Source for spacecraft kinematic state"

```

```

    PGE_PARAMETER_DEFAULT = "SDP Toolkit"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 119
    LOGICAL_ID = 600310
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Terrain Correction Flag"
    PGE_PARAMETER_DEFAULT = "TRUE"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 120
    LOGICAL_ID = 600001
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "LOCALVERSIONID"
    PGE_PARAMETER_DEFAULT = "2.1.2"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY

END

```

#### **C.4.2 MODIS ESDT MOD000 ODL**

```

DATA_TYPE_NAME = "MOD000"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MODIS"
PLATFORM = "EOSAM1"
DATA_TYPE_DESCRIPTION = "L0 Input of PGE MOD_PR01"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = 569.0
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "GSFC"
PERIOD = "HOURS=2"
BOUNDARY = "START_OF_DAY"
DELAY = 43200
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
OBJECT = FILETYPE
    CLASS = 1
    FILETYPE_NAME = "L0 Data Files"
    MAXIMUM_NUM_FILES = 6
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
END

```

### C.4.3 MODIS ESDT MOD01 ODL

```
DATA_TYPE_NAME = "MOD01"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MODIS"  
PLATFORM = "EOSAM1"  
DATA_TYPE_DESCRIPTION = "An Input of PGE MOD_PR02"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 569.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "Y"  
DYNAMIC_FLAG = "I"  
PREDICTION_METHOD = "ROUTINE"  
SUPPLIER_NAME = "GSFC"  
PERIOD = "MINS=5"  
BOUNDARY = "START_OF_MIN"  
DELAY = 43200  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "GSFC"  
END_OBJECT = USE_OBJECT  
ARCHIVED_AT = "GSFC"  
PROCESSED_AT = "GSFC"  
END
```

### C.4.4 MODIS ESDT MOD01LUT ODL

```
DATA_TYPE_NAME = "MOD01LUT"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MODIS"  
PLATFORM = "EOSAM1"  
DATA_TYPE_DESCRIPTION = "An Input (static) of PGE MOD01"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 0.357  
PROCESSING_LEVEL = "L1"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "GSFC"  
END_OBJECT = USE_OBJECT  
END
```

### C.4.5 MODIS ESDT MOD03 ODL

```
DATA_TYPE_NAME = "MOD03"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "MODIS"  
PLATFORM = "EOSAM1"  
DATA_TYPE_DESCRIPTION = "Input/Output of PGE MOD_PR29/MOD_PR03"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 58.0  
PROCESSING_LEVEL = "Geo"  
HDF_DATA = "Y"  
DYNAMIC_FLAG = "I"  
PREDICTION_METHOD = "ROUTINE"
```

```

SUPPLIER_NAME = "GSFC"
PERIOD = "MINS=5"
BOUNDARY = "START_OF_MIN"
DELAY = 43200
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
END

```

#### C.4.6 MODIS ESDT MOD03LUT ODL

```

DATA_TYPE_NAME = "MOD03LUT"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MODIS"
PLATFORM = "EOSAM1"
DATA_TYPE_DESCRIPTION = "An Input (static) of PGE MOD_PR03"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = 0.357
PROCESSING_LEVEL = "L1"
HDF_DATA = "N"
DYNAMIC_FLAG = "S"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
OBJECT = FILETYPE
  CLASS = 1
  FILETYPE_NAME = "Two GEO_parameter data files"
  MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE
END

```

#### C.4.7 MODIS PGE PGE03 ODL

```

PGE_NAME = "TerraPGE03"
PGE_VERSION = "3.0.0"
PROFILE_ID = 1
PROFILE_DESCRIPTION = "First Step in Level 2 Processing"
PGE_DEFAULT_PROFILE = "N"
PLATFORM = "AM1"
INSTRUMENT = "MODIS"
MINIMUM_OUTPUTS = 0
SCHEDULE_TYPE = "Time"
PROCESSING_PERIOD = "MINS=5"
PROCESSING_BOUNDARY = "START_OF_MIN"
PGE_SSW_VERSION = "3.0.0"
QUERY_DELAY = 0
OBJECT = EXIT_MESSAGE
  CLASS= 1
  EXIT_CODE = 0
  EXIT_MESSAGE = "none"
END_OBJECT = EXIT_MESSAGE
OBJECT = EXIT_DEPENDENCY
  CLASS= 1
  DEPENDENCY_PGE_NAME = "none"
  DEPENDENCY_SSW_VERSION = "none"

```

