611-TD-593-001

EOSDIS Core System Project

M&O Procedures: Section 26—SSI&T Operational Procedures

Interim Update

August 2001

Raytheon Company
Upper Marlboro, Maryland

Preface

This document is an interim update to the Mission Operations Procedures Manual for the ECS Project, document number 611-CD-600-001. This document has not been submitted to NASA for approval, and should be considered unofficial.

This document has been revised to meet The Science System Release Plan 6A.05.

Changes from the Previous Version

1. PGE Checkout, checking for ESDIS Standards Compliance in C amended to include C++. Deleted ADA Compliance which is no longer supported in ECS.
2. PGE Registration and Test Data Preparation concerning Operational Metadata (item 6) Information corrected.
3. Changes made to enhance PDPS Troubleshooting in section 26.17.
4. ESDT Management, A new Chapter replaced the old with emphasis on ESDT Maintainance added. Additional ESDT information to prepare Data Types is in section 26.28.
5. Appendix A, Updated to current operational scenario. Examples of Terra MODIS PGE & ESDT ODL files, AIRS PGE &ESDT ODL added. Aqua MODIS PGE ODL section added to highlight the differences between TERRA and AQUA scenarios.

Any questions should be addressed to:

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

This page intentionally left blank.

26. SSIT Operational Procedures

The purpose of this section 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. The SSI&T Operational Procedures for the ECS Project (aka, The Green Book) was the main reference for this section. 162-TD-001-007. 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 have 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 6A.05.05 of ECS to accomplish the steps outlined in the DAAC-IT procedures. The SSI&T operational procedures are given in this section. They are organized by activity. The order in which the procedures appear loosely follows the order in which they will usually be performed. It is highly recommended that a copy of Release 5B/6A ECS Internal Interface Control Document for the ECS Project 313-CD-510-002 be readily available. This document contains more specific details for SSI&T that apply to the many variants of ECS processes that you may encounter.

These procedures present the use of GUIs supplied in Release 6A.05. Some procedures may have a command line equivalent; these are documented in the corresponding GUI help screens but are not presented here in the interest of simplicity. Version 2.0 Operations Tools Manual 609-CD-003-004 should be referred to for more detailed information on how to use GUI’s and command line equivalent usage.

A Note about the Order of Procedures

The science softwareI&T operational procedures contained within this document are ordered. The order is intended to *loosely* suggest a logical sequence which, when used as a "road map", represents an overall, sensible end-to-end SSI&T activity at the Release 6A DAACs. The ordering cannot, however, be interpreted as a detailed, step-by-step guide to SSI&T activities. In addition, since there are many factors that affect the actual SSI&T activities during Release 6A (e.g. Instrument Team deliveries, DAAC policies, agreements between the Instrument Teams and the DAACs, etc.), the ordering in this document can only be suggestive.

Many of the procedures outlined in this document are inter-related. A procedure may assume that another procedure has been completed. In general, the ordering of the procedures reflects this. The user should be aware, however, that this is not the case for all procedures. Therefore, depending on the SSI&T activity, the ordering suggested may not apply. Procedures may require other procedures that appear *after* the procedure requiring them.

Assumptions

All procedures in this section assume the following: that the Instrument Team has delivered the science software to the DAAC and that Release 6A.05 ECS is available at the DAAC.

Conventions

The following conventions are followed for explaining procedures:

Text that should be typed literally in the "Action" column of the procedures is displayed in `courier` font. Text within a literal command that represents a fill-in-the-blank object is displayed in ***`italic courier`*** font. (Example: **`cd`** ***`mydir`*** means type "cd" and then type the name of the correct directory.)

A command line in the "Action" column that should be typed in without a line break will be indicated by an indent in any following lines. The end of the command is indicated by **<ENTER>,** which stands for pressing the **ENTER or RETURN** key.

26.1 DAAC Science Software I&T Support Engineer

Provide DAAC SSI&T execution support, ECS tool and system expertise and science S/W processing problem support. Provide support to scientists in the development and integration of science software for both updates and new science software into the DAAC ECS system.

Interfaces

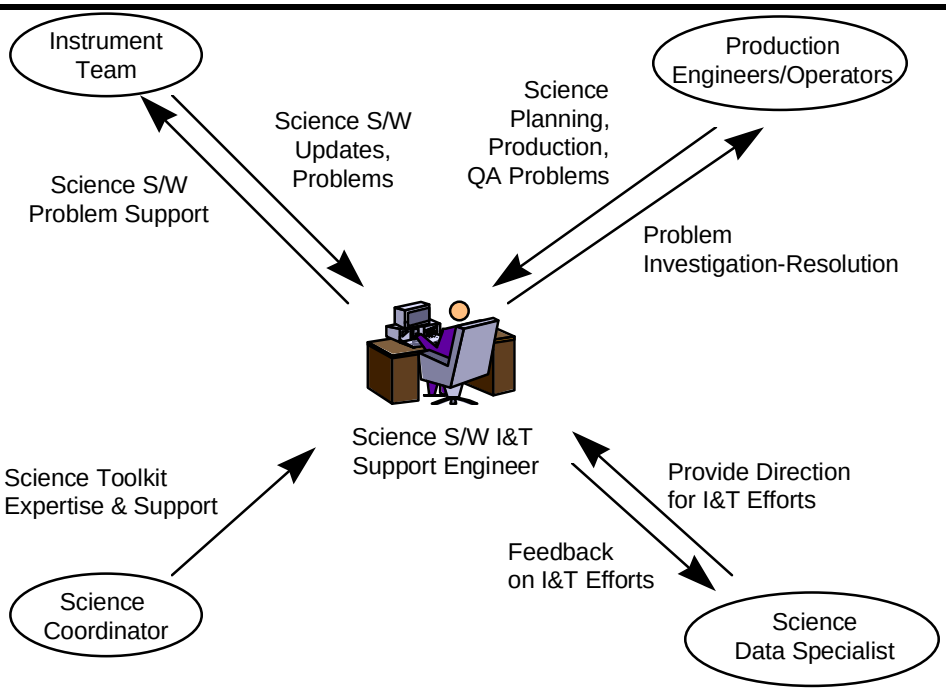


Figure 26.1-1. DAAC Science Software I&T Support Engineer Interfaces

Roles and Responsibilities

1. Provide support to Instrument Teams for the development and integration of science software into the DAAC ECS system.
2. Perform standards checking on all delivered software, including source code, scripts, process control files and related documentation.
3. Provide support for metadata updates and additions for science data products.
4. Support science processing problem investigation and resolution.
5. Recommend - assess, develop, implement changes to science toolkit software.
6. Provide support for integration and test of new and modified toolkit functions into the science software.
7. Support and track configuration of delta changes to science software, calibration coefficient files, relevant input files and documentation.
8. Facilitate migration and integration of major and minor modified versions of the science software into production versions.
9. Assess impacts and support integration and test for production planning and processing software changes. (Planning Workbench GUI)
10. Provide feedback and receive direction on I&T efforts.
11. Recommend-assess improvements for automated tools for SSI&T activities, such as file comparison and viewing data.
12. Write and implement procedures to examine non-standard auxiliary files and files not in HDF EOS format.

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

26.1.1.1 Key Operator Roles

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

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

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

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

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

MODIS Science Data Processing Software Version 2.0 System Description Manual

This manual should be referred to for more detailed information on how to perform the SSI&T operational procedures as they apply to MODIS PGE's. It covers the specific attributes for each individual PGE and setup criteria.

26.1.2 COTS Software Tools

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

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

26.1.3 General Process

The SSI&T process consist of two activities:

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

Key Terms:

- **Product Generation Executives (PGEs)** - The smallest scheduled unit of science software.
- **Delivered Algorithm Package (DAP)** - An ensemble of PGE source code, makefile, documentation, and other related files delivered in a package from the SCF to the DAAC for SSI&T..
- **Process Control File (PCF)** - Relate logical identifiers to physical files and other parameters required by the PGE.
- **Strings** - The processing hardware on which the science software runs.
- **Archive** - A File Storage Type indicating that granules that will be inserted Data Server are intended for long term storage and acquisition for distribution.
- **Collection** - A related group of data granules.
- **Granule** - The smallest data element which is identified in the inventory tables.

- **Product** - A set of output values generated by a single execution of a PGE for archival by ECS. A PGE may generate one or more products whose attributes are defined by the data provider.
- **Reliability** - Software reliability means that the software runs to normal completion repeatedly over the normal range of data inputs and running conditions.
- **Safety** - Software safety means that the software executes without interfering with other software or operations.

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

The following list provides a suggested, logical "road map" for getting science software tested and integrated into the ECS. This list is not intended to cover every situation and variations may be required.

26.1.3.1 GENERAL

- Science Software Integration and Test (SSI&T) is the process by which the science software is tested for production readiness in the DAACs in order to assure its (1) reliability and (2) safety. Prior to the delivery of the ECS software to the DAACs, SSI&T Checkout is conducted on early versions of the Products Generation Executives (PGEs) using separate system modes in the ECS PVC, VATC (Verification and Acceptance Test Configuration), or the DAAC environments.

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

- Most steps in the SSI&T process are inter-related and some steps may assume that another step has been completed. The ordering of the steps is very important but it cannot, however be interpreted as a detailed, step-by-step guide to SSI&T activities.

- Science Software Integration and Test consists of the following activities most of which are fully detailed in Science Software Integration & Test Operational Procedures for the ECS Project (162-TD-001).
- The activities described in the following list are also depicted in a (SSI&T Process Flow Diagrams 1 and 2) SO-1-003 Science Software Integration and Test (SSI&T) Web site listed in the next section below.

26.1.3.2 Science Office Project Instructions

The following are procedures listed from Science Office Project Instructions as they apply to the building of Science Data Products and SSI&T. A review of these documents is highly recommended at this time. They can be accessed from the Web using the following URL: http://dmserver.gsfc.nasa.gov/proj_instr/sopi_index.html

Number:	Subject:	Issue Date
SO-1-002	Earth Science Data Type Generation Procedures PDF	5/11/98
SO-1-003	Science Software Integration and Test (SSI&T) PDF	9/14/98
SO-1-004	Science Office Science Support Internal Processes PDF,	
Science Office Science Support Internal Processes (161-IT-003-001)PDF		6/30/98
SO-1-005 Product Specific Attribute (PSA) Analysis PDF		7/23/98
SO-1-006	PGE Testing PDF	7/07/98
SO-1-007	Earth Science Data Types Testing and Integration PDF	7/24/98
SO-1-008	QA Metadata Update Tool (QAMUT) PDF	7/23/98
SO-1-009	Metadata Works PDF	7/23/98
SO-1-010 ECS Science Metadata Valids Update Procedures PDF		7/24/98
SO-1-011 Metadata-Process Established with MODIS PDF		8/20/98
SO-1-013 Test Data Configuration Management Project Instruction PDF		3/8/00

Figure 26.1.3.2-2. Science Office ECS Project Instructions

26.1.3.3 Pre-SSI&T Activities

1 As the DAP is delivered to a DAAC by the Instrument Team for SSI&T, the PGE listing documentation is reviewed.

2 The DAP is acquired and unpack and the documentation (i.e., packing list, readme, etc.) checked. The DAP contents are further checked by the Science Data Specialist to verify that

the contents match the packing list, agreed-upon directory structures are employed, location of files are correct, and all intended files and directories are present.

3 The Science Data Specialist requests that the CM Administrator place the DAP under Configuration Management control using ClearCase.

4 The SSI&T team checks the science software for standards compliance using the Process Control File Checker to check process control files (PCF), and the Prohibited Function Checker to check source files. Extract and check prologs.

5 The SSI&T team builds the science software into PGEs using the SCF version of the SDP Toolkit. Compile all source code. Link object code with appropriate libraries. . If the SMF files compile successfully, then proceed to Step 11 below; otherwise. the problem needs to be fixed and a successful compile must occur before proceeding further. This may require one or more of the following:

6 Note any error messages and review the included documentation to ensure a proper compile;

7 Check environmental variables;

8 Review the setup files for proper directories and variables;

9 Make corrections and recompile.

10 I f the executable builds successfully, proceed to Step 12. If the build fails, it may necessary to do one or more of the following before proceeding:

11 Check environmental variables;

12 Make corrections and repeat the build.

13 Run the PGE from the Command Line.

14 I f it the execution is successful, then the output files (products) are checked using the SSIT Manager file comparison tools; otherwise, one or more of the following needs to be done before proceeding:

15 Check error logs;

16 Check environmental variables;

17 Review and source the setup files;

18 I f necessary files are missing, then review the PCF file to ensure the existence of correct reference directories.

19 T he SSI&T team runs and profiles the PGEs from the UNIX command line on the SGI, saving the profiling results. They will be used later when entering operational metadata into the PDPS.

20 The SSI&T team collects performance statistics for the PGEs.

21 T he SSI&T team examines the output log files from the PGE runs for any anomalous message. The SSI&T team compares the output product data with the delivered test data using the file comparison tools. If the products do not match the delivered test outputs

(expected outcome), the outputs should be analyzed and the PGE must be re-run. If the products match the delivered test outputs then

22 Steps 10 through 13 are repeated once using the DAAC Toolkit. If the products generated with the DAAC Toolkit match the delivered test output, formal SSI&T may begin.

23 SSI&T team reports any science software problems using the DDTS NCR process.

24 The SSI&T team reports any ECS problems using the DDTS NCR process.

25 The SSI&T team collects and logs all lessons learned.

26.1.3.4 Formal SSI&T Activities

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

2 Construct a PGE ODL file for updating the PDPS database. This involves using the delivery PCF to construct an initial PGE ODL template file, which must then be hand edited to add required metadata. A mapping between logical IDs in the PCF and ESDT ShortNames must be known before this step is done.

3 Install ESDTs on the Science Data Server if verification indicates that they do not already exist. Installation links the PGE to all input and output ESDTs which allows the PGE to run within the PDPS. The Advertising Server must also receive notification of the update. If this fails then the ESDT's must be re-installed again after removing original ESDT's from the SDSRV. Note: While installing ESDT's the SDSRV intermittently coredumps. To clean-up you must remove the ESDT from ADSRV, SBSRV and DDICT and then try again.

4 The Science Metadata.met is updated (PGE & ESDT Object Description Language or ODLs are created). This supplies metadata to the PDPS database

- If the Science Metadata update is successful, then the Operational Metadata is updated; otherwise, the ESDT ODL files may have to be checked for correctness before updating the Operational Metadata.

5 Register the PGEs with associated data in the PDPS database. This step uses the PGE ODL from step 2 above.

6 For each input dynamic data granule needed by the PGE, construct a Target MCF and insert both granule and .met files into the Science Data Server.

7 For each input static granule needed by the PGE, construct a Target MCF.met and insert both into the Science Data Server.

8 Assemble the SSEP (as a tar file) and Insert it to the Science Data Server.

9 Initiate a Production Request (PR) that will result in one or more DPRs.

10 Use the Planning Workbench to plan the PR and hence, run the PGE.

11 M onitor the PGE run using AutoSys. The PGE's progress is monitored using the AutoSys COTS. The distinct steps that are visible on the AutoSys GUI and whose success is evident are Resource Allocation (.Al), Staging (.St), Pre-Processing (.Pr), Execution of the PGE (.EX), Post-processing (.Ps), De-staging (.Ds), and De-Allocation of resources (.Da).

12 I f any of the steps in the execution is not successful, then each failure must be identified and corrected before proceeding to the next step.

13 E xamine the output Production History File from the PGE runs for any anomalous messages. Compare the output product data with the delivered test data using the file comparison tools. . If any of the steps in the execution is not successful, then each failure must be identified and corrected before proceeding to the next step.

14 If the output files match the test output files and they are in Hierarchical Data Format (HDF), they are visualized using the EOSView tool, or the Interactive Display Language (IDL) tool. If the files are not HDF, then IDL is used.

15 Us ing the Planning subsystem, initiate more complex Production Requests if chaining is required.

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

RECORDS

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

PERFORMANCE MEASUREMENTS

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

26.1.4 Preparation and Setup to use the SSI&T Manager Tool

1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

Prior to the remote login, enter **setenv DISPLAY <local workstation IP address>:0.0** where the local workstation IP address represents the IP address you where you are located**.**

You may need to setup the terminal so that the remote host is displayed on your screen. (Sun machine) This is done by clicking on the **Application Manager** icon (the file drawer located at the bottom of the screen), followed by the **Desktop Tools** icon**,** followed by the **Terminal Console** icon

3 Perform a remote login by typing **rlogin [host]** then press the **Enter** key.

The **Enter Password** prompt is displayed.

4 Enter the **password** then press the **Enter** key.

5 Enter the directory where the setup script is located by typing **cd [**directory name**]** then press the **Enter** key.

6 Source the setup script by typing **source [**script name**]** then press the **Enter** key.

The setup script contains directory paths, sets of alias commands, and tools for SSI&T.

- For example, source the SSI&T script: Type **source /usr/ecs/{MODE}/CUSTOM/utilities /.buildrc <RETURN>**
  Note: This step only needs to be done once per login.
- source .buildrc may not be supported on a particular software drop. Therefore the SSI&T scripts will be built into other another script.

7 To ensure access to the multi server environment when needed, the following generic login commands have been established and should be used routinely:

- From a terminal: **xterm -n (host) &**
- From the xterm invoked: **telnet (host)**
- **login ID**
- **pw:**
- **setenv DISPLAY …..:0.0**

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

- **ECS Assistant, ADSRV/DM/IOS, p0ins02,**
- **ECS Assistant, SDSRV/DSS, p0acs03**
- **ECS Assistant, DPS, p0sps06**
- **ECS Assistant, SBSRV/CSS/IOS, p0ins01**
- **SSIT Manager tools, AITTL/DPS t1ais01**
- **Production Request, PLS, p0pls01**
- **Planning Workbench, PLS, p0pls01**
  Note: NETSCAPE should be closed to allow for a full screen GUI to be activated.
- **Monitor PGE, p0pls01**

9 A second xterm should be activated with the same login procdures so as to monitor the (log files) when entering SSI&T files from GUI'**s.**

10 Servers can be brought down in any order. To bring them backup requires that they be brought up in **a sequential order to ensure connectivity**, the order is listed as follows:

- **STMGT, MSS, DDIST, IOS, SDSRV, PDPS**

11 The above servers have unique hosts assigned. Each host needs to be logged into the **generic login: ID, pw:, and then press Enter Key.**

26.1.5 SSIT Software Operating Instructions:

Starting the SSIT Manager GUI:

On workstation **x0ais##**, at the UNIX prompt in a terminal window, type as in step 1 below your user id and password.

NOTE: The **x** in the workstation name will be a letter designating your site:
g = GSFC, **m** = SMC, **l** = LaRC, **e** = EDC, **n** = NSIDC, **o** = ORNL, **a** = ASF, **j** = JPL, **p**=PVC; the ## will be an identifying two-digit number (e.g.,**g0ais01** indicates a Data Processing Subsystem (DPS) workstation at GSFC).

Prior to the rlogin, enter **setenv DISPLAY <local_workstation IP address>:0.0.** The **<ipaddress>** is the ip address of **x0ais##,** and **xterm** is required when entering this command on a Sun terminal.

What the user must do before trying SSIT functionality:

1 **Example:** Log into an Algorithm and Test Tools (AITTL) environment using using a machine so configured. At the PVC this machine is **p0ais01.** A special host has been established using the **id**: and **password**:. Type: **setenv DISPLAY ….:0.0**

2 **setenv <mode>** : **(cd /usr/ecs/<MODE>/CUSTOM/utilities** Note that this only has to be done once per login.

3 This directory should contain scripts pertaining to setting the environment for SSIT Manager. Type in: **EcDpAtMgrStart <mode> &**

- This invokes the **SSIT Manager GUI** which should be displayed.

What must be done via SSIT tools:

Since SSIT is just a calibration of various tools, there is no specific order for which they must be run. Most tools can be brought up from the SSIT Manager GUI as well as started on their own.

The File menu provides the capability to exit the manager. The Tools menu provides access to the various tools that make up SSIT. The Run menu is customizable (allowing you to add your own scripts and tools) by editing the file *ssit_run_menu* in the *data/DPS* directory.

The checklist (first window on the GUI) allows you to check off various activities by double clicking on them. You may enter a commentary on the activity in the second window when checking off a particular item. The file checklist.sample in the data/DPS directory can be edited to change the items in the checklist or its' location.