```

EXIT_OPERATION = "="
EXIT_CODE = 0
END_OBJECT = EXIT_DEPENDENCY
OBJECT = PCF_ENTRY
CLASS = 11
LOGICAL_ID = 600000
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MOD03"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
BEGIN_PERIOD_OFFSET = 0
END_PERIOD_OFFSET = 0
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 12
LOGICAL_ID = 700000
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MOD02QKM"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
BEGIN_PERIOD_OFFSET = 0
END_PERIOD_OFFSET = 0
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 13
LOGICAL_ID = 700002
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "MOD021KM"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
BEGIN_PERIOD_OFFSET = 0
END_PERIOD_OFFSET = 0
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"

```

```

OBJECT = FILETYPE
  FILETYPE_NAME = "Single File Granule"
  CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 21
  LOGICAL_ID = 900000
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "GDAS_0ZF"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  BEGIN_PERIOD_OFFSET = -10650
  END_PERIOD_OFFSET = 10650
  INPUT_TYPE = "Required"
  ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 22
  LOGICAL_ID = 900020
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "OZ_DAILY"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  BEGIN_PERIOD_OFFSET = -43200
  END_PERIOD_OFFSET = 43200
  INPUT_TYPE = "Required"
  ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  SPATIAL_TIME_DELTA = 0
  KEY_INPUT = "N"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 23
  LOGICAL_ID = 900030
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "REYNSST"
  DATA_TYPE_VERSION = "001"
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  BEGIN_PERIOD_OFFSET = 0
  END_PERIOD_OFFSET = 0
  INPUT_TYPE = "Required"
  ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"

```

```

SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 24
    LOGICAL_ID = 900040
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "SEA_ICE"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = -43200
    END_PERIOD_OFFSET = 43200
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 25
    LOGICAL_ID = 900100
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "NISE"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 31
    LOGICAL_ID = 420011
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD07LUT"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L1"
    INPUT_TYPE = "Required"

```

```

ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 32
    LOGICAL_ID = 420012
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD07LUT"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L2"
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 33
    LOGICAL_ID = 422501
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD35ANC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L1"
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 34
    LOGICAL_ID = 900600
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD35ANC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1

```

```

MAX_GRANULES_REQUIRED = 1
BEGIN_PERIOD_OFFSET = 0
END_PERIOD_OFFSET = 0
SCIENCE_GROUP = "L2"
INPUT_TYPE = "Required"
ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
SPATIAL_TIME_DELTA = 0
KEY_INPUT = "N"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 35
    LOGICAL_ID = 900601
    PCF_FILE_TYPE = 1
    DATA_TYPE_NAME = "MOD35ANC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    BEGIN_PERIOD_OFFSET = 0
    END_PERIOD_OFFSET = 0
    SCIENCE_GROUP = "L3"
    INPUT_TYPE = "Required"
    ALIGN_DPR_TIME_WITH_INPUT_TIME = "N"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    SPATIAL_TIME_DELTA = 0
    KEY_INPUT = "N"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 122
    LOGICAL_ID = 402500
    PCF_FILE_TYPE = 2
    DATA_TYPE_NAME = "MODVOLC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULE_YIELD = 1
    MAX_GRANULE_YIELD = 1
    ASSOCIATED_MCF_ID = 402503
    SCIENCE_GROUP = "S1"
    INSTANCE = 0
    MINIMUM_SIZE = 0
    MAXIMUM_SIZE = 0
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 123
    LOGICAL_ID = 420000
    PCF_FILE_TYPE = 2
    DATA_TYPE_NAME = "MOD07_L2"
    DATA_TYPE_VERSION = "001"

```

```

MIN_GRANULE_YIELD = 1
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 420001
SCIENCE_GROUP = "S2"
INSTANCE = 0
MINIMUM_SIZE = 0
MAXIMUM_SIZE = 0
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 124
    LOGICAL_ID = 420002
    PCF_FILE_TYPE = 2
    DATA_TYPE_NAME = "MOD07_QC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULE_YIELD = 0
    MAX_GRANULE_YIELD = 1
    ASSOCIATED_MCF_ID = 420003
    SCIENCE_GROUP = "S3"
    INSTANCE = 0
    MINIMUM_SIZE = 0
    MAXIMUM_SIZE = 0
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 125
    LOGICAL_ID = 422500
    PCF_FILE_TYPE = 2
    DATA_TYPE_NAME = "MOD35_L2"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULE_YIELD = 1
    MAX_GRANULE_YIELD = 1
    ASSOCIATED_MCF_ID = 422506
    SCIENCE_GROUP = "S4"
    INSTANCE = 0
    MINIMUM_SIZE = 0
    MAXIMUM_SIZE = 0
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 126
    LOGICAL_ID = 422551
    PCF_FILE_TYPE = 2
    DATA_TYPE_NAME = "MOD35_QC"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULE_YIELD = 1
    MAX_GRANULE_YIELD = 1
    ASSOCIATED_MCF_ID = 422507
    SCIENCE_GROUP = "S5"
    INSTANCE = 0
    MINIMUM_SIZE = 0
    MAXIMUM_SIZE = 0

```