26.1.6 Updating the Leap Seconds and the Earth Motions files

The toolkit requires Leap Second and Earth Motion updates, weekly and twice weekly respectively, to accurately compute most time conversions. The following scripts have been established to accomplish these tasks as part of ECS support.

- **update_leapsec.sh**

This script updates the leapsec.dat file by ftp-ing to USNO and reformatting the information into the leap seconds file: $PGSHOME/database/common/TD/leapsec.dat
The present script, after obtaining the required file "tai-utc.dat" in the same Series 7 mentioned above, invokes PGS_TD_NewLeap, a C program that performs the actual update work. The function puts the current date in the header of the new leapsec.dat, with a remark that the file was either "Checked" (no new leap second) or "Updated" (new leap second). The date at which the USNO file used in the updating process was put on their server is also listed in the header.

- **update_utcpole.sh**

This script updates the **utcpole.dat** file on the basis of new data obtained by ftp to the U.S. Naval Observatory in Washington, D.C (USNO). Their data file is excerpted and the required fields are reformatted and written into the utcpole file: $PGSHOME/database/common/CSC/utcpole.dat

- **The Leap Seconds file:**

leapsec - file ID: **$PGSHOME/database/common/TD/leapsec.dat**
(Atomic time from International Earth Rotation Service)

Introduced every 12 to 24 months, announced almost 6 months in advance or as little as 90 days notice. Update available from U.S Navy Observatory (USNO).

Interval of update recommend: weekly, except Sundays 17:45 hours to 17:55 Eastern US time. Runtime is approximately 30 seconds**.**

- **The Earth Motion file:**

utcpole – file ID: **$PGSHOME/database/common/CSC/utcpole.dat**
(Record of the Earth's variable of slowing rotation with respect to UTC Time.)
Interval of update recommended: Twice weekly except Sundays 17:45 hours to 17:55 Eastern US time. Recommended scripts be run in the afternoon or evening each Tuesday and Thursday.

26.1.7 Script Name: update_leapsec.sh

The following processing tasks are carried out automatically by the use of this script:

- **Update via: Ftp to USNO, "maia.usno.navy.mil"** file accessed for leapsec**: tai-utc.dat. (Tests connectivity by using "Whazzup")**

- **Function to be applied: PGS_TD_NewLeap,** excerpts and reformats the new information and appends new data and date to **leapsec.dat** file. A remark that the file was either "Checked" (no new Leap second) or "Updated" (new leap second).
26.2.2 Script Name: update_utcpole.sh

The following processing tasks are carried out automatically by the use of this script

- **Update via: Ftp to USNO, "maia.usno.navy.mil"** file accessed for utcpole**: finals.data. (Tests connectivity by using "Whazzup")**

Function to be applied: PGS_CSC_UT1_update, excerpts and reformats the new information and appends new data to **utcpole.dat** file.

Guidelines:

1 The script must be run on a machine that has the Toolkit mounted and which can access the USNO site via ftp and access e-mail. (p0spg01 used at the Performance Verification Center (PVC))

2 For each installed Toolkit (including all modes, such as debug, F77, F90, etc.) the scripts need to be run only once, even if different platforms or operating systems are run. However, if entirely separate Toolkits exist at your installation, with different $PGSHOME home directories, then either the scripts need to be run in each, or the data files can be propagated from a primary Toolkit to the others.

3 It is highly desirable to have outgoing e-mail mounted on the machine of choice, so that error messages may be issued automatically from the scripts in case of failure.

4 If the updating process fails, then the script must be rerun. The Toolkit team should be contacted anytime the scripts are not giving the correct or accurate It is highly desirable to have outgoing e-mail mounted on the machine of choice, information. The 2 sets of scripts do also send an email message to SDP Toolkit mail address when a script fails

5 The Toolkit requires that the two data files not be too stale. Therefore the useful lifetime of the utcpole.dat and leapsec.dat files is 83 days. The Toolkit will issue an error message if no update was performed beyond 83 days. If this occurs you can expect geolocation accuracy to deteriorate to an extent that could require re-running for some of the more stringent users. If Tookit requires a leap second value after this date, an error message will be returned. This generally means that production will cease.

6 Keep the Latest files until your updates are completed! They are useful for a backup should they be needed.

Hardware Needed and Setup Procedures

The user's environment needs to be set up by running the script $PGSBIN/pgs-dev-env.csh or $PGSBIN/pgs-dev-env.ksh, depending on the shell being used. $PGSBIN stands for $PGSHOME/bin/mach, where "mach" stands for one of: sun5, sgi64, sgi, sgi32, ibm, dec, or hp. In other words it is a shorthand for the machine "flavor" you are using, and for sgi, the compiler option. Not all versions are necessarily at each DAAC or SCF, and in some cases the path may be more complicated. For example, at Goddard Space Flight Center DAAC, typical binary directories are /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/bin/sgi64_daac_f77/, or

/usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/bin/sgi64_daac_f90_debug/, for example.

Once the setup script is located and sourced, $PGSBIN is defined and your path includes it. Furthermore, a "PCF", or process control file, $PGS_PC_INFO_FILE is defined, which allows the executable functions invoked by the scripts to find the old data files, which are needed for the updates.

To run the scripts successfully, you must have write permission on the data files.
After the setup is done, just run the scripts. Both scripts (update_utcpole.sh and update_leapsec.sh) are located in the directory $PGSBIN, which will be in your path after the Setup script has been run.

On workstation **x0spg##**, at the UNIX prompt in a terminal window, type **source /data3/ecs/TS1/CUSTOM/daac_toolkit_f90/TOOLKIT/bin/sgi64/pgs-dev-env.csh .** This will set up the various environment parameters, such as PGSHOME, to enable the 64 bit version of the FORTRAN 90 compiler to be run.

NOTE: The **x** in the workstation name will be a letter designating your site:
g = GSFC, **m** = SMC, **l** = LaRC, **e** = EDC, **n** = NSIDC, **o** = ORNL, **a** = ASF, **j** = JPL **p**=PVC; the ## will be an identifying two-digit number (e.g.,**g0spg03** indicates a Science Processor Subsystem workstation at GSFC).

Prior to the rlogin, enter **setenv DISPLAY <local_workstation IP address>:0.0.** The **<ipaddress>** is the ip address of **x0spg**

Example: To Update the Latest Leapsec.dat and Utcpole.dat files perform the following steps:

---

1 telnet to a machine that supports the Toolkit. (**telnet p0spg01**)
2 login: **ID**, **Password:**
3 **setenv DISPLAY ….:0.0**
4 **setenv PGSHOME /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit**
5 **cd /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/bin/sgi_daac_f77** then
6 **source pgs_dev-env.csh**
7 For **leapsec: cd /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/database/common/TD**
8 **cp leapsec.dat leapsec.dat_old**
9 Know thread for Leap Second run:
10 **cd /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/src/TD** then do **ls** – select: update_leapsec.sh or run script for Leap Second type in: **update_leapsec.sh**

A successful update will look like the following

P0spg01{cmops}[288]->update_leapsec.sh
Status of PGS_TD_NewLeap call was (0)
Status of MOVE command was (0)

1 For **utcpole:**
2 **cd /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/database/common/CSC**
3 **utcpole.dat utcpole.dat_old**
4 Know thread for utcpole run:

5 **cd /usr/ecs/OPS/CUSTOM/TOOLKIT/toolkit/src/CSC** then do **ls** – select: update_utcpole.sh or run script for utcpole type in: **update_utcpole.sh**

6 A successful update will look like the following:
p0spg01{cmops}[294]->update_utcpole.sh
Status of PGS_CSC_UT1_update call was (0)
Status of MOVE command was (0)

26.2 Science Software Integration and Test (SSIT) Manager

26.2.1 SSIT Manager Overview

The principal tool used during SSI&T is the SSIT Manager. The SSIT Manager is the top-level graphical user interface (GUI) environment presented to SSI&T personnel. Its purpose is to bring together the tools needed for SSI&T into a single, graphical environment. Please refer to Release 5B Operational Tools Manual 609-CD-510-002, section 4.5 for the latest update on expanded uses of the SSIT Manager.

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

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

Table 26.1-1 presents a summary of the capabilities provided via the SSIT Manager GUI.

Table 26.1-1. Common ECS Operator Functions Performed through the SSIT Manager GUI (1 of 3)

Operating Function	Command/Script or GUI	Description	When and Why to Use
Prepare SSI&T checklist.	SSIT Manager GUI	• SSIT Manager GUI requires a checklist which it can access • the checklist contains all the operational procedures for science software integration and test	• during normal SSIT operations, used to keep track of activities pending and completed • checklist of operational procedures must first be prepared using a text editor. • SSIT Manager must then be linked to the checklist before invoked
Change SSI&T Checklist.	SSIT Manager GUI	• SSIT Manager GUI requires to change the checklist • the checklist contains all the operational procedures for science software integration and test	• Update tracking of activities pending and completed • checklist of operational procedures must first be prepared using a text editor. • SSIT Manager must then be linked to the checklist before invoked
open xterm session.	• open xterm window via Tools pull-down menu • also can be opened via Unix command *xterm*	Standard Unix command line window.	As needed for ad hoc use.
Code analysis.	select SPARCworks via Tools: Code Analysis pull-down menu	Used for ad hoc analysis of science software.	used to debug problems (e.g., memory leaks)
check for standards compliance.	select the following via Tools: Standards Checkers pull-down menu: • FORCHECK • Prohibited Function Checker • Process Control File (PCF) Checker • Prolog extractor	• check FORTRAN 77 science software • check if certain functions are used in the science software which conflict with the production environment • check the syntax of the data in the Process Control File • Extract prologs from science software	• to ensure that science code conforms to ECS standards • to ensure that the delivered PCF is of the proper syntax • To extract prologs from science software

Table 26.1-1. Common ECS Operator Functions Performed through the SSIT Manager GUI (2 of 3)

Operating Function	Command/Script or GUI	Description	When and Why to Use
Product Examination.	select the following via Tools:Product Examination pull-down menu: • IDL • EOSView	Opened via Tools pull-down menu.	• Ad-hoc graphical analysis • For viewing an arbitrary file (e.g., standard product) in HDF format
File Comparison.	select the following via Tools: Product Examination File Comparison • ASCII files • Binary files • HDF files (GUI) • HDF files (hdiff)	Compares the outputs of the science software between the DAAC and SCF.	Ensures that the output that was generated at the SCF when running the science software is the same output that is generated at the DAAC.
Edit text.	select the following via Tools: Text Editors pull-down menu: • Emacs • Xedit	Text editors.	Edit arbitrary text file.
PDPS Database	select the following via Tools:PDPS Database pull-down menu: • PCF ODL Template • Check ODL Files • SSIT Science Metadata Update • SSIT Operational Metadata Update GUI	• creates an ODL file template from the science software PCF • Check the ODL file updates PGE and ESDT SCIENCE metadata in the PDPS /SSIT database • updates PGE OPERATIONAL metadata via GUI in the PDPS /SSIT database	To initialize and update the Planning/Production (PDPS) databases: • SSIT version • Production version

Table 26.1-1. Common ECS Operator Functions Performed through the SSIT Manager GUI (3 of 3)

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

26.1.1 Quick Start Using SSIT Manager

The SSIT Manager provides a common interface to the SSI&T tools. An overview of the SSIT Manager GUI is provided in Section 26.1.2. A more detailed discussion of the tools accessed via this GUI is provided in subsequent sections.

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

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

26.1.1.1 Invoking SSIT Manager From the Command Line Interface

To start SSIT Manager at the Unix command line:

***DpAtMgr ConfigFile* <config_filename> ecs_mode <MODE> &**

where config_filename is the name of the user's personal Process Framework configuration file for this program, as customized from **$ECS_HOME/CUSTOM/cfg/**.

ecs_mode is the ECS mode of operation, e.g., **OPS, TS1**

26.1.1.1.1 Sun Platform

Table 26.1-2 lists the SSI&T command line interfaces for the Sun workstation.

Table 26.1-2. Command Line Interfaces (Sun) (1 of 2)€

Command Line Interface	Description and Format	When and Why Used
EcDpAtMgr	Startup of SSIT Manager.	To do SSI&T, and record items accomplished in the log.
EcDpAtMgrLogDump	Used to dump/print a log file to the screen.	As needed.
xterm	Open a Unix command line window.	As needed.
sparcworks	Ad hoc code analysis	As needed.
ghostview	Postscript file viewer	As needed.
netscape	WWW browser Netscape	As needed.
acroread	PDF file viewer Adobe Acrobat	As needed.
DpAtMgrForcheck	FORTRAN 77 code analysis	Determine whether FORTRAN 77 science software adheres to standards.
EcDpAtBadFuncGui	Prohibited function checker (GUI)	Determine whether science software adheres to standards.
EcDpAtBadFunc	Prohibited function checker (command line)	Determine whether science software adheres to standards.
EcDpAtCheckPCF	Process Control File checker (GUI)	Determine whether PCF is valid.
EcDpAtMgrPrologs	Prolog extractor	Extract science software code prologs.
/data/IDL/idl_4/bin/idl	IDL	As needed.
EOSView	EOSView	HDF file viewer.
EcDpAtMgrXdiff	ASCII file comparison	Compare 2 text files.
EcDpAtBinDiffGui	Binary file difference environment	Compare 2 binary files.
DpAtCheckHdfFile	HDF file comparison (GUI)	Compare 2 HDF files.
DpAtHdiff	HDF file comparison (command line)	Compare 2 HDF files.
xedit	Text editor	As needed.
emacs	Text editor	As needed.
EcDpAt CreateODLtmplate	Create PGE metadata ODL template file.	Before running EcDpAtDefinePGE .
EcDpAtVerifyODL	Verify PGE metadata ODL template file.	Before Running EcDpAtDefinePGE.

Table 26.1-2. Command Line Interfaces (Sun) (2 of 2)€

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

26.1.1.1.2 SGI Platform

It is intended that the SSI&T tools be most often run from the SSIT Manager. A small number of SSIT tools run only on the SGI platform. Because of security considerations, these tools cannot be run from the SSIT Manager on the Sun. They may only be run from the Unix command line on the SGI platform as indicated in Table 26.1-3.

Table 26.1-3. Command Line Interfaces (SGI)€

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

Table 26.1-4 lists SGI platform tools associated with the SSIT process.

Table 26.1-4. SGI Tools Description€

Categories/Tools	Tool Description & Use	Further Information
ProDev Workshop	• ProDev Workshop is a COTS package developed by SGI • This tool is targeted within ECS for applications running on the SGI science processors • ProDev Workshop is a software development support tool which includes several tools that may have applicability to SSIT • Among these tools is the capability to perform static code analysis to aid in the detection of memory leaks	• ProDev Workshop includes online documentation describing its features • Other ProDev Workshop documentation is delivered with ECS. • ProDev Workshop is not available from the SSIT menu. This tool must be started from the Command Line Interface: see Table 4.5.1-3.
PGE Performance	• DpAtRusage is a custom tool developed by ECS • It measures performance parameters such as CPU time used for a PGE linked to the SDP Toolkit, SCF version.	A help message is printed if the tool is invoked without input parameters.

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

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

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

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.

26.2.2 SSIT Manager GUI

This GUI (Figure 26.2-1) is the starting point for SSI&T activities. It provides access to a collection of tools that will be useful for this purpose.

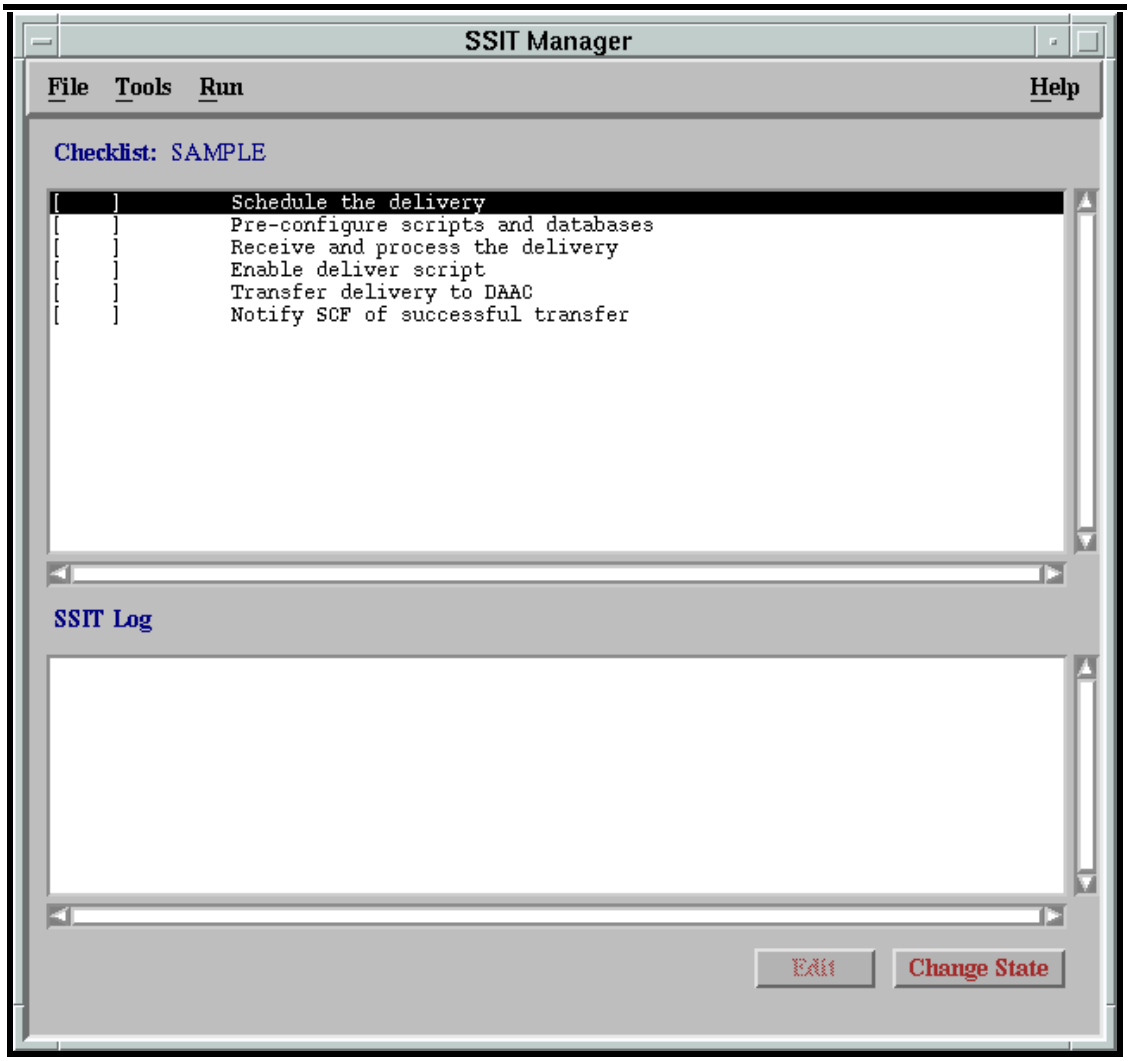


Figure 26.2-1. SSIT Manager Window€

26.2.2.1 General Set Up of the SSIT Manager

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

To set up the environment for the SSIT Manager, execute the procedure steps that follow.

(This procedure was tested by **telnet p0ais01**, ID: **___,** PW:**___, setenv DISPLAY** ***Example:*** **155.157.123.34:0.0** or **setenv DISPLAY p0ais01:0.0** .

1 **setenv ECS_HOME /usr/ecs & setenv <mode>**

2 **cp /usr/ecs/*mode*/CUSTOM/data/DPS/DpAtMgrInternal.pcf $HOME/*mySSITpcf***, press **Return**.

- The ***mode*** is the ECS mode in which you are operating. This mode should be **TS1.**
- The ***mySSITpcf*** is the file name of the private copy of the PCF that the SSI&T operator will use when running the SSIT Manager. The **$HOME** is the environment variable for the user's home directory. For example, **cp /usr/ecs/TS1/CUSTOM/data/DPS/DpAtMgrInternal.pcf $HOME/myPCF**, press **Return**.

3 At the UNIX prompt on the AIT Sun, type **setenv PGS_PC_INFO_FILE $HOME/*mySSITpcf***, press **Return**. (Check **env** for proper home path)

- The ***mySSITpcf*** is the full path name to the private copy of the PCF to be used with the SSIT Manager when you run it (from step 1).
- It may be useful to add this line to your **.cshrc** (or other start up script) so that it is set every time you login.

4 At the UNIX prompt on the AIT Sun, type **cd /usr/ecs/*mode*/CUSTOM/utilities**, press **Return**.

- The ***mode*** is the ECS mode in which you are operating. This mode should be **TS1 or another mode assigned beforehand to operate in.**

5 At the UNIX prompt on the AIT Sun, type **EcDpAtMgrStart <mode> &**
This invokes the **SSIT Manager GUI** which should be displayed.

- The checklist displayed within the GUI will be the default.
  This sets environment variables and other settings needed for running the SSIT Manager.

26.2.2.2 Set Up of a Checklist for the SSIT Manager

The SSIT Manager offers the capability of maintaining user-defined checklist of SSI&T activities. The checklist is presented in the main window of the SSIT Manager. A default checklist is displayed unless a new checklist is specifically created. This procedure explains how to set up a customized checklist.

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

Creating a User-Defined Checklist for the SSIT Manager:

1 a From the SSIT Manager, click on the **Tools** menu, then choose **Product Examination**, then **EOSView**.

- 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.

1 Go to the proper area by typing **cd /usr/ecs/TS1/CUSTOM/eosview** <**RETURN**>
2 Start EOSView by typing **EOSView** <**RETURN**>

2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Open.**

- The **Filter** 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/****.

- 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 **Directories** 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***, 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 **Dimension** GUI to be displayed instead.

6 In the GUI labeled **EOSView - *MyOutputFile.hdf***, click on the **Attributes** menu and select **Global**.

- 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.

26.2.3 SSIT Manager Tools

There are several tools that are accessible through the SSIT Manager GUI. After selecting the TOOLS menu option of the menu bar, a set of options is available. See Figure 26.2.3-1, which indicates the use of the Tool menu item.

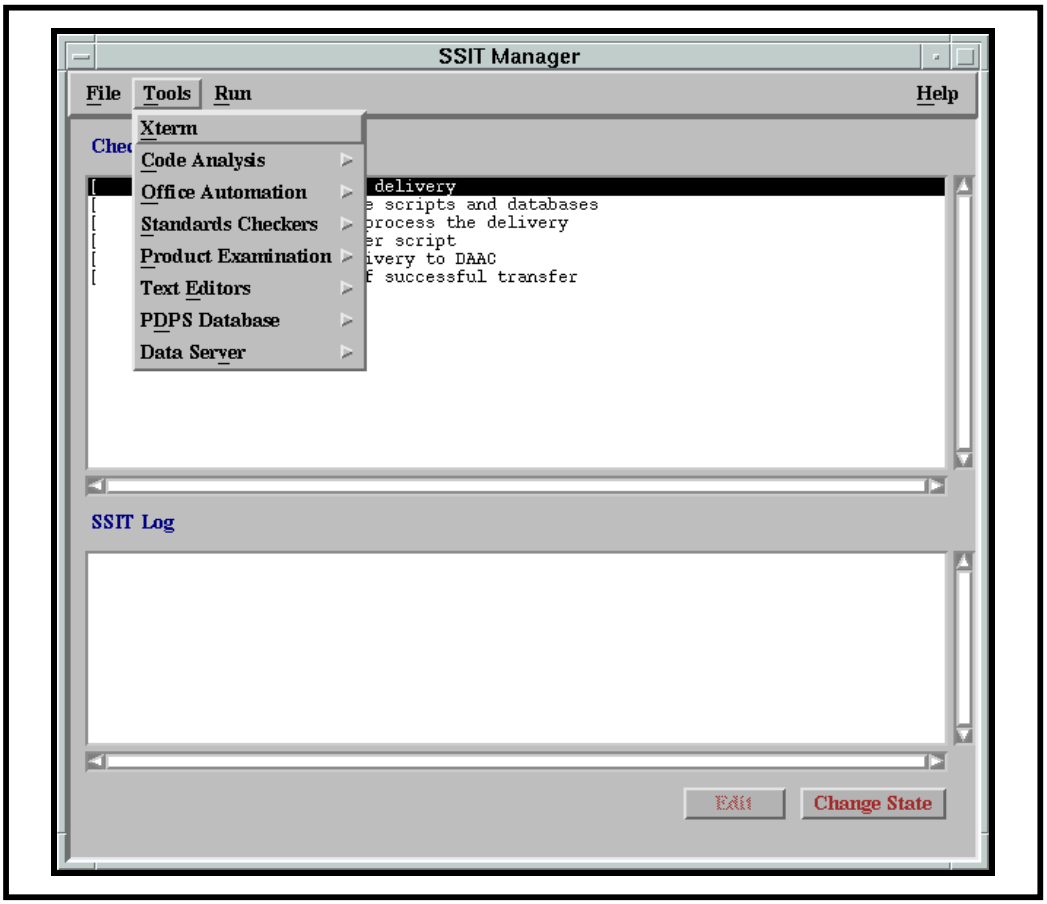


Figure 26.2.3-1. SSIT Manager Window - Tools Menu

26.2.4 Using the SSIT Manager:

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

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

6. prohibitedFunctions.C.txt
7. prohibitedFunctions.F77.txt
8. prohibitedFunctions.F90.txt
9. If the source code files to be checked are in a VOB in ClearCase, a view has been set before the SSIT Manager was started.

26.3 Delivered Algorithm Package (DAP) - Insert/Acquire, Unpack, Subscription

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

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

The delivery mechanism for DAPs can be electronic (e.g. via UNIX ftp) or physical media (4 mm or 8 mm digital audio tapes).

26.3.1 Acquiring the Delivered Algorithm Package (DAP)

The following procedures are used by the SSIT team to acquire DAPs.

26.3.1.1 Acquiring the DAP via FTP

FTP is a method that the SSIT team uses in order to receive the science software. The following example demonstrates the FTP of the tar file from a remote machine.

Acquiring the DAP via FTP

---

1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

3 At a UNIX prompt, type **cd *DeliveryPathname***, then press the **Enter** key.

- The ***DeliveryPathname*** is the full path name to the directory that has been set aside for ftp pull of DAPs from the Instrument Team. For example, **cd /home/*user*** where ***user*** is the user's login directory, then press the **Enter** key.
- If the DAP is to be copied into a subdirectory, change to this subdirectory.

4 At a UNIX prompt, type **ftp *machineIPaddress***, then press the **Enter** key.

The ***machineIPaddress*** is the IP address or fully qualified domain name of the remote SCF machine. For example, **ftp 192.266.53.2**, then press the **Enter** key.

Or for example, **ftp g0ais01.gsfc.ecs.nasa.gov,** then press the **Enter** key. The remote machine will likely display some messages and then prompt for a login name.

An ftp session is established.

5 At the ftp prompt on the remote machine, enter user login name, then press the **Enter** key.

The remote machine will typically respond with **331 Password required for *username*:**

6 At the ftp prompt on the remote machine, enter user password, then press the **Enter** key.

- The remote machine will typically respond with **230 User *username* logged in** and display the **ftp>** prompt for further ftp commands.

7 At the ftp prompt on the remote machine, type **cd *DAPpathname*** then press the **Enter** key.

- The ***DAPpathname*** is the full path name to the directory on the remote machine containing the DAP to retrieve. For example, **cd /home/mac ,** then press the **Enter** key. The directory location should be known.

8 At the ftp prompt on the remote machine, type **binary**, then press the **Enter** key.

- The **binary** command causes subsequent file transfers to be in binary mode, preserving the integrity of the file to retrieve without interpretation (as would be done in ASCII mode.
- The system will typically respond with the message **200 Type set to I** indicating that binary mode has been set.

9 At the ftp prompt on the remote machine, type **get *DAPfilename***, then press the **Enter** key.

- The ***DAPfilename*** is the file name of the DAP to retrieve.
  For example, type **get TestPGE.tar**, then press the **Enter** key.
- The user may need to type **dir** then press **Enter** to display a listing of the files in the current directory. The system will likely display several lines of messages once the transfer has completed. For large files, this may take a long time (minutes to hours depending upon the size of the DAP and the bandwidth of the connection).

10 At the ftp prompt on the remote machine, repeat step 9 or type **quit**, then press the **Enter** key.

- Typing **quit** and pressing **Enter** closes the ftp connection with the remote machine.
- Retrieve other DAP files by repeating step 9. The DAPs retrieved will reside in ***DeliveryPathname*** on the local machine.

11 At the UNIX prompt type **cp /home/mac/TestPGE.tar**, then press the **Enter** key.

- This step will copy the DAP tar file into their working directory.

26.3.1.2 Acquire the DAP from the Archive after Ingest

The **Insert** service is used to put the DAP into the Science Data Server. After it is ingested, the **Acquire** service is used to retrieve it.

The DAP is acquired from Data Server and placed in the specified directory. Note there will be 2 files, the DAP itself (a big tar file) and the metadata associated with the DAP. The metadata may be helpful in the creating the SSAP.

When a DAP is inserted into the Data Server by Ingest, an email is sent to all users who subscribe to that event (Section 26.3.2).

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

DAP Acquire Procedures:

1 If not already on an AIT Sun, log onto one from your current machine.

2 Bring up the SSIT Manager GUI. At the UNIX prompt, type **mgr** (if alias has been established)

3 After a short while, the SSIT Manager GUI will appear. From the SSIT Manager top menu bar, select **Tools -> Data Server -> Acquire DAP**
See figure 26.2.3-1. If the SSIT Manager GUI is used to initiate the DAP processing, Step 4 can be skipped.

4 Alternately, one can initiate the DPA processing sequence from the command line. To do this.

5 Type **source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc <RETURN>**
Note: This step only needs to be done once per login

6 Type **/usr/ecs/TS1/CUSTOM/bin/DPS/DpAtStageAlgorithmPackage.sh <RETURN>**

7 The user will be prompted with:

```
        ** DAP Staging Tool **
Configuration filename? (enter for default: DpAtAA.CFG)
```

To respond, type **<RETURN>**

8 The user will be prompted with:

ECS Mode of operations? (enter for default: OPS)

To respond, type **TS1 <RETURN>**

9 The user will be prompted with:

Name of email message file (including path)?

To respond, type the required file name plus the path, e.g.,
/home/diascone/emessage01.asc <RETURN>

The user will be prompted with:

Directory to receive staged file?

To respond, type the required directory, e.g.,
/home/diascone/staged <RETURN>

26.3.2 Unpacking a DAP

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

The file name extension is usually an indication of the packing utility used and DAP files should use this convention. DAP files that have been packed using the UNIX *tar* utility will usually have .tar as a file name extension indicating a tar file. If the DAP has been further compressed using the UNIX *compress* utility, the file name extension is typically .tar.Z indicating a compressed tar file. For DAP files packed with the *gzip* utility, the .zip file name extension is generally used.

When unpacking is performed on a DAP, the contents of the packed file are moved from the tar archive to local disk. If the DAP tar file contains directories as well as files, these directories will be created in the same structure as in the tar file. This structure typically reflects the directory structure from which the tar file was created in the first place at the SCF. Once a tar file has been unpacked, the original tar file will still exist unaltered.

Unpacking a DAP

1 Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

3 At a UNIX prompt, type **cd *UnpackPathname***, then press the **Enter** key.

4 The ***UnpackPathname*** is the path name of the directory that has been set aside for unpacking of DAPs.

5 This directory now contains the DAP tar file. For example, **cd /home/*user***, where ***user*** is the user's login directory, then press the **Enter** key.

6 If the tar file is compressed, at a UNIX prompt, type **uncompress *PackedDAP.Z***, then press the **Enter** key.

7 The ***PackedDAP.Z*** is the file name of the compressed DAP file.

- The file name extension of **.Z** is a convention indicating UNIX compressed files. The **uncompress** utility expects this file name extension by default. A resulting error may indicate that the DAP file was not compressed or that another compression utility was used. If the file name extension was **.Z**, the uncompressed version will have the same file name but without the **.Z**, for example ***PackedDAP***.
- The tar file for the SSI&T Training will not be compressed.

8 At the UNIX prompt, type **tar xvf *PackedDAP***, then press the **Enter** key.

- The ***PackedDAP*** is the file name of the uncompressed DAP file.
- The tar archive will be unpacked in the current directory. If the archive contained directories and subdirectories, these will be created by the tar utility and populated by the files that belong.

26.3.4 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.
2. The Target MCF for the DAP has been created for the Insert.
3. The SSIT Manager is running.

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

1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Insert Test Dynamic**.

- An 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. For example, **TS1** or **OPS.** Press **Return.**

4 At the program prompt **ESDT short name for the file(s) to insert?** type ***ESDTShortName***, press **Return**

- The ***ESDTShortName*** is the ShortName for the DAP file, which is DAP.

5 At the program prompt **ESDT Version for the file(s) to insert?** Type in the ESDT version and press **Return.**

6 At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?**

7 At the program prompt **Single Filename to Insert? (including FULL path)** type ***pathname/GranuleFilename***, press

- The ***pathname/GranuleFileName*** is the full path name and DAP file name.

8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** , Type ***pathname/DAP.tar.met*** and press **Return**.

- ***pathname*** is full name of the path and ***DAP.tar.met*** is the name of the associated .met file.

9 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.

- If continuing, repeat steps 2 through 8.

26.3.5 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, DAPPGEVersion, and DAPSWVersion.

GROUP = INVENTORYMETADATA

```
GROUP            = INVENTORYMETADATA
  GROUPTYPE        = MASTERGROUP

  GROUP            = ECSDATAGRANULE

    OBJECT         = LOCALGRANULEID
```

```
    NUM_VAL           = 1
    VALUE             =
"MOD04L_O.A1997226.0704.002.1999204222146.hdf"
  END_OBJECT          = LOCALGRANULEID

  OBJECT              = PRODUCTIONDATETIME
    NUM_VAL           = 1
    VALUE             = "1999-07-23T22:21:46.000Z"
  END_OBJECT          = PRODUCTIONDATETIME

  OBJECT              = DAYNIGHTFLAG
    NUM_VAL           = 1
    VALUE             = "Day"
  END_OBJECT          = DAYNIGHTFLAG

  OBJECT              = REPROCESSINGACTUAL
    NUM_VAL           = 1
    VALUE             = "processed once"
  END_OBJECT          = REPROCESSINGACTUAL

  OBJECT              = LOCALVERSIONID
    NUM_VAL           = 1
    VALUE             = "002"
  END_OBJECT          = LOCALVERSIONID

  OBJECT              = REPROCESSINGPLANNED
    NUM_VAL           = 1
    VALUE             = "further update is anticipated"
  END_OBJECT          = REPROCESSINGPLANNED

 END_GROUP            = ECSDATAGRANULE

 GROUP                = COLLECTIONDESCRIPTIONCLASS

  OBJECT              = VERSIONID
    NUM_VAL           = 1
    VALUE             = 1
  END_OBJECT          = VERSIONID

  OBJECT              = SHORTNAME
    NUM_VAL           = 1
    VALUE             = "MOD04L_O"
  END_OBJECT          = SHORTNAME

 END_GROUP            = COLLECTIONDESCRIPTIONCLASS

 GROUP                = INPUTGRANULE

  OBJECT              = INPUTPOINTER
    NUM_VAL           = 20
    VALUE             =
("MOD04_L2.A1997226.0725.002.1999204060629.hdf",
```

```
"MOD04_L2.A1997226.0730.002.1999204060716.hdf",
"MOD04_L2.A1997226.0735.002.1999204060620.hdf",
"MOD04_L2.A1997226.0740.002.1999204060637.hdf",
"MOD04_L2.A1997226.0745.002.1999204060647.hdf",
"MOD04_L2.A1997226.0750.002.1999204060644.hdf",
"MOD04_L2.A1997226.0800.002.1999204195105.hdf",
"MOD04_L2.A1997226.0805.002.1999204194908.hdf",
"EOSAM1_1997-08-14.eph", "EOSAM1_1997-08-14.att")
   END_OBJECT          = INPUTPOINTER

 END_GROUP             = INPUTGRANULE

 GROUP               = PGEVERSIONCLASS

  OBJECT               = PGEVERSION
   NUM_VAL             = 1
   VALUE               = "2.3.0"
  END_OBJECT           = PGEVERSION

 END_GROUP             = PGEVERSIONCLASS

 GROUP               = RANGEDATETIME

  OBJECT               = RANGEENDINGDATE
   NUM_VAL             = 1
   VALUE               = "1997-08-14"
  END_OBJECT           = RANGEENDINGDATE

  OBJECT               = RANGEENDINGTIME
   NUM_VAL             = 1
   VALUE               = "08:45:02"
  END_OBJECT           = RANGEENDINGTIME

  OBJECT               = RANGEBEGINNINGDATE
   NUM_VAL             = 1
   VALUE               = "1997-08-14"
  END_OBJECT           = RANGEBEGINNINGDATE

  OBJECT               = RANGEBEGINNINGTIME
   NUM_VAL             = 1
   VALUE               = "07:04:58"
  END_OBJECT           = RANGEBEGINNINGTIME

 END_GROUP             = RANGEDATETIME

 GROUP               = ORBITCALCULATEDSPATIALDOMAIN

   OBJECT              =
ORBITCALCULATEDSPATIALDOMAINCONTAINER
    CLASS              = "1"

    OBJECT             = EQUATORCROSSINGDATE
```

```
      CLASS             = "1"
      NUM_VAL           = 1
      VALUE             = "1997-08-14"
    END_OBJECT          = EQUATORCROSSINGDATE

    OBJECT              = EQUATORCROSSINGTIME
      CLASS             = "1"
      NUM_VAL           = 1
      VALUE             = "07:52:01.841571Z"
    END_OBJECT          = EQUATORCROSSINGTIME

    OBJECT              = ORBITNUMBER
      CLASS             = "1"
      NUM_VAL           = 1
      VALUE             = 24565
    END_OBJECT          = ORBITNUMBER

    OBJECT              = EQUATORCROSSINGLONGITUDE
      CLASS             = "1"
      NUM_VAL           = 1
      VALUE             = 0.700127
    END_OBJECT          = EQUATORCROSSINGLONGITUDE

  END_OBJECT            =
ORBITCALCULATEDSPATIALDOMAINCONTAINER

 END_GROUP              = ORBITCALCULATEDSPATIALDOMAIN

 GROUP                  = SPATIALDOMAINCONTAINER

  GROUP                 = HORIZONTALSPATIALDOMAINCONTAINER

   GROUP                = GPOLYGON

    OBJECT              = GPOLYGONCONTAINER
      CLASS             = "1"

     GROUP              = GRINGPOINT
       CLASS            = "1"

      OBJECT            = GRINGPOINTLONGITUDE
        NUM_VAL         = 6
        CLASS           = "1"
        VALUE           = (160.828918, 57.056137, 44.058830, -
1.859100, 34.022232, -114.758156)
      END_OBJECT        = GRINGPOINTLONGITUDE

      OBJECT            = GRINGPOINTLATITUDE
        NUM_VAL         = 6
        CLASS           = "1"
        VALUE           = (69.526398, 23.441734, -65.489883, -
58.831890, 26.757355, 81.381241)
```

```
        END_OBJECT          = GRINGPOINTLATITUDE

        OBJECT              = GRINGPOINTSEQUENCENO
          NUM_VAL           = 6
          CLASS             = "1"
          VALUE             = (1, 2, 3, 4, 5, 6)
        END_OBJECT          = GRINGPOINTSEQUENCENO

      END_GROUP             = GRINGPOINT

      GROUP                 = GRING
        CLASS               = "1"

        OBJECT              = EXCLUSIONGRINGFLAG
          NUM_VAL           = 1
          CLASS             = "1"
          VALUE             = "N"
        END_OBJECT          = EXCLUSIONGRINGFLAG

      END_GROUP             = GRING

    END_OBJECT              = GPOLYGONCONTAINER

  END_GROUP                 = GPOLYGON

 END_GROUP                  = HORIZONTALSPATIALDOMAINCONTAINER

END_GROUP                   = SPATIALDOMAINCONTAINER

GROUP                       = ASSOCIATEDPLATFORMINSTRUMENTSENSOR

  OBJECT                    =
ASSOCIATEDPLATFORMINSTRUMENTSENSORCONTAINER
    CLASS                   = "1"

    OBJECT                  = ASSOCIATEDSENSORSHORTNAME
      CLASS                 = "1"
      NUM_VAL               = 1
      VALUE                 = "CCD"
    END_OBJECT              = ASSOCIATEDSENSORSHORTNAME

    OBJECT                  = ASSOCIATEDPLATFORMSHORTNAME
      CLASS                 = "1"
      NUM_VAL               = 1
      VALUE                 = "AM-1"
    END_OBJECT              = ASSOCIATEDPLATFORMSHORTNAME

    OBJECT                  = ASSOCIATEDINSTRUMENTSHORTNAME
      CLASS                 = "1"
      NUM_VAL               = 1
      VALUE                 = "MODIS"
    END_OBJECT              = ASSOCIATEDINSTRUMENTSHORTNAME
```

```
   END_OBJECT          =
ASSOCIATEDPLATFORMINSTRUMENTSENSORCONTAINER

 END_GROUP           =
ASSOCIATEDPLATFORMINSTRUMENTSENSOR

END_GROUP           = INVENTORYMETADATA

END
```

26.3.6 Mail Template

```
From ts2cm@t1ins01.vatc.ecs.nasa.gov Thu Jan  8 16:32 EST 1998
Received: from t1ins02 (t1ins02.vatc.ecs.nasa.gov [198.118.232.41])
          by t1ins01.vatc.ecs.nasa.gov (8.8.6/8.8.4) with SMTP
        id QAA10950 for <dps@t1ins01.vatc.ecs.nasa.gov>; Thu, 8 Jan 1998
16:32:01 -0500 (EST)
Received: by t1ins02 (SMI-8.6) id QAA19469; Thu, 8 Jan 1998 16:32:01 -0500
From: Code2 Install Team <ts2cm@t1ins01.vatc.ecs.nasa.gov>
Date: Thu, 8 Jan 1998 16:32:01 -0500
Message-Id: <199801082132.QAA19469@t1ins02>
Subject: ECS Notification
To: dps@t1ins01.vatc.ecs.nasa.gov
Content-Type: text
Content-Length: 79
Status: RO

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[VTC:DSSDSRV]:20:SC:MODPTQKM.001:50
43
```

UR to be replaced with new UR.