```

OBJECT = FILETYPE
  FILETYPE_NAME = "Single File Granule"
  CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 127
  LOGICAL_ID = 422552
  PCF_FILE_TYPE = 2
  DATA_TYPE_NAME = "MODCSR_G"
  DATA_TYPE_VERSION = "001"
  MIN_Granule_YIELD = 1
  MAX_Granule_YIELD = 1
  ASSOCIATED_MCF_ID = 422510
  SCIENCE_GROUP = "S6"
  INSTANCE = 0
  MINIMUM_SIZE = 0
  MAXIMUM_SIZE = 0
OBJECT = FILETYPE
  FILETYPE_NAME = "Single File Granule"
  CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 129
  LOGICAL_ID = 800510
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "SatelliteInstrument"
  PGE_PARAMETER_DEFAULT = "AM1M"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 130
  LOGICAL_ID = 402502
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "RP Reference to VOLCALERT"
  PGE_PARAMETER_DEFAULT = "402500:1"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 131
  LOGICAL_ID = 420004
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "MOD_PR07.qc"
  PGE_PARAMETER_DEFAULT = "420002:1"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 132
  LOGICAL_ID = 421000
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Number_Of_Invent_RP"
  PGE_PARAMETER_DEFAULT = "4"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 133

```

```

LOGICAL_ID = 421001
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_1 "
PGE_PARAMETER_DEFAULT = "REPROCESSINGACTUAL"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 134
LOGICAL_ID = 421002
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_1"
PGE_PARAMETER_DEFAULT = "processed once"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 135
LOGICAL_ID = 421003
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_2 "
PGE_PARAMETER_DEFAULT = "REPROCESSINGPLANNED"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 136
LOGICAL_ID = 421004
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_2"
PGE_PARAMETER_DEFAULT = "further update is anticipated"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 137
LOGICAL_ID = 421005
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_3 "
PGE_PARAMETER_DEFAULT = "LOCALVERSIONID"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 138
LOGICAL_ID = 421006
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_3"
PGE_PARAMETER_DEFAULT = "002"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 139
LOGICAL_ID = 421007
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_4 "
PGE_PARAMETER_DEFAULT = "PGEVERSION"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY

```

```

OBJECT = PCF_ENTRY
  CLASS = 140
  LOGICAL_ID = 421008
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Inventory_RP_Value_4"
  PGE_PARAMETER_DEFAULT = "3.0.0"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 141
  LOGICAL_ID = 421100
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Number_Of_Archive_RP"
  PGE_PARAMETER_DEFAULT = "8"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 142
  LOGICAL_ID = 421101
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Name_1 "
  PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEACCEPTANCEDATE"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 143
  LOGICAL_ID = 421102
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Value_1"
  PGE_PARAMETER_DEFAULT = "June 1997"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 144
  LOGICAL_ID = 421103
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Name_2 "
  PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEMATURITYCODE"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 145
  LOGICAL_ID = 421104
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Value_2"
  PGE_PARAMETER_DEFAULT = "at-launch"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 146
  LOGICAL_ID = 421105
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Name_3 "
  PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGENAME"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"

```

```

    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 147
    LOGICAL_ID = 421106
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive_RP_Value_3"
    PGE_PARAMETER_DEFAULT = "ATBD-MOD-07"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 148
    LOGICAL_ID = 421107
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive RP_Name_4 "
    PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEVERSION"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 149
    LOGICAL_ID = 421108
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive_RP_Value_4"
    PGE_PARAMETER_DEFAULT = "2"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 150
    LOGICAL_ID = 421109
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive RP_Name_5 "
    PGE_PARAMETER_DEFAULT = "INSTRUMENTNAME"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 151
    LOGICAL_ID = 421110
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive_RP_Value_5"
    PGE_PARAMETER_DEFAULT = "Moderate Resolution Imaging Spectroradiometer"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 152
    LOGICAL_ID = 421111
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive RP_Name_6 "
    PGE_PARAMETER_DEFAULT = "Profiles_Algorithm_Version_Number"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 153
    LOGICAL_ID = 421112
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Archive_RP_Value_6"

```

```

PGE_PARAMETER_DEFAULT = "1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 154
LOGICAL_ID = 421113
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive RP_Name_7 "
PGE_PARAMETER_DEFAULT = "Total_Ozone_Algorithm_Version_Number"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 155
LOGICAL_ID = 421114
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Value_7"
PGE_PARAMETER_DEFAULT = "1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 156
LOGICAL_ID = 421115
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive RP_Name_8 "
PGE_PARAMETER_DEFAULT = "Stability_Indices_Algorithm_Version_Number"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 157
LOGICAL_ID = 421116
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Value_8"
PGE_PARAMETER_DEFAULT = "1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 158
LOGICAL_ID = 422508
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "MOD35_QC.qc"
PGE_PARAMETER_DEFAULT = "422551:1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 159
LOGICAL_ID = 424000
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "MOD35_Num_InvMet_RP_Pairs"
PGE_PARAMETER_DEFAULT = "4"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 160
LOGICAL_ID = 424001

```