26.3.7 Examining the Validity of Product Metadata

1. Login to the Science Data Server platform (p0acs03 in the PVC).
2. "**cd /usr/ecs/<*MODE*>/CUSTOM/logs**"
3. Open, in a text editor, EcDsScienceDataServer.ALOG
4. Search for the string "**Begin Metadata Validation (*ESDTShortName*)**".

Between this location in the file and the corresponding End Metadata Validation, you will find errors, if present, in the metadata test products. Also, there will be a message stating if metadata is Valid, metadata has Warnings or metadata is Invalid. This will depend on the nature of any errors, if present, in the product.

26.3.8 Insert Testing of Products

This same system is used to test products that are the DAP (Delivered Algorithm Package) products delivered to a DAAC for SSI&T. Typically, the products are a compressed TAR file with a file name of the form *string*.tar.Z After initial processing, the product is broken apart into its components, and those components will be subsequently processed and used based on their intented function.

The **insert** service is used to put the test .hdf file into the Data Server. Once the test .hdf file is in the Data Server, the **acquire** service is 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 **Tools** menu, then choose **Data Server** and then **Insert Test Dynamic**.

- An 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. For example, **TS1** or **OPS.** Press **Return.**

4 At the program prompt **ESDT short name for the file to insert?** type ***ESDTShortName***, press **Return**

- The ***ESDTShortName*** is the ShortName for the product to test.

5 At the program prompt **ESDT Version for the file(s) to insert?** Type in the ESDT version and press **Return.**

6 At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?**

7 At the program prompt **Single Filename to Insert? (including FULL path)** type ***pathname***/faked.hdf, press

- The ***pathname/faked.hdf*** is the full path name and file name of a test .hdf file that is inserted into the Data Server.See description of this file below..

8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** , Type ***pathname/ESDT_ShortName.met*** and press **Return**.

- ***pathname*** is full name of the path and ***ESDT_ShortName.met*** is the name of the .met file of the product to be tested.

9 If the **insert** is successful, a **UR** is returned. Create a mail message with the name ESDT_ShortName.mail and copy the mail header described below. Copy and paste the **UR** into this .mail file and save it.

10 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to test **insert**s of additional products.

- If continuing, repeat steps 2 through 9.

26.4 Science Software 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).

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

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

26.4.1 ClearCase Overview

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

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

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

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

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.

A Versioned Object Base is defined by the following characteristics:

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

A VOB is comprised of:

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

26.4.2 Creating a View in ClearCase

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

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

A view is defined by the following characteristics:

A working context for an individual developer or closely coordinated group.

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

A view is comprised of:

View storage area (typically in a local machine) - private storage for checked-out files, derived objects and private files.
Configuration Specification - set of rules which determine the version of a file the view will see.
View-tag - Name given to the view (ex. angies_view), view-tags are registered in /urs/adm/atria/view_tags.
Objects stored in a view:
Checked-out versions of file elements.
Unshared derived objects.

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

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

Assumptions:

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

26.4.2.1 Creating a View in ClearCase Using Command Lines

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

3 At a UNIX prompt type **cleartool lsview**, then press the **Enter** key.

- The **lsview** command displays the pathname to the storage location of the views.

4• At a UNIX prompt type **cleartool mkview -tag *ViewName ViewPath/ViewName*.vws,** then press the **Enter** key.

The ***ViewPath*** is the full path to the directory where views are stored. The SA should supply this information. A typical example is **/net/mssg1sungsfc/viewstore/.**

The ***ViewName*** is the user selected name for the view. The file name for the view must end in "**.vws".**

For future reference, the corresponding ClearCase GUI procedures are included in the following section.

26.4.2.2 Creating a View in ClearCase using the File Browser Screen

Selecting a view listed in the View Tag Browser screen brings up the File Browser, or main screen, shown in Figure 26.4.2-1.

Displays the directory name of the current VOB, just below the toolbar.
Displays the content of the directory in the space below the directory's name.

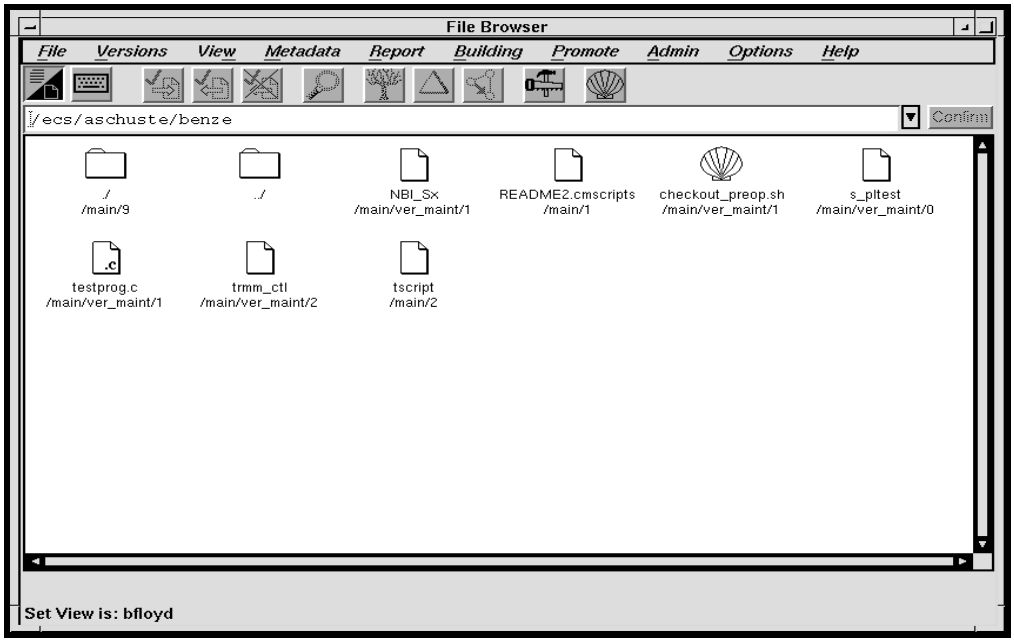
- 

Figure 26.4.2-1. ClearCase File Browser Screen (Main Screen)

Procedures

1• The user should log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.
Cursor moves to the **Password** field.

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

3 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
- The ClearCase **Transcript** screen is displayed as the View Tag Browser loads.
- The ClearCase **View Tag Browser** screen is displayed listing available views.

4 To create a view for checking in the software change package, select a known View and press the **Enter** key.
- The File Browser window is displayed.

5 Select **File→Execute→Single Command.**
- The String Browser window is displayed.

The prompt Enter shell command to run is displayed.

6 Invoke the make view command by typing **mkview [filename]** on the UNIX command line and press the **Enter** key.
- The **tempdisp** window appears.

- The **View [filename] Created Successfully** and the **Cache Updated for View [filename]** prompts are displayed.

7 Close the **tempdisp** window by clicking on the window and press the **Enter** key.

- The **tempdisp** window closes.

8 Select **View →List** from the menu.

- The **View Tag Browser** is displayed.

9 Find the new view by scrolling through the list until the new view is observed.

26.4.3 Setting a View in ClearCase

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

26.4.3.1 Setting a View in ClearCase Using Command Lines

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2• At a UNIX prompt type **cleartool setview *ViewName*** where ***ViewName*** is the user's view created in the previous section, then press the **Enter** key.

26.4.3.2 Setting a View Using the File Browser Screen in ClearCase

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

Cursor moves to the **Password** field.

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

3 Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
The ClearCase **Transcript** screen is displayed as the View Tag Browser loads.
The ClearCase **View Tag Browser** screen is displayed listing available views.

4• To set a view, select a known View and press the **Enter** key.
The File Browser window is displayed.

5• Select **File→Execute→Single Command.**
The String Browser window is displayed.
The prompt **Enter** shell command to run is displayed.

6• Invoke the set view command by typing **setview ViewName** on the UNIX command line and press the **Enter** key.
ViewName is the name of the view to set.

26.4.4 Creating a New Directory

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

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

1. A VOB has been created at the UNIX directory.
2. A view has been created.

26.4.4.1 Creating a New Directory in ClearCase Using Command Lines

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

3 At a UNIX prompt type **cleartool setview *ViewName***, then press the **Enter** key.

The ***ViewName*** is the user's view.

4 At a UNIX prompt type **cleartool lsvob,** then press the **Enter** key.

This command lists all the VOBs and allows the identification of the SSI&T VOB.

5 At a UNIX prompt type **cd *pathname***, then press the **Enter** key.

- The ***pathname*** is the full path name of the parent directory in the VOB in which the new directory is to be added. . For example, if a new directory is to be added under /VOB1/pge4, type **cd /VOB1/pge4**, . (note the space and then "dot" at the end of the command).

6• At a UNIX prompt type **cleartool checkout -nc .** then press the **Enter** key.

This command checks out the current directory. Note the dot for the directory.

The **-nc** is a keyword used when no comments are to be made for this action.

7• At a UNIX prompt type **cleartool mkdir -nc *dirname,*** then press the **Enter** key.

The ***dirname*** is the name of the new directory being created.

8• At a UNIX prompt type **cleartool checkin -nc *dirname,*** then press the **Enter** key.

This command checks in the new directory named ***dirname***.

9 At a UNIX prompt type **cleartool checkin -nc .** then press the **Enter** key.

This command checks in the current directory.

26.4.4.2 Entering a New Directory Using the File Screen Browser into ClearCase

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

Cursor moves to the **Password** field.

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

3• Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.

- The ClearCase **Transcript** screen is displayed as the View Tag Browser loads.
- The ClearCase **View Tag Browser** screen is displayed listing available views.

4• Select **File→Execute→Single Command.**
The String Browser window is displayed.
The prompt **Enter shell command to run** is displayed.

5• Invoke the make directory element by typing **mkdir [filename]** on the UNIX command line and press the **Enter** key.

6• Invoke the make element command by typing **mkelem [directory name]** on the UNIX command line and press the **Enter** key.

7• Type into the directory input box of the **File Browser** the name of the directory in the VOB to be checked out, press the **Enter** key, then follow the menu path **Version→Checkout→Reserved: no comment**.

- In order to add new files to ClearCase, the directory in which the files are to be added must be checked out first.
- ClearCase forces the checkout onto a maintenance branch to isolate the maintenance activity.
- If someone else has already checked out the directory, permission to check out the directory is denied. A separate shell window is displayed.

8• Cancel the checkout of the element if it is decided that no changes are to be made by typing into the directory input box of the **File Browser** the name of the directory to be checked in, press the **Enter** key, then follow the menu path **Version→Uncheckout→Unreserved: no comment**,

9• On the **File Browser** screen, follow the menu path **File→Exit**.
The ClearCase Graphical User Interface session is closed.

26.4.5 Importing files into ClearCase

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

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

1. A VOB and subdirectory are created to hold these files.
2. No object files or executables exist in the source code directory.

3. The PGE was received with a directory structure that contains various types of files.
4. These files will be entered into ClearCase and will maintain the same directory structure as the delivery structure.

26.4.5.1 Importing a Single File into ClearCase

Procedure:

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

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

3 At a UNIX prompt, type **cleartool setview *ViewName***, press **Enter**
The ***ViewName*** is the name of the ClearCase View.

4• At the UNIX prompt, type **cd *pathname***, then press the **Enter** key.
The ***pathname*** is the full path name of the subdirectory in the VOB into which the file is to be checked in. . 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 CM Administrator.

5• At a UNIX prompt, type **cp *pathname/filename* .**, press **Enter** (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 .**,

6• At the UNIX prompt, type **cleartool checkout -nc .**, press **Enter** (note the space and then "dot" at the end of the command).
This command checks out the current directory (represented by the "dot") from ClearCase.
Adding a new file (or element) to a directory represents a modification of the directory. Hence, the directory must be checked out before a file can be checked in.

7 At a UNIX prompt, type **cleartool mkelem -nc *filename***, then press the **Enter** key.
The ***filename*** is the name of the file that was copied over in step 5 and is the file that will be checked into ClearCase.
This command creates a ClearCase element from the file in preparation for checking it in.
The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the make element step.

8• At the UNIX prompt, type **cleartool checkin -nc *filename***, then press the **Enter** key.
The ***filename*** is the name of the file to be checked into ClearCase.
This command performs the check in of the file.

*The **-nc flag** means **"no comment"**; it suppresses ClearCase from prompting for a comment to be associated with the checkin step.* 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).]

9• At the UNIX prompt, type **cleartool checkin -nc .**, press **Enter** (note the space and then "dot" at the end of the command).

This command checks in the current directory (represented by the "dot") into ClearCase.

The adding of an element (here, a file) represents a modification to the directory and hence, the new version of the directory must be checked back in.

The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the checkin step.

26.4.5.2 Importing Multiple Files into ClearCase

The DAP for the synthetic PGE contains only one source code module and a minimal number of other files. A real PGE will generally contain many source files, header files, and multiple other types of files stored in a standard type of directory structure which is retained when the PGE is packed into the tar file. The script provided by ClearCase is used for the purpose of making another load script to enter all of the DAP files along with the directory structure at one time. The final step of running the load script can only be performed by the DAAC Administrator.

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

This procedure is geared toward science software deliveries. In such cases, science software is delivered in the form of a UNIX *tar* files. A *tar* file has been unpacked (un*tar*-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.

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

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

Importing Multiple Files Into ClearCase

---

1 At a UNIX prompt, type **cd *ParentPathname***, then press the **Enter** key

The ***ParentPathname*** is the path name of the directory that *contains* the directory structure to be brought into ClearCase. This is *not* the VOB.

2 At the UNIX prompt, type **clearcvt_unix -r *DirName***, then press the **Enter** key.

The *DirName* is the name of the directory in which it and everything below it is to be brought into ClearCase.

A conversion script will be then be created. The -r causes all subdirectories to be recursively included in the script created.

3• Contact the VOB Administrator and request that the utility script cvt_script be run on the script created in step 2.

The VOB Administrator is the only one who can run the cvt_script because it modifies the VOB.

4• At this time the user logs out from this workstation. The VOB Administrator completes the procedure.

The remaining steps are accomplished by the VOB Administrator.

5• The VOB Administrator logs into the AIT Sun workstation by typing **username** then press the **Enter** key.

Cursor moves to the **Password** field.

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

7• Invoke ClearCase by typing **xclearcase &** on the UNIX command line then press the **Enter** key.

The ClearCase **Transcript** screen is displayed as the View Tag Browser loads.
The ClearCase **View Tag Browser** screen is displayed listing available views.

8• To create a view for checking in the software change package, select a known View and press the **Enter** key. If you are using an existing view, select the desired existing view and proceed to step 14.

The File Browser window is displayed.

9 Select **File→Execute→Single Command.**

The String Browser window is displayed.

The prompt Enter shell command to run is displayed.

10• Invoke the make view command by typing **mkview [filename]** on the UNIX command line and press the **Enter** key.

The **tempdisp** window appears.

The **View [filename] Created Successfully** and the **Cache Updated for View [filename]** prompts are displayed.

11 Close the **tempdisp** window by clicking on the window and press the **Enter** key.

The **tempdisp** window closes.

12• Select the VOB where the software change package is to be imported then press the **Enter** key.

13• To create a subdirectory for the software change package in that VOB, which is a modification to the parent directory (for the VOB) the parent directory must be checked out by following the menu path **Version→Checkout→Reserved: no comment**.

In order to add new files to ClearCase, the directory in which the files are to be added must be checked out first.

ClearCase forces the checkout onto a maintenance branch to isolate the maintenance activity.

If someone else has already checked out the directory, permission to check out the directory is denied.

A separate shell window is displayed.

14• Start a shell process in a separate window by clicking on the shell icon button of the **File Browser** toolbar.

A separate shell window is displayed.

15 To run the script, type **cvt_script** then press the **Enter** key.

The VOB Administrator is the only person who can run the **cvt_script** because it modifies the VOB.

16• To check in the new directory, type into the directory input box of the **File Browser** screen: **path** [where **path** is the full path identification for the new directory (**directoryname**)], then press the **Enter** key. Then select **Versions→Checkin** from the menu.

17• To check in the parent directory (for the VOB), type into the directory input box of the **File Browser** screen: **VOBpath** (where **VOBpath** is the full path identification for the parent directory), then press the **Enter** key. Then select **Versions→Checkin** from the menu.

18• On the **File Browser** screen, follow menu path **File→Exit**.

The ClearCase Graphical User Interface session is closed.

26.4.6 Checking Out a File From ClearCase

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

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

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

Checking Out an Element/File from the Command Line

1• Log into one of the AIT Sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

3• At a UNIX prompt type **cleartool setview *ViewName***, then press the **Enter** key. The ***ViewName*** is the name if the user's view.

4 At a UNIX prompt type **cleartool checkout -nc *element*** then press the **Enter** key.
- The ***element*** is the name of the file or directory that is to be checked out.
- The **-nc** flag means "no comment" which will suppress the ClearCase prompting for a comment to be associated with the check out step.

Checking Out an Element/File from the File Screen Browser

---

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

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

3 Invoke ClearCase GUI by typing **xclearcase &** on the UNIX command line then press the **Enter** key.
The ClearCase **Transcript** screen is displayed as the View Tag Browser loads.
The ClearCase **View Tag Browser** screen is displayed listing available views.

4• To check out the directory where the controlled files were place, type into the directory input box of the **File Browser** screen: **path** [where **path** is the full path identification for the directory (**directoryname**)], then press the **Enter** key. Then select **Versions→Checkout** from the menu.

5• Select **File→Execute→Single Command.**
The String Browser window is displayed.
The prompt **Enter shell command to run** is displayed.

6 To determine editing privileges, type **ls -l,** then press the **Enter** key.
A prompt displaying read/write/execute privileges will be displayed. There will be three groupings:
- **User Group Others**
- **r=read, w=write, x=execute**

7• If you have editing/execute privileges, you can revise the contents of the file with any text editor.

8• To checkin a controlled file, select **Versions→Checkin** from the menu.
The file/directory will be checked in to ClearCase and the version will be updated.

26.4.7 Checking a Modified Element into ClearCase

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

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

1. A VOB exits and is mounted at a known UNIX directory.
2. A ClearCase view exists for the SSI&T operator.
3. The element or file has been checked out and modified.

4. The modified file is now in the user's directory on the VOB from which it was checked out.

26.4.7.1 Checking a Modified Element/File into ClearCase

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

26.5 Standards Checking of Science Software

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

26.5.1 Checking FORTRAN 77 ESDIS Standards Compliance

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

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

Assumptions:

1. The FORTRAN 77 science software source code is available, accessible, and has read permissions 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 If not already on an AIT Sun, log into one from your machine.
- Once logged onto proper Sun, remember to set the DISPLAY environmental variable to point to your X Window screen.

2• If required, at the UNIX prompt on the AIT Sun, type **cleartool setview *ViewName***, press **Return**.
- The ***ViewName*** is the name of a view allowing the FORTRAN 77 source files to be accessible.
- This step is only necessary if any of the FORTRAN 77 source files are in ClearCase (in the VOB under configuration management).

3 If your general environment setup does not include transparent access to the SSIT Manager GUI, then you need to set that up. One way to do it is as follows:
- Set up an alias, manually or from shell script, to set up preliminary environment. At UNIX prompt, type **alias do_buildrc "source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc"**
- Set up an alias, manually or through shell script, to invoke SSIT Manager. At UNIX prompt, type **alias do_ssit_man "/usr/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgr ConfigFile /usr/ecs/TS1/CUSTOM/cfg/EcDpAtMG.CFG ecs_mode TS1& "**

4• Set up the preliminary environment (do_buildrc). This only needs to be done once per session. Then, run SSIT Manager (do_ssit_man).
- Type **do_buildrc**
- Type **do_ssit_man**

5• Once the SSIT Manager comes up, the following steps need to be taken to invoke FORCHECK
- From the top menu bar, select **Tools.**
- From the Tools menu, select **Standards Checkers.**
- From the Standards Checkers menu, select **FORCHECK.**
- See Figure 26.8.5-2. for a screen snapshot of this step.

6 A separate FORCHECK window will now open.
- The user will be prompted 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 find hardcopy documentation for FORCHECK or to use the UNIX man facility and type *man*

forchk .

7 At the UNIX prompt on the AIT Sun, type **vi *FORCHECKoutput***, press **Return**.

- The ***FORCHECKoutput*** is the file name for the output file produced in step 6.
- The ***FORCHECKoutput*** file will contain any warnings, errors, and other messages from FORCHECK. A summary will be at the bottom of the file.
- Any text editor may be used for this procedure step.

8 At the UNIX prompt on the AIT Sun, type **vi *ListFile***, press **Return**.

- The ***ListFile*** is the file name for the list file specified at the FORCHECK prompt.
- The ***ListFile*** file will contain FORCHECK messages similar to the ***FORCHECKoutput*** file embedded in the source code listing.
  Any text editor may be used for this procedure step.

26.5.2 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.
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• From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the SPR SGI.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.

2• At the UNIX prompt on the SPR SGI, set up the proper environment for the compiler to be used by typing **source *ToolkitPathname* /bin/*sgiXX*/pgs-dev-env.csh .**

- ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version .
- The directory ***sgiXX*** should be replaced with **sgi32** or **sgi64** as appropriate for the specific compiler desired.
- For example, on the PVC platform p0spg01, type **source /data3/ecs/TS1/CUSTOM/daac_toolkit_f90/TOOLKIT/bin/sgi64/pgs-dev-env.csh .** This will set up the various environment parameters, such as PGSHOME, to enable the 64 bit version of the FORTRAN 90 compiler to be run.

3• If required, at the UNIX prompt on the SPR SGI, type **cleartool setview *ViewName***, press **Return**.

- The ***ViewName*** is the name of a view allowing the Fortran 90 source files to be accessible.
- This step is only necessary if any of the Fortran 90 source files are in ClearCase (in the VOB under configuration management).

4 At the UNIX prompt on the SPR SGI, type **cd *SrcPathname***, press **Return**.

- The ***SrcPathname*** is the full path name to the location of the Fortran 90 source files to be checked.
- The ***SrcPathname*** will be in the ClearCase VOB is the Fortran 90 source files are checked into ClearCase.

5• At the UNIX prompt on the SPR SGI, type **f90 -c -ansi *[*-I$PGSINC*] [*-I$HDFINC*] [[*-IOtherIncFiles*…] SourceFiles* >& *ReportFile***, press **Return**.

- The terms in square brackets (*[ ]*) are used to optionally specify locations of include and module (.mod) files. The **$PGSINC** already contains the SDP Toolkit include directory and **$HDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include or module directories.
- The ***SourceFiles*** is a list (space delimited) of Fortran 90 source files or a wildcard template (*e.g.* *.f90).
- The **>&** is a C shell construct that causes standard error (where the output from the Fortran 90 compiler normally emerges) to be redirected to a file.
- The ***ReportFile*** is the file name under which to save the results of the compile process.
- The **-c** flag causes only compilation (no linking).
- The **-ansi** flag enables ANSI checking.
- Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
- Do not use the **-I** option for include or module files that are in the standard directories or in the current directory.
- The makefile for the science software may contain the names of additional include files needed by the software.
- For example, type **f90 -c -I$PGSINC -I$HDFINC -I/ecs/modis/pge5/include/ *.f90 >& pge10.report**, press **Return**.

6 At the UNIX prompt on the SPR SGI, type **vi *ReportFile***, press **Return**.

- The ***ReportFile*** is the file name for the compilation results as produced in step 5.

- Any text editor may be used for this procedure step.

26.5.3 Checking for ESDIS Standards Compliance in C and C++

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

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

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

Assumptions:

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

- 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• From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the SPR SGI.
Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.

2 • At the UNIX prompt on the SPR SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, source **$PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.
The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version.
The ***sgiX*** refers to the appropriate processor. For example, type **source $PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.

3• If required, at the UNIX prompt on the SPR SGI, type **cleartool setview *ViewName***, press **Return**.
The ***ViewName*** is the name of a view allowing the C source files to be accessible.
This step is only necessary if any of the C source files are in ClearCase (in the VOB under configuration management).

4• At the UNIX prompt on the SPR SGI, type **cd *SrcPathname***, press **Return**.
The ***SrcPathname*** is the full path name to the location of the C source files to be checked.

The ***SrcPathname*** will be in the ClearCase VOB is the C source files are checked into ClearCase.

5• At the UNIX prompt on the SPR SGI, type **cc -c -ansi *[*-I$PGSINC*] [*-I$HDFINC*] [[*-IOtherIncFiles*…] SourceFiles* >& *ReportFile***, press **Return**.
The terms in square brackets (*[ ]*) are used to optionally specify locations of include and module (.mod) files. The **$PGSINC** already contains the SDP Toolkit include directory and **$HDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include directories.
The ***SourceFiles*** is a list (space delimited) of C source files or a wildcard template **(*e.g.* *.c or *.cpp).**
The **>&** is a C shell construct that causes standard error (where the output from the C compiler normally emerges) to be redirected to a file.
The ***ReportFile*** is the file name under which to save the results of the compile process.
The **-c** flag causes only compilation (no linking).
The **-ansi** flag enables ANSI checking.
Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
Do not use the **-I** option for include files that are in the standard directories (*e.g.* /usr/include) or in the current directory.
The makefile for the science software may contain the names of additional include files needed by the software.
For example, type **cc -c –ansi-I$PGSINC -I$HDFINC -I/ecs/modis/pge5/include/ *.c >& pge10.report**, press **Return**.

6• At the UNIX prompt on the SPR SGI, type **vi *ReportFile***, press **Return**.
The ***ReportFile*** is the file name for the compilation results as produced in step 5.
Any text editor may be used for this procedure step.

26.5.4 Prohibited Function Checker

The use of certain functions in the PGE is prohibited. The Prohibited Function Checker (Figure 26.5.4-1) is used to check C, FORTRAN 77 and FORTRAN 90 language source files for the occurrence of functions that are prohibited in the ECS DAAC production environment.

26.5.4.1 Checking for Prohibited Functions: Command-Line Version

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

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

Assumptions:

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

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

1• If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName***, press **Return**.
- The ***ViewName*** is the name of a view allowing the source files to be accessible.
- This step is only necessary if any of the source files are in ClearCase (in the VOB under configuration management).

2 At the UNIX prompt on the AIT Sun, type **cd *SrcPathname***, press **Return**.
- The ***SrcPathname*** is the full path name to the location of the source files to be checked.
- The ***SrcPathname*** will be in the ClearCase VOB if the source files are checked into ClearCase.
- The ***SrcPathname*** can contain other directories that contain source files and/or more directories. The Prohibited Function Checker will search out all source files in subdirectories recursively.

3• At the UNIX prompt on the AIT Sun, type **/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrBadFunc ConfigFile /data3/ecs/TS1/CUSTOM/cfg/EcDpAtBA.CFG *FilesOrDirectories* > *ResultsFile***, press **Return**.
- The ***FilesOrDirectories*** is a list of source file names or directory names of directories containing source files.
- The ***ResultsFile*** is the file name for the results that are output.
- For example, type **/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrBadFunc ConfigFile /data3/ecs/TS1/CUSTOM/cfg/EcDpAtBA.CFG main.c utils/ > myOutput**, press **Return**. Here, **main.c** is a source file and **utils/** is a directory that contains other source files.

4 At the UNIX prompt on the AIT Sun, type **vi *ResultsFile***, press **Return**.
- The ***ResultsFile*** is the file name for the output results as produced in step 3.
- Any text editor may be used for this procedure step.

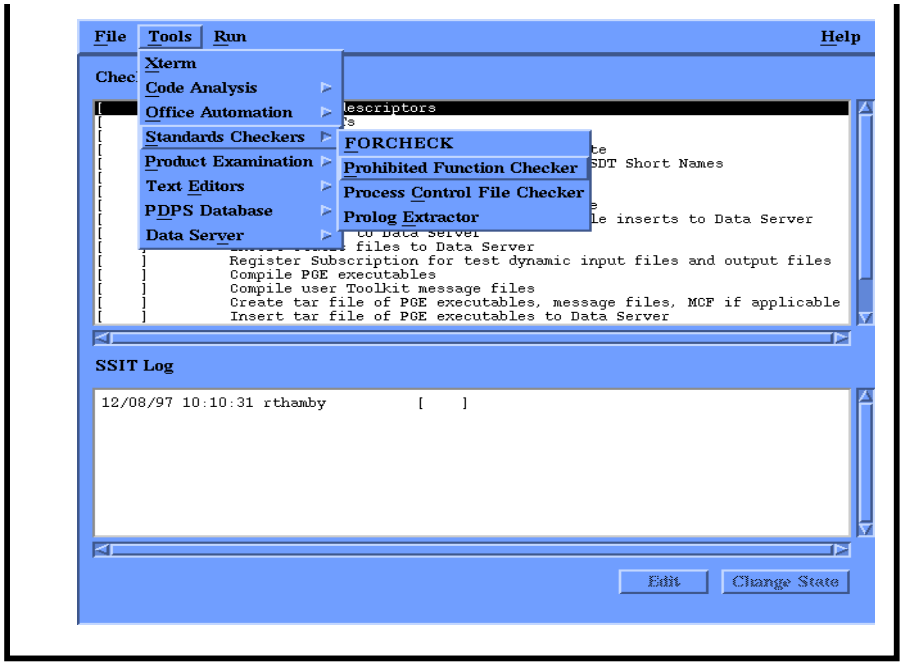


Figure 26.5.4-1. Invoking the Prohibited Function Checker

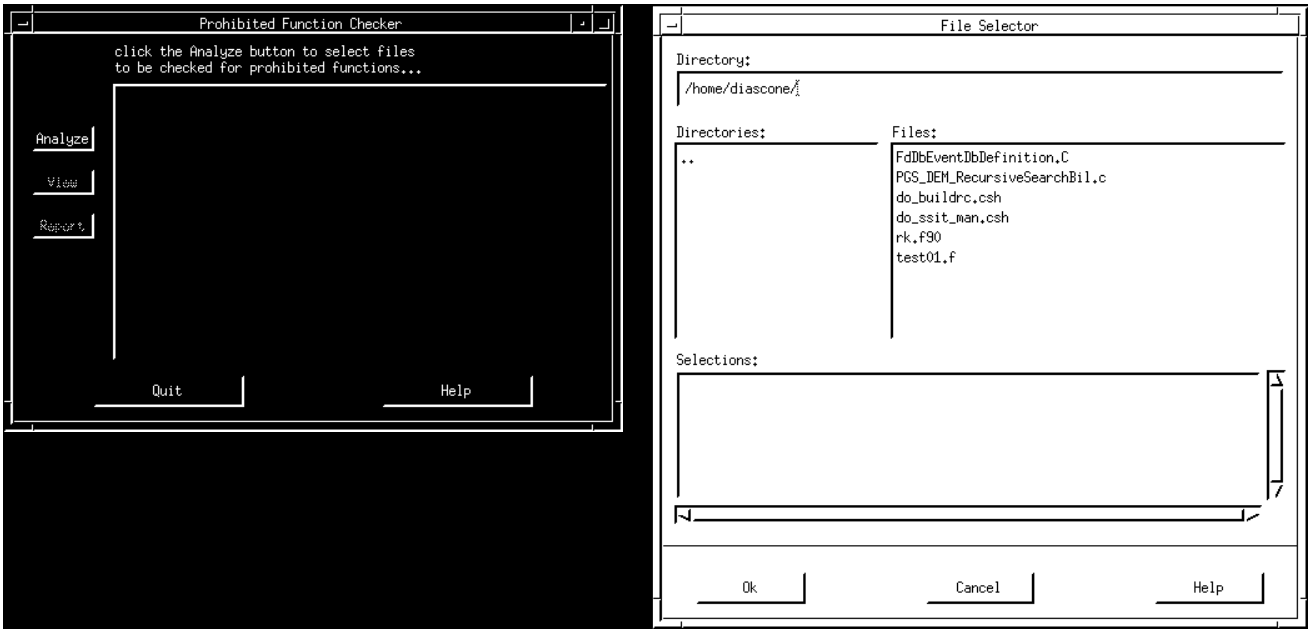


Figure 26.5.4-2. Starting Screen for Prohibited Function Checker

Figure. 26.5.4-3 File Selection Menu

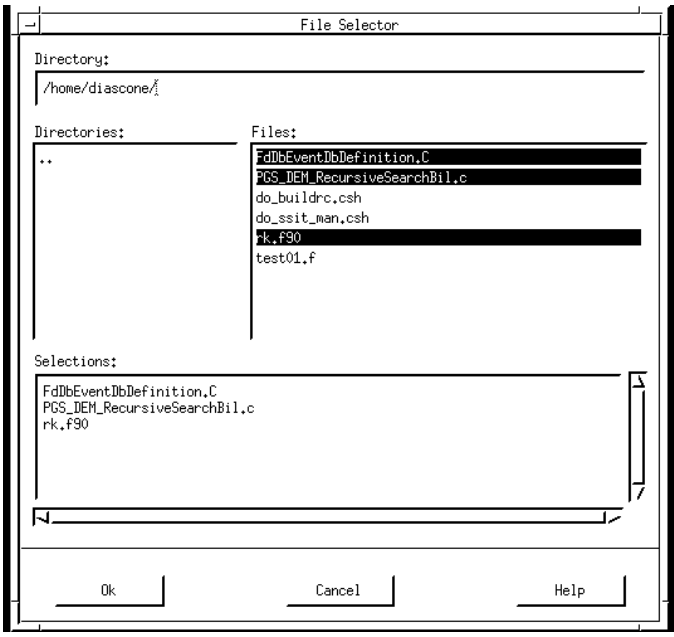


Figure 26.5.4-4. Selected Files

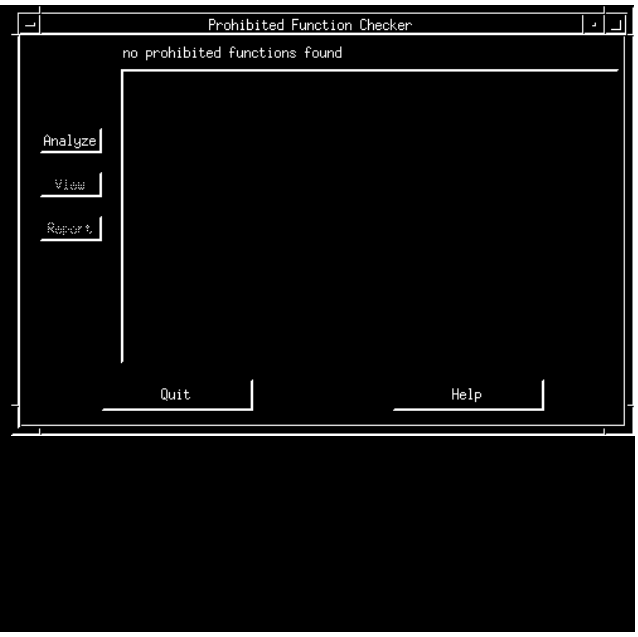


Figure 26.5.4-5. Results Screen

26.5.4.2 Prohibited Function Checker GUI Version

1 From the SSIT Manager, select **Tools → Standards Checkers → Prohibited Function Checker** from the menu.

The Prohibited Function Checker GUI will be displayed.

2 In the Prohibited Function Checker GUI, click on the **Analyze** button.

The File Selector GUI will be displayed.

3 Within the **Directories** subwindow, double click on the desired directory.

Repeat this step until the directory with the source files to be checked are displayed in the **Files** subwindow.

4 Within the **Files** subwindow, click on the source files to be checked. Each file clicked on will be highlighted.

To choose groups of contiguous files, hold down the left mouse button and drag the mouse.

To choose non-contiguous files, hold down the Control key while clicking on file names.

5 In the File Selector GUI, click on the **OK** button.

The File Selector GUI will disappear.

The files selected in step 5 will be displayed in the Prohibited Function Checker GUI window as they are being checked.

6 In the Prohibited Function Checker GUI, click on the **Report** button.

The **Report** GUI will be displayed.

For each file, a list of prohibited functions found will be displayed.

7 Optionally, click on the **Print** button or the **Save** button.

Choose **Save** to save the results to a file; choose **Print** to have the results printed on the default printer.

Choosing **Save** will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.

8 Optionally, in the Prohibited Function Checker GUI, highlight one of the source files listed. Then click on **View**.

The **Source Code** GUI will be displayed.

Occurrences of prohibited functions found in that source file will be highlighted.

Click on the **Next** button to bring into the window successive occurrences of prohibited functions (the **Next** button does not bring in the next source file).

Click on the **Done** button to close the **Source Code** GUI. Other source files may be examined similarly, one at a time.

9 In the Prohibited Function Checker GUI, click on the **Quit** button.

The Prohibited Function Checker GUI will disappear.

This ends the session.

26.5.5 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 26.5.5-1).

Table 26.5.5-1. 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 26.5.4-1 and 26.5.4-2.

26.5.6 Checking Process Control Files

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

26.5.6.1 Checking Process Control Files GUI

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

1. The SSIT Manager is running.
2. The Process Control File(s) are available, accessible, and have read permissions.

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

Checking Process Control Files GUI

1 From the SSIT Manager, select **Tools → Standards Checkers → Process Control File Checker** from the menu, see figure 26.5.6-1.

2

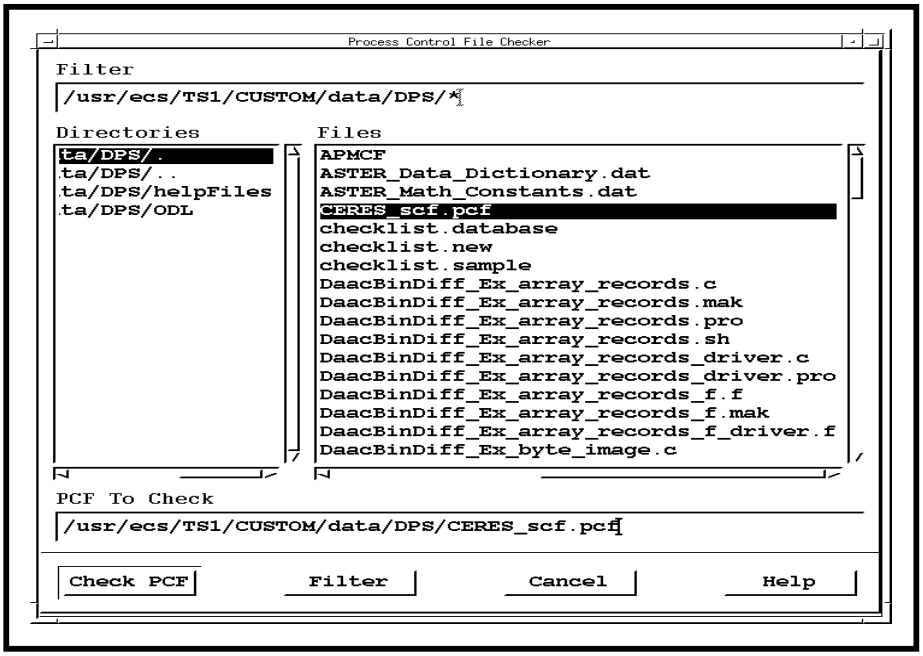


Figure 26.5.6-1. Process Control File Checker GUI

The Process Control File Checker GUI will be displayed.