```

PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_1 "
PGE_PARAMETER_DEFAULT = "REPROCESSINGACTUAL"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 161
LOGICAL_ID = 424002
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_1"
PGE_PARAMETER_DEFAULT = "processed once"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 162
LOGICAL_ID = 424003
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_2 "
PGE_PARAMETER_DEFAULT = "REPROCESSINGPLANNED"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 163
LOGICAL_ID = 424004
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_2"
PGE_PARAMETER_DEFAULT = "further update is anticipated"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 164
LOGICAL_ID = 424005
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_3 "
PGE_PARAMETER_DEFAULT = "LOCALVERSIONID"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 165
LOGICAL_ID = 424006
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_3"
PGE_PARAMETER_DEFAULT = "002"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 166
LOGICAL_ID = 424007
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Name_4 "
PGE_PARAMETER_DEFAULT = "PGEVERSION"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY

```

```

CLASS = 167
LOGICAL_ID = 424008
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Inventory_RP_Value_4"
PGE_PARAMETER_DEFAULT = "2.6.1"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 168
LOGICAL_ID = 424100
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "MOD35_Num_ArchMet_RP_Pairs"
PGE_PARAMETER_DEFAULT = "5"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 169
LOGICAL_ID = 424101
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Name_1 "
PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEACCEPTANCEDATE"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 170
LOGICAL_ID = 424102
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Value_1"
PGE_PARAMETER_DEFAULT = "June 1997"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 171
LOGICAL_ID = 424103
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Name_2 "
PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEMATURITYCODE"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 172
LOGICAL_ID = 424104
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Value_2"
PGE_PARAMETER_DEFAULT = "at-launch"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 173
LOGICAL_ID = 424105
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Archive_RP_Name_3 "
PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGENAME"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
PROFILE_SELECTOR_PGE_PARAMETER = "N"

```

```

END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 174
  LOGICAL_ID = 424106
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Value_3"
  PGE_PARAMETER_DEFAULT = "ATBD-MOD-06"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 175
  LOGICAL_ID = 424107
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive RP_Name_4 "
  PGE_PARAMETER_DEFAULT = "ALGORITHMPACKAGEVERSION"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 176
  LOGICAL_ID = 424108
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Value_4"
  PGE_PARAMETER_DEFAULT = "2"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 177
  LOGICAL_ID = 424109
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive RP_Name_5 "
  PGE_PARAMETER_DEFAULT = "INSTRUMENTNAME"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 178
  LOGICAL_ID = 424110
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Archive_RP_Value_5"
  PGE_PARAMETER_DEFAULT = "Moderate Resolution Imaging Spectroradiometer"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 179
  LOGICAL_ID = 424300
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "UW DEBUG; 0 to 4, no output to reams"
  PGE_PARAMETER_DEFAULT = "0"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
  PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 180
  LOGICAL_ID = 424301
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Processing Range Begin Line"
  PGE_PARAMETER_DEFAULT = "0"

```

```

    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 181
    LOGICAL_ID = 424302
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Processing Range Number of Lines"
    PGE_PARAMETER_DEFAULT = "0"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 182
    LOGICAL_ID = 424303
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Processing Range Begin Element"
    PGE_PARAMETER_DEFAULT = "0"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 183
    LOGICAL_ID = 424304
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Processing Range Number of Elements"
    PGE_PARAMETER_DEFAULT = "0"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
    PROFILE_SELECTOR_PGE_PARAMETER = "N"
END_OBJECT = PCF_ENTRY
END

```

#### **C.4.8 GDAS\_0ZF ODL**

```

DATA_TYPE_NAME = "GDAS_0ZF"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "MODIS"
PLATFORM = "EOSAM1"
DATA_TYPE_DESCRIPTION = "NCEP 6-Hour Atmospheric Profile"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = 21.0
PROCESSING_LEVEL = "L1"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "NCEP"
PERIOD = "HOURS=6"
BOUNDARY = "START_OF_6HOUR"
DURATION = "SECS=1"
DELAY = 10
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
    CLASS = 1
    USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
END

```

#### **C.4.9 OZ\_DAILY ODL**

The same as GDAS\_0ZF ODL except the following items:

```
DATA_TYPE_NAME = "OZ_DAILY"  
DATA_TYPE_DESCRIPTION = "TOVS Column Ozone Daily Product"  
NOMINAL_SIZE = 0.10  
PERIOD = "DAYS=1"  
BOUNDARY = "START_OF_DAY+43200"  
DURATION = "SECS=1"
```

#### **C.4.10 REYNSST ODL**

The same as GDAS\_0ZF ODL except the following items:

```
DATA_TYPE_NAME = "REYNSST"  
DATA_TYPE_DESCRIPTION = "Reynolds Weekly SST"  
NOMINAL_SIZE = 0.30  
PERIOD = "SECS=604800"  
BOUNDARY = "START_OF_WEEK-86400"  
DURATION = "SECS=604800"
```

#### **C.4.11 SEA\_ICE ODL**

The same as GDAS\_0ZF ODL except the following items:

```
DATA_TYPE_NAME = "SEA_ICE"  
DATA_TYPE_DESCRIPTION = "NCEP Ice Concentration"  
NOMINAL_SIZE = 0.30  
PERIOD = "SECS=86400"  
BOUNDARY = "START_OF_DAY"  
DURATION = "SECS=1"
```

#### **C.4.12 NISE ODL**

The same as GDAS\_0ZF ODL except the following items:

```
DATA_TYPE_NAME = "NISE"  
DATA_TYPE_DESCRIPTION = "NSIDC NISE snow/ice extent"  
NOMINAL_SIZE = 0.03  
PERIOD = "DAYS=1"  
BOUNDARY = "START_OF_DAY"  
DURATION = "DAYS=1"
```

## C.5 Typical AIRS PGE & ESDT ODL Files

Listings are provided for the following AIRS ODL files:

- C.5.1 AIRS PGE ODL for PGE\_NAME AiL1A\_AMSU
- C.5.2 AIRS ESDT AIR10SCI ODL
- C.5.3 AIRS ESDT AIR10SCC ODL
- C.5.4 AIRS ESDT AIR20SCI ODL
- C.5.5 AIRS ESDT PMCO\_HK ODL
- C.5.6 AIRS ESDT PM1EPHND ODL
- C.5.7 AIRS ESDT PM1ATTNR ODL
- C.5.8 AIRS ESDT AIRAASCI ODL
- C.5.9 AIRS ESDT AIRXADCM ODL
- C.5.10 AIRS ESDT AIRXATCM ODL
- C.5.11 AIRS ESDT AIRXATCS ODL
- C.5.12 AIRS ESDT AIRXARYL ODL
- C.5.13 AIRS ESDT AIRXAGEO ODL

A typical AIRS PGE will differ from the examples here by the PGE\_NAME, the specific input/output files referenced, and runtime parameters. However, the overall structure of a given AIRS PGE ODL file would be similar to the ones used here. (N.B. The ODL files shown here are associated with the AIRS version 2.1.2 software)

### C.5.1 AIRS PGE AiL1A\_AMSU ODL

```
PGE_NAME = "L1A_AMSU"
PGE_VERSION = "212"
PROFILE_ID = 1
PROFILE_DESCRIPTION = "GRAN01"
PLATFORM = "EOSPM1"
INSTRUMENT = "AIRS"
MINIMUM_OUTPUTS = 0
SCHEDULE_TYPE = "Time"
PROCESSING_PERIOD = "MINS=6"
PROCESSING_BOUNDARY = "START_OF_DAY-31"
PGE_SSW_VERSION = "212"

/***** Primary Inputs *****/
OBJECT = PCF_ENTRY
  CLASS = 11
  LOGICAL_ID = 261
  PCF_FILE_TYPE = 1
  DATA_TYPE_NAME = "AIR10SCC"
  DATA_TYPE_VERSION = "001"
  BEGIN_PERIOD_OFFSET = 31
  END_PERIOD_OFFSET = 31
  MIN_GRANULES_REQUIRED = 1
  MAX_GRANULES_REQUIRED = 1
  INPUT_TYPE = "Required"
/*  ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y" */
  NUMBER_NEEDED = 1
  QUERY_TYPE = "Temporal"
  OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
  END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
```

```

CLASS = 12
LOGICAL_ID = 262
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "AIR10SCI"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
/* ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y" */
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "L0 Data Files"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 13
LOGICAL_ID = 290
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "AIR20SCI"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
/* ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y" */
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "L0 Data Files"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*****Dynamic ancillary inputs *****/
OBJECT = PCF_ENTRY
CLASS = 14
LOGICAL_ID = 4007
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "PMCO_HK"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
/* ALIGN_DPR_TIME_WITH_INPUT_TIME = "Y" */
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY

```