2 In the **Directories** subwindow, double click on the desired directory.

Repeat this step until the directory with the PCF(s) to be checked are displayed in the Files window.

Use the **Filter** subwindow to limit which files are displayed.

3 Within the **Files** subwindow, click on the PCF to be checked**.**

The file clicked on will be highlighted.

Only one PCF can be checked at a time.

4 Click on the **Check PCF** button.

A GUI labeled **PCF Checker Results** will be displayed.

Results will be displayed in this window.

5 Optionally, click on the **Save** button or on the **Print** button.

Choose **Save** to save the results to a file; choose **Print** to have the results printed on the default printer.

Choosing **Save** will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.

Choosing **Print** and then clicking on the **OK** button will send the results to the default printer.

6 Click on the **Check Another** button or on the **Quit** button.

Choosing **Check Another** allows another PCF to be checked. Repeat steps 2 through 5.

Choosing **Quit** causes the Process Control File Checker GUI to disappear and ends the session.

26.5.6.2 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 pccheck.sh**, press **Return**. If a path is displayed, then the directory is in your path. On the PVC Sun platform **p0ais01,** the pathname for the command is **/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh**. In this case, you will have to set a ClearCase view to access that area.

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

---

1 If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName***, press **Return**.

- The ***ViewName*** is the name of a view allowing the Process Control File(s) to be accessible.
- This step is only necessary if any of the Process Control Files are in ClearCase (in the VOB under configuration management).

2 At the UNIX prompt on AIT Sun, type **cd *PCFpathname***, press **Return**.

- The ***PCFpathname*** is the full path name to the location of the Process Control File(s) to be checked.
- The ***PCFpathname*** will be in the ClearCase VOB if the Process Control Files are checked into ClearCase.

3 At the UNIX prompt on an AIT Sun, type **/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh -i *PCFfilename* > *ResultsFile***, press **Return**.

- The ***PCFfilename*** is the full path name (directory and file name) to the Process Control File to check.
- The ***ResultsFile*** is the file name for the results that are output.
- The PCF Checker is also available on the SPR SGI machines. The easiest way to access it is to set a SDP Toolkit environment (any will do for purposes here, see Section 9.2) and type **$PGSBIN/pccheck.sh -i *PCFfilename* > *ResultsFile***, press **Return**.

4 At the UNIX prompt on the SPR SGI, **p0spg01**, type **vi *ResultsFile***, press **Return**.

- The ***ResultsFile*** is the file name for the output results as produced in step 4.
- Any text editor may be used for this procedure step.

26.5.7 Extracting Prologs

The Project standards and guidelines are contained in the latest version of 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.

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

1. The SSIT Manager is running.
2. Prologs are assumed to be delimited by particular delimiters depending on the language type. Delimiters are listed in the table below:

Prolog Delimiters

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

The Prolog Extractor recognizes the language type of the file by its file name extension. The table below lists assumed file name extensions:

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

26.5.7.1 Extracting Prologs

The Prolog Extractor can be started from the UNIX prompt. To do this, at the UNIX prompt on the AIT Sun, type **/data3/ecs/TS1/CUSTOM/bin/DPS/EcDpAtMgrPrologs**, press **Return**

or

1 From the SSIT Manager, select the **Tools → Standards Checkers → Prolog Extractor** from the menu.

An xterm will be displayed on the AIT Sun.

Select the default ConfigFile. The output goes to a file called Prologs.txt in the directory from which the SSIT Manager was started.

The Prologs.txt file can be viewed by changing directories to the SSIT Manager directory and invoking a text editor. The file may also be sent to a printer.

2 At the **Files(S)? (-h help)** prompt, type in the file names and/or directory names containing the files.

Separate items with spaces.

The contents of the directory will be search recursively for files with valid file name extensions.

Use **./** to indicate current directory.

The time needed for the Prolog Extractor could be very long for large numbers of files and directories.

When extraction is complete, the message **Output written to file: ./prologs.txt** will be displayed.

3 At the program prompt **Hit Enter for another, "q <Enter>" to quit:** , press **Enter** to repeat process with another set of source files or type **q** and press **Enter** to quit.

- The xterm will disappear.

4 At a UNIX prompt on the AIT Sun, type **vi prologs.txt,** then press the **Enter** key.

The extracted prologs file, named **prologs.txt,** will be brought into the editor.

The default location of the **prologs.txt** file is the directory from which the SSIT Manager was invoked.

5 Once the extracted prologs file has been examined, exit the editor.

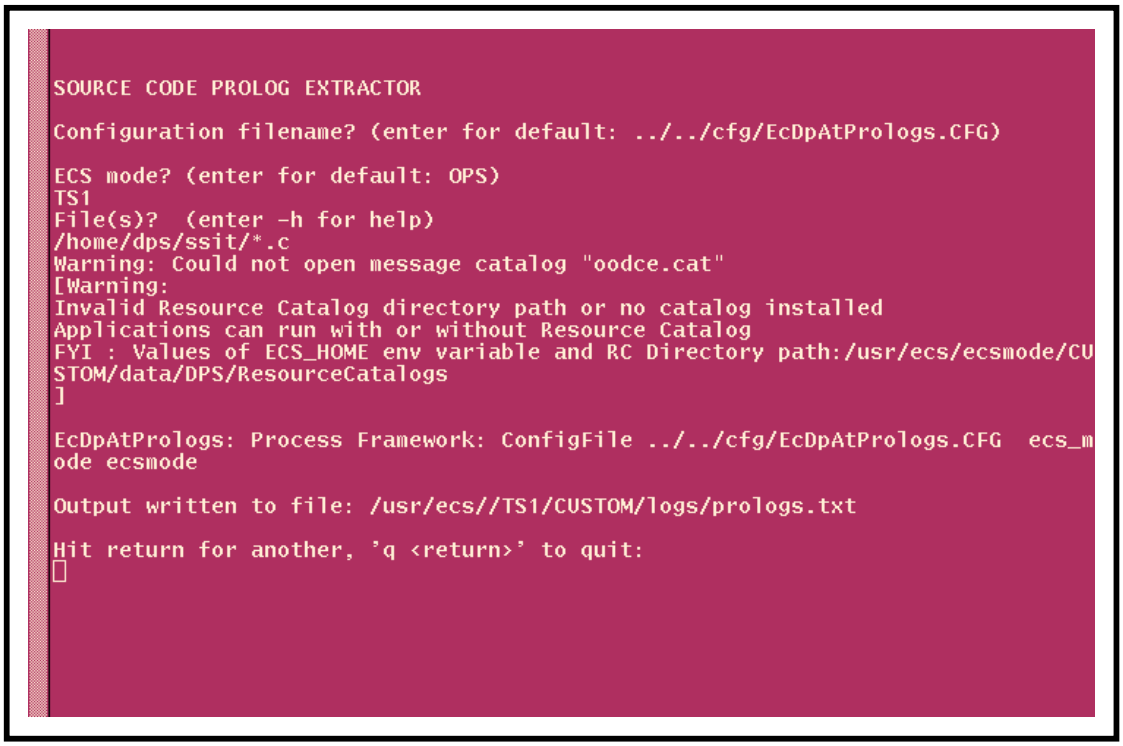


Figure 26.5.7-1. Prolog Extractor Sample Run.

26.6 Compiling and Linking Science Software

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

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

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

26.6.1 Updating the Process Control 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 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the SGI.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.

2 If required, at the UNIX prompt on the SGI, type **cleartool setview *ViewName***, press **Return**.

- The ***ViewName*** is the name of a view allowing the PCF to be accessible.
- This step is only necessary if the PCF is in ClearCase (in the VOB under configuration management).

3 At the UNIX prompt on the Sun or on the SGI, type **cd *PCFpathname***, press **Return**.

- The ***PCFpathname*** is the full path name to the location of the PCF. This location will be in the ClearCase VOB if the PCF is under configuration management.

4 At the UNIX prompt on the Sun or on the SGI, type **cleartool checkout -nc *PCFfilename***, press **Return**.

- The ***PCFfilename*** is the file name of the PCF that is to be checked out (and later modified). The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check out step.

5 Run the Process Control File Checker on the delivered PCF.

- This will verify that the delivered PCF is correct before editing.

6 At a UNIX prompt on the Sun, type **vi *PCFfilename***, press **Return**.

- The ***PCFfilename*** is the file name of the PCF to update.
- Any text editor may be used such as *emacs*. For example, **emacs AST02.pcf**, press **Return**.

7 In the file, make changes to the default directories specified in each section of the PCF. All path names specified in the PCF must exist on the SGI.

- Each section begins with a line consisting of a **?** in the first column followed by a label:

```
? PRODUCT INPUT FILES
? PRODUCT OUTPUT FILES
? SUPPORT INPUT FILES
? SUPPORT OUTPUT FILES
? INTERMEDIATE INPUT
? INTERMEDIATE OUTPUT
? TEMPORARY I/O
```

- Each of the above section heading lines will then be followed (not necessarily immediately; there may be comment lines) by a line that begins with a **!** in the first column. These lines specify the default path names for each section.
  - If the line reads:
    ```
    ! ~/runtime
    ```
    leave it unchanged. The tilde (~) is a symbol that represents $PGSHOME.
  - If another path name is listed instead, it will probably need to be changed to a path name that exists at the DAAC on the SGI. When specifying a path name, use an absolute path name, not a relative path name.

8 In the file, look for science software specific entries in each section and make changes to the path names (field 3) as necessary. All path names specified in the PCF must exist on the SGI.

- The science software specific entries will have logical IDs (first field) *outside* of the range 10,000 to 10,999.
- Where necessary, replace the path names in the third field of each entry with the path names appropriate to the DAAC environment.
- Do not alter file entries that are used by the SDP Toolkit itself. These have logical IDs *in* the range 10,000 to 10,999.
- For example, if the following entry was found in the PCF:
  ```
  100|A.granule|/MODIS/run/input||||1
  ```
  change /MODIS/run/input to the appropriate path name in the DAAC where the file A.granule is stored.
- When specifying a path name, use an absolute path name, not a relative path name.
- Do not include the file name with the path name. The file name belongs in field 2 by itself.

9 In the file, verify that the SUPPORT OUTPUT FILES section contains an entry to the shared memory pointer file.

- Look for the entry:
  ```
  10111|ShmMem|~/runtime||||1
  ```
  The third field may be blank; this will work too.
- If this entry is not within this section, add it.

10 Once changes have been made to the PCF, save the changes and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
- For other editors, refer to that editor's documentation.

11 Again, run the Process Control File Checker on the PCF.

12 If the PCF had been checked out of ClearCase, at the UNIX prompt on the SGI, type **cleartool checkin -nc *PCFfilename***, press **Return**.

- The ***PCFfilename*** is the file name of the modified PCF. The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check in step.

26.6.2 Setting up a SDP Toolkit Environment

The purpose of the SDP Toolkit is to allow science software to be developed for ECS at independent SCFs and to provide:

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

The latest versions of Toolkit and accompaning Metadata tools can be found on the WWW.

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

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

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

The SDP Toolkit is divided into two groups of tools:

26.6.2.1 Mandatory Tools

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

26.6.2.2 Optional Tools

- Ancillary Data Access - provides access to NMC data and Digital Elevation (DEM) data.
- Celestial Body Position - locates the sun, moon and the planets.
- Coordinate System Conversion - coordinate conversions between celestial reference.
- Constant and Unit Conversion - physical constants and unit conversions.
- IMSL - mathematical and statistical support.

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

TOOL TITLE

NAME:	Procedure or routine name
SYNOPSIS:	C: C language call
FORTRAN:	FORTRAN77 or Fortran90 language call
DESCRIPTION:	Cursory description of routine usage
INPUTS:	List and description of data files and parameters input to the routine
OUTPUTS:	List and description of data files and parameters output from the routine
ENTERS:	List of returned parameters indicating success, failure, etc.
EXAMPLES:	Example usage of routine
NOTES:	Detailed information about usage and assumptions
REQUIREMENTS:	Requirements from PGS Toolkit Specification, Oct. 93 which the routine satisfies

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

There are several versions of the SCF/DAAC SDP Toolkit installed on the SGI Power Challenges at the DAACs for the Release 4 system. The toolkit versions at the DAACs differ according to:

Object Type - The operating system on the SGI Power Challenges on Release 4 is IRIX 6.2, a 64-bit operating system. To be backward compatible, the SGI operating system will allow new 64-bit and 32-bit objects to be built as well as the older 32-bit machines. Each of these object types are designated by placing a cc flag on the command line to enable a particular mode with the SGI C compiler.

New 64-bit: cc flag = -64
New 32-bit: cc flag = -n32
Old 32-bit: cc flag = -32 (SCF's only)

Library Type - The SDP Toolkit uses different libraries depending upon whether FORTRAN 77 or FORTRAN 90 source code is being linked. If C source code is to be linked, then either language version of the library will work.

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

The following Table summarizes the available SDP Toolkits used by the SGI science processors.

Table 26.6.2-1. SDP Toolkits used by the SGI science processors

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

Table 4. SDP Toolkits used by the SGI science processors.

$CUSTOM_HOME is an environment variable set to: **/usr/ecs/*MODE*/CUSTOM**
$PGSHOME is an environment variable set to: **usr/ecs/*MODE*/CUSTOM/TOOLKIT/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. For running a PGE in the fully functional DAAC environment, the DAAC version should be used.

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

If both FORTRAN 77 and Fortran 90 are being used, the procedure becomes more complex. Under such circumstances, refer to document *333-CD-510-002, Release 5B SDP Toolkit Users Guide for the ECS Project, April 2000* .

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

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

26.6.2.3 Setting Up the SDP Toolkit Environment

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 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**.

- The ***ToolkitPathname*** is the home directory of the particular SDP Toolkit version being used. Refer to Table 26.6.2-1. Note that the setting of PGSHOME shown in this table may differ in your local DAAC.
- Korn shell users, type **PGSHOME=*ToolkitPathname*; export PGSHOME**, press **Return**.

2 At the UNIX prompt on the SGI, type **source $PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.

- The ***sgiX*** is one of: **sgi** for 32-bit version of the Toolkit or **sgi64** for 64-bit version of the Toolkit. Refer to the last column of Table 26.6.2-1. for path names to the file to source.
- Korn shell users, type **. $PGSHOME/bin/*sgiX*/pgs-dev-env.ksh**, press **Return** (note the "dot" and then space at the beginning of this command).

3 This step is optional. Edit the file $HOME/.cshrc and add the line **alias *aliasname* 'setenv PGSHOME *ToolkitPathname*; source $PGSHOME/bin/*sgiX*/pgs-dev-env.csh; echo "*textmessage*" '**.

- The ***aliasname*** is the name of the alias. For example, to set up an environment for the DAAC version of the Toolkit for FORTRAN 77 (or C), you might use **DAACf77** as an ***aliasname***.
- The ***ToolkitPathname*** is the home directory of the particular SDP Toolkit version being used. Refer to Table 26.6.2-1. Note that the setting of PGSHOME shown in this table may differ in your local DAAC.
- The ***sgiX*** is one of: **sgi** for 32-bit version of the Toolkit or **sgi64** for 64-bit version of the Toolkit.
- The ***textmessage*** is a message that will be echoed to the screen signifying that a new Toolkit environment has been set up. It must be enclosed within
- double quotes ("). An example may be, **"DAAC F77 Toolkit environment is now set."**
- A complete example (it should be all on one line in the .cshrc file):

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

26.6.2.4 An example of Compile procedures used to produce a PGE.exe :

1 Setup for PGE07:

```
/home/emcleod/MODIS/STORE/PGE07/MOD_PR10/source
rm MOD_PR10.exe
rm *.o
setenv PGSHOME /usr/ecs/OPS/CUSTOM/TOOLKIT/sgi32_daac_f77/
source $PGSHOME/bin/sgi32/pgs-dev-env.csh
source
/home/emcleod/MODIS/STORE/PGE07/MOD_PR10/source/MODIS_setup.csh.pge07
alias
n32_f77
env
make -f MOD_PR10.mk &
ls -l *exe
setenv PGS_PC_INFO_FILE
```

```
/home/emcleod/MODIS/STORE/PGE07/MOD_PR10/source/MOD_PR10.pcf
ls
MOD_PR10.exe &
confirm execution when done by looking at   file :  vi
MOD_PR10_ClopyL1BmetaToSnow.c
see if job is running:  ps -u emcleod  "time updating for MOD_PR10"
p0spg01{emcleod}88: ps -u emcleod
   PID TTY     TIME CMD
   267 ?       3:13 biod
 25825 pts/11  0:01 csh
 21994 pts/10  0:01 csh
 23215 pts/16  0:00 csh
 26242 pts/10  0:07 MOD_PR10.
 26089 pts/11  0:01 xedit
 26318 pts/10  0:00 ps
p0spg01{emcleod}105: pwd
/tmp_mnt/home/emcleod/MODIS/STORE/PGE07/MOD_PR10/source
p0spg01{emcleod}106: ls
MODIS_setup.csh.pge07        MOD_PR10_CopyL1BmetaToSnow.c
MODIS_setup_OPS              MOD_PR10_CopyL1BmetaToSnow.o
MOD_PR10.exe                 MOD_PR10_MakeMeta.c
MOD_PR10.h                   MOD_PR10_MakeMeta.o
MOD_PR10.mcf                 MOD_PR10_Process_Cloud.c
MOD_PR10.mk                  MOD_PR10_Process_Cloud.o
MOD_PR10.pcf                 MOD_PR10_Process_GEO.c
MOD_PR10_AAmain.c            MOD_PR10_Process_GEO.o
MOD_PR10_AAmain.o            MOD_PR10_Process_L1B.c
MOD_PR10_Compute_Snow.c      MOD_PR10_Process_L1B.o
MOD_PR10_Compute_Snow.o      MOD_PR10_Process_SnowFile.c
MOD_PR10_CopyGEOmetaToSnow.c  MOD_PR10_Process_SnowFile.o
MOD_PR10_CopyGEOmetaToSnow.o  compile_smf.csh
p0spg01{emcleod}107:
```

26.6.2.5 Example of a PGE Executables Tar File Insertion Script

This example was produced in Drop 4 and is provided for review only. Go to the section Placing the Science Software Executable (SSEP) on the Data Server which includes the Insertion of a PGE Tar file..

```
Configuration filename?  (enter for default:
../../cfg/EcDpAtInsertExeTarFile.CFG)
ECS Mode of operations?  (enter for default: OPS)
Name of PGE?  (enter for default: PGE07)
Science software version of PGE?  (enter for default: 2)
Staged filename to insert (including FULL path)?  (enter for default:
/home/emcleod/SSEP/PGE07.tar)
Associated ASCII metadata filename to insert (including FULL path)?  (enter for
```

```
defaul /home/emcleod/SSEP/PGE07.tar.met)
Top level shell filename within tar file?  (enter for default: PGE07.csh)
PGE07.csh
Warning: Could not open message catalog "oodce.cat"
/usr/ecs//OPS/CUSTOM/bin/DPS/EcDpAtInsertExeTarFile: Process Framework:
ConfigFile ..//cfg/EcDpAtInsertExeTarFile.CFG  ecs_mode OPS
Performing INSERT.....
Retrieved from IOS for ESDT = PGEEXE the DSS UR =
UR:15:DsShSciServerUR:13:[MDC:DSSDSR
Trying to make a request to [MDC:DSSDSRV]
Trying to make a request to [MDC:DSSDSRV]
Insert to Data Server and PDPS database update successful for:
   PGE name = 'PGE07'
   Ssw version = '2'
   ESDT = 'PGEEXE'
   ESDT Version = "001"
   staged file = '/home/emcleod/SSEP/PGE07.tar'
   metadata file = '/home/emcleod/SSEP/PGE07.tar.met'
   Top level shell name = 'PGE07.csh'
Inserted at UR:
'UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[MDC:DSSDSRV]:14:LM:PGEEXE:94'
Hit return to run again, 'q <return>' to quit:
```

26.6.3 Compiling Status Message Facility (SMF) Files

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

Science software making use of the SMF need particular header (include) files when being built and also need particular runtime message files when being run. Both the header and message files are produced by running a SMF "compiler" on a message text file. These message text files should be part of the science software delivery to the DAAC. They typically have a .t file name extension.

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

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

Assumptions:

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 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the SGI.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **SGI.**
- It is recommended that this procedure begin within a new command shell on the SGI.

2 If required, at the UNIX prompt on the SGI, type **cleartool setview *ViewName***, press **Return**.

- The ***ViewName*** is the name of a view allowing the SMF files to be accessible.
- This step is only necessary if any of the SMF files are in ClearCase (in the VOB under configuration management).

3 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, source **$PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.

- The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version.
- The ***sgiX*** refers to the appropriate processor. For example, type **source $PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.

4 At the UNIX prompt on the SGI, type **cd *pathname***, press **Return**.

- The ***pathname*** is the full path name to the directory containing the SMF text files.
- The SMF text files will typically have .t file name extensions.

5 At the UNIX prompt on the SGI, type **smfcompile -f *textfile*.t -lang** , press **Return**.

- The ***-lang*** is a flag that indicates for what language to compile. This flag can be one of **-c** to produce C header files and **-f77** to produce FORTRAN 77 include files. The default is for C include files. For example, type **smfcompile -f77 PGS_MODIS_39123.t**, press **Return**.
- The ***textfile*** is the file name of the SMF text file delivered with the science software.
- The SMF text files will typically have .t file name extensions.
- File names for SMF text files usually have the "seed" value used by the file as part of its file name (*e.g.* PGS_MODIS_39123.t where 39123 is the seed number).
- Only one such SMF text file can be compiled at a time; wildcards cannot be used.
- The SMF compiler may be run with the additional flags **-r** and **-i** as in, **smfcompile -f *textfile*.t -r -i**. The **-r** automatically places the runtime message file in the directory given by the environment variable PGSMSG. The **-i** automatically places the include file in the directory given by the environment variable PGSINC. For example, type **smfcompile -f90 -r -i -f PGS_MODIS_39123.t**, press **Return**. Note that the **-f** flag must always be immediately followed by the name of the text file.

6 If necessary, at the UNIX prompt on the SGI, type **mv *IncludeFilename* $PGSINC**, press **Return**. Then, type **mv *RuntimeFilename* $PGSMSG**, press **Return**.

- This step is only required if either the **-r** or the **-i** flag were not used in step 5.
- The ***IncludeFilename*** is the name of the include file created in step 5.
- The ***RuntimeFilename*** is the name of the runtime message file created in step 5.

- For example, type **mv PGS_MODIS_39123.h $PGSINC**, press **Return**. And then type, **mv PGS_MODIS_39123 $PGSMSG**, press **Return**.

26.6.4 Building Science Software with the SCF Version of the SDP Toolkit

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

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

Assumptions:

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

To build science software with the SCF 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 ***Systems Description*** document and the ***Operations Manual*.** Both of these or their equivalent should be in the delivery.
- Typically, there will be "readme" files accompanying each PGE in the directory structure, perhaps in a doc directory.
- Text files (ASCII) may be viewed with the UNIX command, ***more*** or with the ***vi* editor.**
- PostScript documents may be viewed with ***ghostview,*** which is accessible via the SSIT Manager.

- PDF formatted documents may be viewed with ***acroread,*** the Acrobat Reader, also accessible via the SSIT Manager.
- Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows™ 3.1 emulator. The MS Windows emulator may be accessed from the SSIT Manager.

2 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then **telnet** to the **SGI.**

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **SGI.**
- It is recommended that this procedure begin within a new command shell on the SGI.

3 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, source **$PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.

- The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version, in this case, an SCF version.
- The ***sgiX*** refers to the appropriate processor. For example, type **source $PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.

4 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview *ViewName***, press **Return**. Then, **cd *pathname***, press **Return**. And **cleartool checkout -nc *makefile***, press **Return**.

- The ***ViewName*** is the name of a view allowing the make files to be accessible.
- The ***pathname*** is the full path name of the directory (in the **VOB**) where the make file has been checked in.
- The ***makefile*** is the name of the make file to examine and possibly modify.
- This step is only necessary if any of the make files (or build scripts) are in **ClearCase** (in the **VOB** under configuration management).

5 Examine and alter (if necessary) any make files using any text editor **(*vi, emacs*).**

- There may be several make files for a particular **PGE**.
- Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.
- The Toolkit set up (from step 3) will set many environment variables which can be used in the make files. To see the current environment variable settings, at the UNIX prompt on the **SGI**, type **env**, press **Return**.

6 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building. Example: **p0spg01{cmshared}>/usr/ecs/TS1/CUSTOM/ssit/PGE32/message**.

7 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.

8 If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc *filename***, press **Return**.

- The ***filename*** is the file name of the **executable, object file,** or **make file** to be checked out of ClearCase. The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- Note that checking in executable or object files is ***not*** recommended in the first place.

9 Build the software in accordance with instructions delivered.

- Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
- Choose the most appropriate optimization/debugger flag. During testing, the **"-g"** is often used. This results in larger and slower executables, but assists in debugging. For production, the **"-O"** flag may be used to optimize execution time. Variants of the **"-g"** and **"-O"** flags may be incompatible.

10 If necessary, at the UNIX prompt on the SGI, type **cleartool checkin *filename* -nc**, press **Return**.

- The ***filename*** is the file name of the executable, object file, or make file to be checked into ClearCase. The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
- Note that checking in executable or object files is ***not*** recommended.

26.6.5 Building Science Software with the DAAC Version of the SDP Toolkit

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

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

Assumptions:

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

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

---

1 Read all instructional material supplied with the science software delivery. Such material should be the primary source of information on how to build the science software.

- Read the ***Systems Description*** document and the ***Operations Manual***. Both of these or their equivalent should be in the delivery.
- Typically, there will be **"readme"** files accompanying each PGE in the directory structure, perhaps in a doc directory.
- Text files (ASCII) may be viewed with the UNIX command, ***more*** or with the ***vi* editor.**
- PostScript documents may be viewed with ***ghostview,*** which is accessible via the SSIT Manager.
- PDF formatted documents may be viewed with ***acroread***, the Acrobat Reader, also accessible via the SSIT Manager.
- Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows™ 3.1 emulator. The MS Windows emulator may be accessed from the SSIT Manager.

2 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then **telnet** to the **SGI**.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.
- It is recommended that this procedure begin within a new command shell on the SGI.

3 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, source **$PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.

- The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version, in this case, a DAAC version.
- The ***sgiX*** refers to the appropriate processor. For example, type **source $PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.

4 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview *ViewName***, press **Return**. Then, **cd *pathname***, press **Return**. And **cleartool checkout -nc *makefile***, press **Return**.

- The ***ViewName*** is the name of a view allowing the make files to be accessible.
- The ***pathname*** is the full path name of the directory (in the VOB) where the make file has been checked in.
- The ***makefile*** is the name of the make file to examine and possibly modify.
- This step is only necessary if any of the make files (or build scripts) are in **ClearCase (**in the **VOB** under configuration management).

5 Examine and alter (if necessary) any make files using any text editor (*vi, emacs*). If the software had already been built and tested with the SCF version of the SDP Toolkit, this step may be unnecessary.

- There may be several make files for a particular PGE.
- Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.

- The Toolkit set up (from step 3) will set many environment variables which can be used in the **make** files. To see the current environment variable settings, at the UNIX prompt on the **SGI**, type **env**, press **Return**.

6 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building.

7 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.

8 If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc *filename***, press **Return**.

- The ***filename*** is the file name of the executable, object file, or make file to be checked out of ClearCase. The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- Note that checking in executable or object files is ***not*** recommended in the first place.

9 Build the software in accordance with instructions delivered.

- Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
- Choose the most appropriate optimization/debugger flag. During testing, the **"-g"** is often used. This results in larger and slower executables, but assists in debugging. For production, the **"-O"** flag may be used to optimize execution time. Variants of the **"-g"** and **"-O"** flags may be incompatible.

10 If necessary, at the UNIX prompt on the **SGI,** type **cleartool checkin *filename* -nc**, press **Return**.

- The ***filename*** is the file name of the executable, object file, or make file to be checked into ClearCase. The **-nc** flag means "no comment"; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
- Note that checking in executable or object files is ***not*** recommended.

26.6.6 PGE Checkout

The following Servers/Services must be up and operational:

NONE.

The following must have occurred between those Servers/Services:

NONE.

What the user must do before trying SSIT functionality:

In normal SSIT (at the DAACs) the DAP would be untared, and the source code recompiled and tested.

What must be done via SSIT tools:

Since SSIT is just a calibration of various tools, there is no specific order for which they must be run. All tools can be started from the SSIT Manager and can be executed on their own.

For SparcWorks (for code analysis and debugging), choose Tools menu and then Code Analysis submenu. See SparcWorks manuals for SparcWorks operation.
For various office tools, choose Tools menu and the Office Automation submenu. Choose from MS Windows (a simulator to allow the user to run Windows programs), Ghostview (a viewer), Netscape (for web access), Acrobat (for document viewing), and DDTS (for problem reporting).
For Standards checkers, choose Tools and then the Standards Checkers submenu.

FORCHECK is a COTS Fortran language checking program.

The Prohibited Function Checker will examine source code for functions that are not permitted On Prohibited Function Gui, choose Analyze to select files to examine. Hit the Ok button once selections are made and a message at the top of the Gui will indicate if prohibited functions have been found. If prohibited functions HAVE been found, use the View button to view the source code with the prohibited call. The Help button gives further information on how to work the Gui.

The Process Control File Checker examines selected PCFs and highlights any errors. The Process Control File Gui allows the user to work through the directory structure on the local machine and select PCFs to be checked. Click the Check PCF button to check a selected PCF. Again, the Help button provides more information.

26.7 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.

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

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

26.7.1 Setting Up the Environment for Running the PGE

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

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

Assumptions:

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

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

---

1 Open an **xterm** window and then **telnet** to the **SPR SGI**.

- It is recommended that this procedure begin within a new command shell on the SPR SGI.

2 At the UNIX prompt on the SPR SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, **source $PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.

- The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version, in this case, an SCF version. For example, **p0spg01{cmshared}>setenv PGSHOME /usr/ecs/TS1/CUSTOM/TOOLKIT/toolkit/bin/sgi64_daac_f90.**
- The ***sgiX*** refers to the appropriate processor. For example, **p0spg01{cmshared}>source $PGSHOME/pgs_dev_env.csh**, press **Return**.

3 At the UNIX prompt on the SPR SGI, type **setenv PGS_PC_INFO_FILE *PCFpathname*/*PCFfilename***, press **Return**.

- The ***PCFpathname*** is the full path name to the location of the Process Control File (PCF) to be associated with this PGE.
- The ***PCFfilename*** is the file name of the PCF.
- For example, **setenv PGS_PC_INFO_FILE /usr/ecs/TS1/CUSTOM/ssit/PGE32/PCF/PGE32.pcf**, press **Return**.

4 This step is optional. At the UNIX prompt on the SPR SGI, type **rm *LogPathname*/*LogFilename***, press **Return**.

- The ***LogPathname*** is the full path name to the location of the PGE log files for this PGE.
- The ***LogFilename*** is the file name of the PGE log file to remove from a previous run of the same PGE. PGE log files can be Status, User, or Report.
- The ***LogFilename*** may use wildcard characters to remove all of the log files at the same time.
- This step is optional. If log files from a previous run of the same PGE are not removed, they will be appended with the information from the current run.
- The environment will then be set up. Continue on to Section 12.2.

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.

- For example, for a PGE requiring IMSL, at the UNIX prompt on the SPR SGI, type **source /usr/ecs/<mode>/COTS/imsl/vni/ipt/bin/iptsetup.cs**, press **Return**.
- For some PGEs, the environment variables to be set will be specified in the documentation or the files to source will be supplied in the delivery. Refer to documentation included in the delivery.

26.7.2 Running and Profiling the PGE

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

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

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

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

Assumptions:

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

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

---

1 At the UNIX prompt on the SPR SGI in the window containing the set up environment, type **cd *PGEbinPathname***, press **Return**.

- The ***PGEbinPathname*** is the full path name of the directory containing the built PGE binary executable. For example, **p0spg01{cmshared}cd /usr/ecs/TS1/CUSTOM/ssit/PGE32/bin/**, press **Return**.

2 At the UNIX prompt on the SPR SGI, type **/usr/ecs/<mode>/CUSTOM/bin/DPS/E*cDpPrR*usage *PGE.exe* >& *ResultsOut***, press **Return**.

- The ***PGE.exe*** is the name given to the PGE binary executable.
- The ***ResultsOut*** is the file name in which to capture the profiling results as well as any messages from standard output (stdout) and standard error (stderr) that may be produced by the running PGE. Note that PGEs should *not* write to stdout or stderr.
- For example:
  p0spg01{cmshared}>/usr/ecs/TS1/CUSTOM/bin/DPS/EcDpPrRusage run_PGE32.exe >& ../logs/PGE32.profile.out
- The **EcDpPrRusage** is the profiling program that outputs information about the runtime behavior of the PGE.
- Depending upon the PGE, it may take some time before the UNIX prompt returns.

3 At the UNIX prompt on the SPR SGI, type **echo $status**, press **Return**.

- The **$status** is an environment variable that stores the exit status of the previous program run, in this case, the PGE.
- A status of zero indicates success; a status of non-zero indicates an error of some kind.
- The meaning of a non-zero exit status should be documented and included with the DAPs.
- This command must be run ***immediately*** after the **EcDpPrRusage** command.

4 At the UNIX prompt on the SPR SGI, type **vi *ResultsOut***, press **Return**.

- The ***ResultsOut*** is the file name under which the profiling output was saved. Other output of the PGE may also be in this file.
- The **EcDpPrRusage** results may then be recorded and used when the PGE is registered in the PDPS.
- Any text editor/viewer may be used.

Sample of an Rusage File produced:

```
p0spg01{emcleod}6: more Profile.out
        #     source .cshrc
        #     cd TEST/MOD*
        #     ls
        #     /usr/ecs/OPS/CUSTOM/bin/DPS/EcDpPrRusage MOD_PR10.exe > Profile.out
        p0spg01{emcleod}9: more profile.out
        # Resource Usage Information
        COMMAND=MOD_PR10.exe
        EXIT_STATUS=0
        ELAPSED_TIME=233.583145
        USER_TIME=10.046158
        SYSTEM_TIME=7.555547
        MAXIMUM_RESIDENT_SET_SIZE=4080
        AVERAGE_SHARED_TEXT_SIZE=0
        AVERAGE_UNSHARED_DATA_SIZE=0
        AVERAGE_UNSHARED_STACK_SIZE=0
        PAGE_RECLAIMS=151
        PAGE_FAULTS=0
        SWAPS=0
        BLOCK_INPUT_OPERATIONS=2
        BLOCK_OUTPUT_OPERATIONS=2710
        MESSAGES_SENT=0
        MESSAGES_RECEIVED=0
        SIGNALS_RECEIVED=0
        VOLUNTARY_CONTEXT_SWITCHES=1095
        INVOLUNTARY_CONTEXT_SWITCHES=2
        p0spg01{emcleod}10:
```

26.7.3 Checking the PGE for Memory Leaks

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

Integration and testing are conducted using the Goddard Automated Tools for Enhanced SSI&T (GATES) procedures. The GATES procedures are available on the web at:

http://gsfcsrvr8.gsfcmo.ecs.nasa.gov/GATES/TEST/AIRS/AIRS_FauxSwatter_EnterInfo.html.TEST

26.8 File Comparison and Data Visualization

The purpose of File Comparison is to verify that the output files produced at the DAAC are identical (within 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.

26.8.1 Using the GUI HDF File Comparison GUI

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

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

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

1 From the SSIT Manager, select **Tools→ Product Examination → HDF** from the menu.

The HDF File Comparison GUI window will be displayed.

2 In the HDF File Comparison Tool GUI, click on the **File 1** button.

- Read the Systems Description document and the Operations Manual. Both of these or their equivalent should be in the delivery.

26.8.2 Using the hdiff HDF File Comparison Tool

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

Comparing two HDF or HDF-EOS Files Using the hdiff File Comparison Tool

1 From the SSIT Manager, select **Tools→ Product Examination → File Comparison → HDF** from the menu.

The HDF File Comparison Tool window will be displayed.

- An xterm window running ***hdiff*** will be displayed.

2 In the xterm window at the prompt **Options? (-h for help)**, type in any desired options then press the **Enter** key.

- To see the list of available options, type **-h** then press the **Enter** key. to the prompt.

3 In xterm window at the prompt **1st file to compare?**, type ***filename1***, then press the **Enter** key.

- The ***filename1*** is the file name of the first of two HDF or HDF-EOS files to be compared.
- If ***filename1*** is not in the current directory (the directory from which the SSIT Manage was run), include the full path name with the file name.

4 In xterm window at the prompt **2nd file to compare?**, type ***filename2***, then press the **Enter** key.

- The ***filename2*** is the file name of the second of two HDF or HDF-EOS files to be compared. Select another studentsÕ file.
- If ***filename2*** is not in the current directory (the directory from which the SSIT Manage was run), include the full path name with the file name. The two files will be compared and the output will be displayed in the xterm window.

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

26.8.3 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.

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

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

Comparing Two ASCII Files

1 From the SSIT Manager, select **Tools→ Product Examination → File Comparison → ASCII** from the menu.
An xterm window running *xdiff* will be displayed.

2 In xterm window at the prompt **1st file to compare?**, type ***filename1***, then press the **Enter** key. Select a descriptor or mcf file in the directory with the PGE.
The ***filename1*** is the file name of the first of two ASCII files to be compared.
If ***filename1*** is not in the current directory (the directory from which the SSIT Manage was run), include the full path name with the file name.

3 In xterm window at the prompt **2nd file to compare?**, type ***filename2***, then press the **Enter** key.

The ***filename2*** is the file name of the second of two ASCII files to be compared. Select another studentÕs corresponding file.

If ***filename2*** is not in the current directory (the directory from which the SSIT Manager was run), include the full path name with the file name.

A window labeled **xdiff** will be displayed.

4 In the window labeled **xdiff**, view the differences between the two files displayed.

File ***filename1*** will be displayed on the left side of the window. File ***filename2*** will be displayed on the right.

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.

For further help on *xdiff*, type **man xdiff**, in an xterm window then press the **Enter** key.

Close the display window by using the pull down menu from the X window in the upper left corner.

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

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

26.8.4 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.

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

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

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

Comparing two Binary Files

---

1 From the SSIT Manager, select **Tools→ Product Examination → File Comparison → Binary** from the menu.
The Binary File Difference Assistant tool GUI will be displayed.

2 In the Binary File Difference Assistant tool GUI, click on one of the languages listed under the **Select Language** label. The choices are C, FORTRAN, or IDL.
The choice of language depends largely on preference. It does not necessarily have to be the language that was used to create the files being compared.

3 Optionally, click on either the **Image** button or the **Structure** button located under the label **Compare Function**.
Clicking on the **Image** button 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.
The displayed listing well documented and should be read.
The language of the code will depend on the language selection made in step 2.

4 Optionally, click on either the **Image** button or the **Structure** button located under the label **Driver**.
Clicking on the **Image** button 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.
The displayed listing well documented and should be read.
The language of the code will depend on the language selection made in step 2.

5 Optionally, click on either the **Help** button.
A Help window will be displayed.
To end help, click on the **Dismiss** button.
The Help window may remain displayed while using the Binary File Difference Assistant.

6 Once familiar with the code examples (steps 3 and 4), click on the **Copy** button.

- A window labeled **Enter Unique ID** will be displayed.
- In the field labeled **Enter unique file identifier:**, type ***fileID***, 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

FORTRAN:

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 with here document

- The files will be copied into the directory from which the SSIT Manager is being run.