```

CLASS = 14
LOGICAL_ID = 4008
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "PMCO_HK"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
/* ALIGN DPR_TIME_WITH_INPUT_TIME = "Y" */
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/**** Attitude/Ephemeris/DEM entry. Please delete if not used by PGE. ****/
OBJECT = PCF_ENTRY
CLASS = 18
LOGICAL_ID = 10501
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "PM1EPHND"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 19
LOGICAL_ID = 10502
PCF_FILE_TYPE = 1
DATA_TYPE_NAME = "PM1ATTNR"
DATA_TYPE_VERSION = "001"
BEGIN_PERIOD_OFFSET = 31
END_PERIOD_OFFSET = 31
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*****Primary output *****/
OBJECT = PCF_ENTRY
CLASS = 110

```

```

LOGICAL_ID = 7120
PCF_FILE_TYPE = 2
DATA_TYPE_NAME = "AIRAASCI"
DATA_TYPE_VERSION = "001"
MIN_GRANULE_YIELD = 1
MAX_GRANULE_YIELD = 1
ASSOCIATED_MCF_ID = 17120
SCIENCE_GROUP = "S1"
MINIMUM_SIZE = 1
MAXIMUM_SIZE = 100
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

/*****Static ancillary inputs *****/
OBJECT = PCF_ENTRY
    CLASS = 116
    LOGICAL_ID = 4001
    PCF_FILE_TYPE = 3
    DATA_TYPE_NAME = "AIRXADCM"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    SCIENCE_GROUP = "O001"
    INPUT_TYPE = "Required"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 117
    LOGICAL_ID = 4002
    PCF_FILE_TYPE = 3
    DATA_TYPE_NAME = "AIRXATCM"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1
    SCIENCE_GROUP = "O002"
    INPUT_TYPE = "Required"
    NUMBER_NEEDED = 1
    QUERY_TYPE = "Temporal"
    OBJECT = FILETYPE
        FILETYPE_NAME = "Single File Granule"
        CLASS = 1
    END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
    CLASS = 118
    LOGICAL_ID = 4003
    PCF_FILE_TYPE = 3
    DATA_TYPE_NAME = "AIRXATCS"
    DATA_TYPE_VERSION = "001"
    MIN_GRANULES_REQUIRED = 1
    MAX_GRANULES_REQUIRED = 1

```

```

SCIENCE_GROUP = "0003"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 119
LOGICAL_ID = 4005
PCF_FILE_TYPE = 3
DATA_TYPE_NAME = "AIRXARYL"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "0004"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 120
LOGICAL_ID = 4006
PCF_FILE_TYPE = 3
DATA_TYPE_NAME = "AIRXAGEO"
DATA_TYPE_VERSION = "001"
MIN_GRANULES_REQUIRED = 1
MAX_GRANULES_REQUIRED = 1
SCIENCE_GROUP = "0005"
INPUT_TYPE = "Required"
NUMBER_NEEDED = 1
QUERY_TYPE = "Temporal"
OBJECT = FILETYPE
    FILETYPE_NAME = "Single File Granule"
    CLASS = 1
END_OBJECT = FILETYPE
END_OBJECT = PCF_ENTRY

OBJECT = PCF_ENTRY
CLASS = 125
LOGICAL_ID = 1001
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Instrument: 0=AMSU, 1=AIRS, 2=HSB(MHS), 3=VIS"
PGE_PARAMETER_DEFAULT = "0"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
CLASS = 225
LOGICAL_ID = 1002
PCF_FILE_TYPE = 5
PGE_PARAMETER_NAME = "Print Level IO: 0=Off, 1=Low, 2=Med, 3=High"
PGE_PARAMETER_DEFAULT = "2"
PGE_PARAMETER_DYNAMIC_VALUE = "NONE"

```

```

END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 226
  LOGICAL_ID = 1003
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Print Level: 0=Off, 1=Low, 2=Med, 3=High"
  PGE_PARAMETER_DEFAULT = "1"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 227
  LOGICAL_ID = 1004
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Exec Development Mode: 0=Off, 1=On"
  PGE_PARAMETER_DEFAULT = "0"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 228
  LOGICAL_ID = 1005
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Run Level 2 Mode: 1=MIT & 2=NOAA & 4=GSFC"
  PGE_PARAMETER_DEFAULT = "7"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 126
  LOGICAL_ID = 1006
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Stats Mode: 0=Off, 1=cmp2truth, 2=cmp2MW-retrieval"
  PGE_PARAMETER_DEFAULT = "0"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 128
  LOGICAL_ID = 1011
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Year (ex: 1998)"
  PGE_PARAMETER_DEFAULT = "1998"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 129
  LOGICAL_ID = 1012
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Month number (1 - 12)"
  PGE_PARAMETER_DEFAULT = "09"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 130
  LOGICAL_ID = 1013
  PCF_FILE_TYPE = 5
  PGE_PARAMETER_NAME = "Day of month (1 - 31)"
  PGE_PARAMETER_DEFAULT = "13"
  PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
  CLASS = 131
  LOGICAL_ID = 1014
  PCF_FILE_TYPE = 5

```

```

    PGE_PARAMETER_NAME = "Orbit of day (1 - 17)"
    PGE_PARAMETER_DEFAULT = "1"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 132
    LOGICAL_ID = 1015
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Granule Number (1 - 17)"
    PGE_PARAMETER_DEFAULT = "01"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 133
    LOGICAL_ID = 1016
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Granule Size in scansets (1 - 45)"
    PGE_PARAMETER_DEFAULT = "45"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 134
    LOGICAL_ID = 1020
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Times Processed: 1 for never before reprocessed"
    PGE_PARAMETER_DEFAULT = "1"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 135
    LOGICAL_ID = 1021
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "Processing Facility: A for TLSCF or G for GDAAC"
    PGE_PARAMETER_DEFAULT = "G"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
OBJECT = PCF_ENTRY
    CLASS = 200
    LOGICAL_ID = 411
    PCF_FILE_TYPE = 5
    PGE_PARAMETER_NAME = "GDAAC Build Version String"
    PGE_PARAMETER_DEFAULT = "PGE=2.1.2, SDPTK=5.2.7.2, HDF=4.1r3,HDFEOS=2.7,
OS=6.5, COMPILER=7.2.1.3, ECS=6A.03"
    PGE_PARAMETER_DYNAMIC_VALUE = "NONE"
END_OBJECT = PCF_ENTRY
END

```

## C.5.2 AIRS ESDT AIR10SCI ODL

```

DATA_TYPE_NAME = "AIR10SCI"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "AIRS"
PLATFORM = "EOSPM1"
DATA_TYPE_DESCRIPTION = "AMSU A1 Science Data Packets"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = .02
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"

```

```

SUPPLIER_NAME = "GSFC"
PERIOD = "HOURS=2"
BOUNDARY = "START_OF_DAY"
SPATIAL_FLAG = "N"
DELAY = 43200
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
OBJECT = FILETYPE
  CLASS = 1
  FILETYPE_NAME = "L0 Data Files"
  MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE
END

```

### C.5.3 AIRS ESDT AIR10SCC ODL

```

DATA_TYPE_NAME = "AIR10SCC"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "AIRS"
PLATFORM = "EOSPM1"
DATA_TYPE_DESCRIPTION = "AMSU_A1 Science Data Packets"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = .02
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "GSFC"
PERIOD = "HOURS=2"
BOUNDARY = "START_OF_DAY"
SPATIAL_FLAG = "N"
DELAY = 43200
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
OBJECT = FILETYPE
  CLASS = 1
  FILETYPE_NAME = "L0 Data Files"
  MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE
END

```

### C.5.4 AIRS ESDT AIR20SCI ODL

```

DATA_TYPE_NAME = "AIR20SCI"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "AIRS"
PLATFORM = "EOSPM1"
DATA_TYPE_DESCRIPTION = "AMSU_A2 Science Data Packets"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = .02
PROCESSING_LEVEL = "L0"
HDF_DATA = "N"
DYNAMIC_FLAG = "E"

```

```

PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "GSFC"
PERIOD = "HOURS=2"
BOUNDARY = "START_OF_DAY"
SPATIAL_FLAG = "N"
DELAY = 43200
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
OBJECT = FILETYPE
  CLASS = 1
  FILETYPE_NAME = "L0 Data Files"
  MAXIMUM_NUM_FILES = 2
END_OBJECT = FILETYPE
END

```

### **C.5.5 AIRS ESDT PMCO\_HK ODL**

```

DATA_TYPE_NAME = "PMCO_HK"
DATA_TYPE_VERSION = "001"
DATA_TYPE_DESCRIPTION = "Aqua Carryout housekeeping file"
INSTRUMENT = "All"
PLATFORM = "Aqua"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = 2.0
PROCESSING_LEVEL = "L0"
DYNAMIC_FLAG = "E"
PREDICTION_METHOD = "ROUTINE"
SUPPLIER_NAME = "GSFC"
PERIOD = "HOURS=2"
BOUNDARY = "START_OF_DAY"
DELAY = 43200
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
END_OBJECT = USE_OBJECT
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"
HDF_DATA = "N"
END

```

### **C.5.6 AIRS ESDT PM1EPHND ODL**

```

DATA_TYPE_NAME = "PM1EPHND"
DATA_TYPE_VERSION = "001"
INSTRUMENT = "All"
PLATFORM = "PM1"
DATA_TYPE_DESCRIPTION = "PM-1 FDD Definitive Ephemeris data in Toolkit format"
PROVIDER = "Goddard Space Flight Center"
NOMINAL_SIZE = 6.0
PROCESSING_LEVEL = "L1"
DYNAMIC_FLAG = "I"
SPATIAL_FLAG = "N"
OBJECT = USE_OBJECT
  CLASS = 1
  USED_BY = "GSFC"
ARCHIVED_AT = "GSFC"
PROCESSED_AT = "GSFC"

```

```
HDF_DATA = "N"  
END
```

### **C.5.7 AIRS ESDT PM1ATTNR ODL**

```
DATA_TYPE_NAME = "PM1ATTNR"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "All"  
PLATFORM = "PM1"  
DATA_TYPE_DESCRIPTION = "PM-1 Refined Attitude Data in Toolkit format"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1"  
DYNAMIC_FLAG = "I"  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "GSFC"  
END_OBJECT = USE_OBJECT  
ARCHIVED_AT = "GSFC"  
PROCESSED_AT = "GSFC"  
HDF_DATA = "N"  
END
```

### **C.5.8 AIRS ESDT AIRAASCI ODL**

```
DATA_TYPE_NAME = "AIRAASCI"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "AMSU-A geolocated science counts"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 3.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
PERIOD = "SECS=360"  
BOUNDARY = "START_OF_SEC"  
DYNAMIC_FLAG = "I"  
SPATIAL_FLAG = "N"  
OBJECT = USE_OBJECT  
    CLASS = 1  
    USED_BY = "GSFC"  
END_OBJECT = USE_OBJECT  
ARCHIVED_AT = "GSFC"  
PROCESSED_AT = "GSFC"  
END
```

### **C.5.9 AIRS ESDT AIRXADCM ODL**

```
DATA_TYPE_NAME = "AIRXADCM"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "Decommutation map"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
END
```

### **C.5.10 AIRS ESDT AIRXATCM ODL**

```
DATA_TYPE_NAME = "AIRXATCM"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "Conversion method file"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
END
```

### **C.5.11 AIRS ESDT AIRXATCS ODL**

```
DATA_TYPE_NAME = "AIRXATCS"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "Constant sets"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
END
```

### **C.5.12 AIRS ESDT AIRXARYL ODL**

```
DATA_TYPE_NAME = "AIRXARYL"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "Red Yellow limits"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
END
```

### **C.5.13 AIRS ESDT AIRXAGEO ODL**

```
DATA_TYPE_NAME = "AIRXAGEO"  
DATA_TYPE_VERSION = "001"  
INSTRUMENT = "AIRS"  
PLATFORM = "EOSPM1"  
DATA_TYPE_DESCRIPTION = "L1A.geolocation.anc"  
PROVIDER = "Goddard Space Flight Center"  
NOMINAL_SIZE = 1.0  
PROCESSING_LEVEL = "L1A"  
HDF_DATA = "N"  
DYNAMIC_FLAG = "S"  
SPATIAL_FLAG = "N"  
END
```

## C.6 Typical Aqua MODIS PGE ODL File

Listings are provided for the following MODIS ODL files:

### C.6.1 ODL files for Aqua MODIS PGE

The Aqua PGE ODL files are similar to those of Terra PGEs. The main differences are DATA\_TYPE\_NAME for input and output granules:

	Terra	Aqua
L0	MOD000	MODPML0
Static input	MOD01LUT	MYD01LUT
	MOD03LUT	MYD03LUT
	MOD02LUT	MYD02LUT
	MOD07LUT	MYD07LUT
	MOD35ANC	MYD35ANC
Ephemeris	AM1EPHN0	PM1EPHND
Attitude	AM1ATTNF	PM1ATTNR
Products	MOD01	MYD01
	MOD03	MYD03
	MOD02QKM	MYD02QKM
	MOD02HKM	MYD02HKM
	MOD021KM	MYD021KM
	MOD02OBC	MYD02OBC
	MODVOLC	MYDVOLC
	MOD07_L2	MYD07_L2
	MOD07_QC	MYD07_QC
	MOD35_L2	MYD35_L2
	MOD35_QC	MYD35_QC
	MODCSR_G	MYDCSR_G

## Appendix D. List of Responsible Engineers for Each Section

---

Section 1.....	John Corbett
Section 2.....	John Corbett
Section 3 .....	John Corbett
Section 4.....	Alward Siyyid
Section 5.....	John Corbett
Section 6.....	Abdul Shash
Section 7.....	Abdul Shash
Section 8.....	Abdul Shash
Section 9.....	Abdul Shash
Section 10.....	Sukhada Gore
Section 11.....	Sukhada Gore
Section 12.....	Kuan Huang
Section 13.....	Kuan Huang
Section 14.....	Xinmin Hua
Section 15.....	Xinmin Hua
Section 16.....	Alward Siyyid
Section 17.....	Alward Siyyid
Section 18.....	Nick Iascone
Appendix A.....	Sukhada Gore
Appendix B.....	Karin Loya
Appendix C.....	Sharon Ni

This page intentionally left blank.