Note: 1, Using any desired text editor, customize the files for the job at hand. Then build the executable using the customized makefile provided (for C and FORTRAN). Then run the program to perform the binary file comparison.

2, These files are templates that can be used to assist SSIT personnel in designing customized code to perform the binary file comparison.

26.8.5 Data Visualization

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

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

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

26.8.5.1 Data Visualization with EOSView

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

Log into an Algorithm and Test Tools (AITTL) environment using using a machine so configured. At the **PVC** this machine is **p0ais01**

1 Telnet into **p0ais01.**

2 **logon using your own ID and Password**

3 **cd /usr/ecs/TS1/CUSTOM/eosview.**

4 **Select EOSView, The EOSView GUI will be displayed.**

5 **Use the select buttons to guide you toward the view desired**

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

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

Viewing Product Metadata

1 From the SSIT Manager, select **Tools→ Product Examination → EOSView** from the menu.

- The EOSView GUI will be displayed.

2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Open.** The **Filter** GUI will be displayed.

3 In the **Filter** 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 **Directories** 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 is 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**.

26.8.5.1.2 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 and 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**.

26.8.5.1.3 Viewing HDF-EOS Grid Objects

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

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

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

Viewing HDF-EOS Grid Objects

1 From the SSIT Manager, select **Tools→ Product Examination → EOSView** from the menu.
The EOSView GUI will be displayed.

2 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.
Information on **Grid Information, Projection Information, Dimensions, Attributes** for the selected object can be displayed by clicking on the appropriate checkboxes.

3 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.

4 To display the data in the form of a table of values, in the GUI labeled **EOSView - Grid - *GridObjectName* - Start/Stride/Edge**, click on the checkboxes for both **YDim** and **XDim** 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.

5 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.

6 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.

7 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.

8 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.

9 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.

10 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.

26.8.5.1.4 Viewing HDF-EOS Swath Objects

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

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

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

Viewing HDF-EOS Swath Objects

---

1 From the SSIT Manager, select **Tools→ Product Examination → EOSView** from the menu.
The EOSView GUI will be displayed.

2 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.
Information on **Dimensions, Geolocation Mappings, Indexed Mappings, Geolocation Fields**, **Attributes** for the selected Swath Object can be displayed by clicking on the corresponding checkboxes.

3 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.

4 To display the data in the form of a table of values, in the GUI labeled **EOSView - Swath - *SwathObjectName* - Start/Stride/Edge**, click on the checkboxes for both **ScanLineTra** and **PixelsXtrac** and then click on the **OK** button.

5 To display the data field in image form, in the GUI labeled ***MyDataField***, click on the **File** menu and then select **Make Image**.

A GUI labeled **EOSView - Swath/Grid Image** will appear.

6 Optional colorization, zooming, panning while zooming features can be used in the GUI labeled **EOSView - Swath/Grid Image** to obtain your desired image.

– To end the session with displaying Swath object, in the GUI labeled **EOSView - Swath/Grid**, click on the **File → Close**.

26.8.5.1.5 Viewing HDF SDS Objects

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

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

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

Viewing HDF SDS Objects

---

1 From the SSIT Manager, select **Tools→ Product Examination → EOSView** from the menu.

The EOSView GUI will be displayed.

2 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**).

3 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.

4 To display the data field in image form, in the GUI labeled ***MySDS***, click on the **File** menu and then select **Make Image**.

A GUI labeled EOSView - Image Display Window - ***MySDS*** will appear,

5 **Optional colorization, zooming, panning** while zooming can be used to obtain your desired output.

6 To end the session with displaying Swath object, in the GUI labeled **EOSView - Image Display Window - *MySDS***, select **File → Close** from the menu.

- The **EOSView - Image Display Window - *MySDS*** GUI will disappear.

26.8.5.2 Data Visualization with the IDL Tool

26.8.5.2.1 Viewing Product Data with the IDL Tool

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

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

The following is a list of tools, and or assumptions:
1.The SSIT Manager is running.
2.The output file is binary, ASCII, or one of the other IDL supported data formats.
3.IDL has been properly installed and is accessible to the user.

Viewing Product Data with the IDL Tool

---

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

2 Select the procedure depending upon the activity to perform.

3 To end the IDL session, close any display windows remaining, then at the IDL prompt type **quit**, then press the **Enter** key.
The IDL session will be closed.

26.8.5.2.2 Creating an Image Display Using IDL

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

The following is a list of tools, and or assumptions:
1.The SSIT Manager is running.
2.The PGE output file to be examined is of an IDL-supported type/format (if in doubt, consult the IDL Reference Guide

3.IDL has been properly installed and is accessible to the user.
4.For binary files, data is assumed to be 8-bit characters

Creating an Image Display Using the IDL Tool - Binary Data

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

2 At the IDL prompt, type ***OPENR,1,'MyBinaryFilename'***, press the **Enter** key.
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.

3 At the IDL prompt, type ***MyImage=BYTARR(dimX, dimY)***, press the **Enter** key.
The ***MyImage*** is the name to be given to the image once created.
The ***dimX*** and ***dimY*** are the dimensions of the input data.

4 At the IDL prompt, type ***READU,1,MyImage***, press the **Enter** key.

5 At the IDL prompt, type ***TV,MyImage***, press the **Enter** key.
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).

6 At the IDL prompt, type ***LOADCT,3***, press the **Enter** key.

- This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.

7 At the IDL prompt, type ***CLOSE,1***, press the **Enter** key.
This closes logical unit 1.
Always close logical units or an error will result the next time an access is attempted.

Creating an Image Display Using the IDL Tool - ASCII Data

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

2 At the IDL prompt, type ***OPENR,1,('MyASCIIfilename')***, then press the **Enter** key.

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.

3 At the IDL prompt, type ***MyImage=BYTARR(dimX,dimY)***, then press the **Enter** key.

The ***MyImage*** is the name to be given to the image once created.
The ***dimX*** and ***dimY*** are the dimensions of the input data.

4 At the IDL prompt, type ***READF,1,MyImage***, then press the **Enter** key.

5 At the IDL prompt, type ***TV,MyImage***, then press the **Enter** key.

The image, **MyImage**, is 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).

6 At the IDL prompt, type ***LOADCT,3***, then press the **Enter** key.

This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.

7 At the IDL prompt, type ***CLOSE,1***, then press the **Enter** key.

This closes logical unit 1.
Always close logical units or an error will result the next time an access is attempted.

Creating an Image Display Using the IDL Tool - 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.

26.8.5.2.3 Creating an Image Display Using the IDL Tool - PGM Data

---

1 From the SSIT Manager, select **Tools→ Product Examination → IDL** from the menu.

An xterm (on the AIT Sun) will be displayed within which the IDL command interpreter will be run.

2 At the IDL prompt, type ***READ_PPM,"MyPGMfilename",MyImage,r,g,b***, then press the **Enter** key.
The MyPGMfilename is the full path name and file name of the PGM formatted data file.
The double quotes (“) must be included around the path/file name.
The MyImage is the name to be given to the image created.

3 At the IDL prompt, type ***TVLCT,r,g,b***, then press the **Enter** key.
Note that r,g,b color table syntax is used for most formatted file types in IDL.

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

26.8.5.2.4 Saving an Image Display Using IDL

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

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

1. The SSIT Manager is running, IDL is running
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
4. The image display is to be saved in a binary (8-bit) or ASCII (comma-delimited characters) format.

26.8.5.2.5 Save an image display using IDL - Binary Data

---

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

2 At the IDL prompt, type ***OPENW,1,('MyBinaryFilename.bin'***), then press the **Enter** key.
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.

3 At the IDL prompt, type ***WRITEU,1,MyImage***, then press the **Enter** key.
The ***MyImage*** is the name of the image to save.

4 At the IDL prompt, type ***CLOSE,1***, then press the **Enter** key.
This closes logical unit 1.
Always close logical units or an error will result the next time an access is attempted.

26.8.5.2.6 Save an image display using IDL - ASCII Data

1 From the SSIT Manager, select **Tools→ Product Examination → IDL** from the menu.

An xterm (on the AIT Sun) will be displayed within which the IDL command interpreter will be run.

2 At the IDL prompt, type ***OPENW,1,('MyASCIIfilename.asc')***, then press the **Enter** key.

The ***MyASCIIfilename.asc*** is the full path name and file name of the binary data file to write out.

The single quotes **(')** must be included around the path/file name.

The **1** is the logical unit number.

3 At the IDL prompt, type ***PRINTF,1,MyImage***, then press the **Enter** key.

The **MyImage** is the name of the image to save.

4 At the IDL prompt, type **CLOSE,1**, then press the **Enter** key.

This closes logical unit 1.

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

26.8.5.2.7 Save an image display using JPEG Data

1 At the IDL prompt, type **WRITE_JPEG,*"MyJPEGfilename.jpg",MyImage***, press **Return**.

- The ***MyJPEGfilename.jpg*** is the full path name and file name of the JPEG data file to write out.

26.8.5.2.8 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]**, and press **Return**.

- 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.

26.8.5.2.9 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'**, and press **Return**.

- 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.

26.8.5.2.10 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**, and press **Return**.

- 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'**, press **Return**.

- The ***MyPlotDisplay*** is the session name of the plot display .
- The ***MyPlotOutput.ps*** is the desired name for the saved file.

3. The SAVE option number 4 sets the output file type to PostScript (ps). There are other options, of course (consult the IDL manuals).

26.8.5.3 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.

26.8.5.4 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 wont 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 following is a list of tools, and or assumptions:

1. The SSIT Manager is running.
2. IDL is running
3. 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**.
4. 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.**

26.8.5.4.1 Raster Mapping - Global Data Set Image

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

2 At the IDL prompt, type ***TV,MyImage***, then press the **Enter** key.
The ***MyImage*** is the image name of the global image data set.
The image, ***MyImage***, should then be displayed.

3 At the IDL prompt, type ***MAP_SET,/ORTHOGRAPHIC***, then press the **Enter** key.
IDL also supports other map projections. Refer to IDL Reference Guide.

4 At the IDL prompt, type ***MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN)***, then press the **Enter** key.
The ***MyNewImage*** is the name to assign to the resulting image.
The ***MyImage*** is the name of the original global image data set.

5 At the IDL prompt, type ***TV,MyNewImage,startx,starty***, then press the **Enter** key.
The image ***MyNewImage*** should then be displayed.

6 Optional overlay Lat/Long. At the IDL prompt, type ***MAP_GRID***, then press the **Enter** key.
This overlays Lat/Long graticule onto ***MyNewImage***.

7 Optional overlay world coastlines. At the IDL prompt, type ***MAP_CONTINENTS***, press the **Enter** key.
This overlays world coastlines onto **MyNewImage**.

Note: IDL also supports other resampling methods besides Bilinear Interpolation (***/BILIN***).
Refer to *IDL Reference Guide*.

26.8.5.5 Raster Mapping - Sub-Global Data Set Image

For a sub-global data set image, one having geocentric-LLR coordinates defined for subintervals of longitude and latitude (e.g. from -88 to -77 degrees East Longitude and 23 to 32 degrees North Latitude.

The following is a list of tools, and or assumptions:
1.The SSIT Manager is running.
2.IDL is running

---

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

2 At the IDL prompt, type ***TV,MyImage***, then press the **Enter** key.
The ***MyImage*** is the image name of the sub-global image data set.
The image, ***MyImage***, should then be displayed.

3 At the IDL prompt, type ***MAP_SET,/MERCATOR,LIMIT=[lat1,lon1,lat2,lon2]***, then press the **Enter** key.

4 Example: ***MAP_SET,/MERCATOR,LIMIT=[23,-88,32,-77],.***
The ***lat1***, ***lon1***, ***lat2***, and ***lon2*** specify the latitude and longitude intervals of the sub-global image data set.

4 At the IDL prompt, type ***MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN.LATMIN=lat1,LATMAX=lat2,LONMIN=lon1,LONMAX=lon2)***, then press the **Enter** key.
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.

5 At the IDL prompt, type ***TV,MyNewImage,startx,starty***, then press the **Enter** key.

The image ***MyNewImage*** should then be displayed.

6 Optional overlay Lat/Long. At the IDL prompt, type ***MAP_GRID***, then press the **Enter** key.

This overlays Lat/Long graticule onto ***MyNewImage***.

7 Optional overlay world coastlines. At the IDL prompt, type ***MAP_CONTINENTS***, then press the **Enter** key.

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.

26.9 Science Software Integration and Test (SSI&T) Release 6A.05.

The process of SSI&T or integration of EOS Instrument Science Software into the ECS has been developed and refined over three iterations of ECS. IR1, the Pre-Release B, and Release 4 through Release 6A.05. This release is the at-launch system supporting EOS-AM1 instruments. Although every attempt has been made to keep integration procedures for science software consistent through succeeding releases, basis architectural differences have led to significant variances. This section describes the architecture of the last iterations of ECS.

26.9.1.1 RELEASE 6A Architecture: Overview

The release 6A architecture overview is located in Release 6A Segment/Design Specification for ther ECS Project 305-CD-600-001. For the purposes of this section (Release 4 architecture) adequately covers relevent information as a primer into the SSI&T process.

The Release 4 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 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.
- 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 number of external interfaces which ECS will have is potentially very large, and the interfaces can serve very diverse functions, such as high-volume ingest of level 0 data and low-volume ingest of data from field campaigns.

Table 26.9.3-1 provides procedural capabilities through RELEASE 6A.05.

26.9.1.2 ECS Subsystems

The following sub-sections provide brief overviews for each of these subsystems. More detailed discussions of their design breakdown can be found in 305-CD-510-001.

26.9.1.2.1 Client Subsystem (CLS)

The Client provides users with an interface through which they can access ECS services and data. It also gives science software access to the ECS services, as well as direct access to ECS data. Access is provided through graphic user interface (GUI) application tools for displaying the various kinds of ECS data (e.g., images, documents, tables), and libraries representing the client APIs to ECS services. The client subsystem follows an object oriented design. The design is built around a core set of 'root' objects from which all other software will inherit its behavior.

26.9.1.2.2 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. The Client Subsystem provides the user interface which enables access to the IOS.

26.9.1.2.3 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.

26.9.1.2.4 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.
- Document Management System - Web server and database technology are used to implement a document management system to provide storage and information retrieval for guide documents, science software documentation, and ECS earth science related documents.
- Data Type Libraries - The Data Server will use custom dynamic linked libraries (DLLs) to provide an extensible means of implementing the variety of ECS earth science data types and services, and will provide a consistent 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.

26.9.1.2.5 Ingest Subsystem (INS)

This subsystem deals with the initial reception of all data received at an EOSDIS 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.

26.9.1.2.6 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. The tool 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.

26.9.1.2.7 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 eventually 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
- need for high availability
- providing 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

26.9.1.2.8 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 infra-structural 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.

Note: DCE has been replaced with Socket Scripts that replicate those functions that DCE provided.

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 inter-process communication and specialized infra- structural services such as security, directory, time, asynchronous message passing, event logging, lifecycle service, transaction processing and World Wide Web (WWW) service.

26.9.1.2.9 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 will be fully 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 of 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 the centerpiece of its system management solution. 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 will be 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.).

26.9.1.2.10 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.

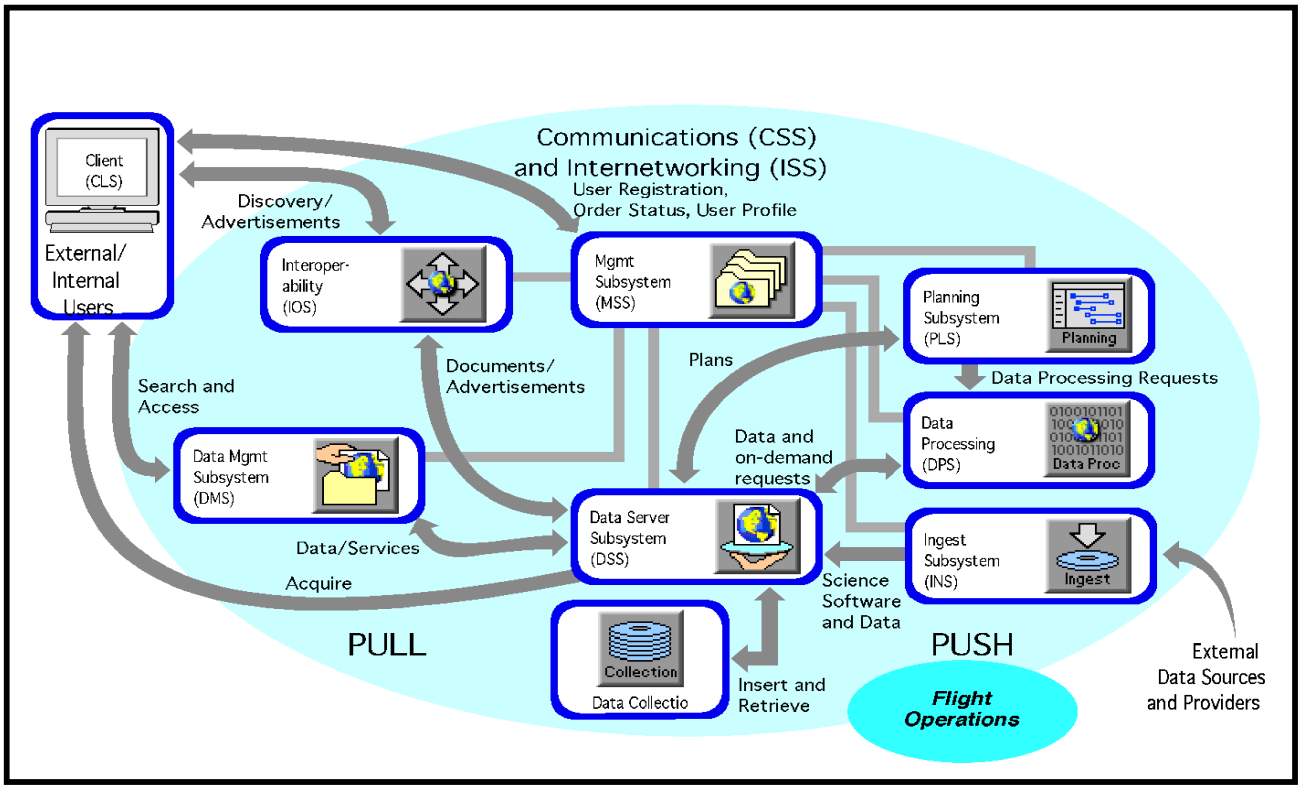


Figure 26.9.1-1. ECS Communications /Interetworking Subsystems

26.9.2 Implications for SSI&T Functions.

Table 26.9.3-1 . List the major Functions that will be encountered by SSI&T staff. These architectural functions characterize the RELEASE 6A systems. The major functions are due to the presence of Ingest and Data Server Subsystems in RELEASE 6A

Table 26.9.3.1. Major SSI&T Functions within VERSION 2.0

Function	**Release 6A.05.**
System Operation	All servers must run and communicate with each other; bring up manually, or use ECS Assist tool.
Ingest Ancillary Data Granules	Ingest GUI, ESDTs must be visible to ADV server.
ESDT Insert	Use Ingest
ESDT Verification	verify through ADV
DAP, SSAP Insert	Use Ingest
PDPS Database Population	More attributes, production rules
PGE Operation	When all data is available; DPR activated. Reprocessing Complex chaining through production rules.
File Access	verify presence through ADV; ftp from SDSRV; access to multiple sites
Multi-file Granule Support	Files inserted together, accessed as a single granule. Granule Deletion.
Subscription Management	Subscription Manager

26.9.2.1 Release 6A Capabilities

The Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS) capabilities are developed in terms of formal releases. Release 6A, which is controlled by Configuration Management, provides capabilities to support the ingest and archive of raw data obtained from the EOS AM 1 mission spacecraft, morning equator crossing spacecraft series (Terra (AM-1)), EOS PM 1 mission spacecraft, afternoon equator crossing spacecraft series (Aqua (PM-1)) and the Land Remote-Sensing Satellite (Landsat 7). Other capabilities provided by Release 6A include processing the data obtained, distributing raw or processed data as requested, quality assurance of processed data, supporting communication networks, and systems monitoring via interfaces with the ECS operations staff.

Release 6A unique capabilities include:

- support the regeneration and archive of products previously produced and archived
- support ingest, production, storage, and distribution of Terra and Aqua instrument data
- support writing files to CD-ROMs and DLT tape drives for distribution
- support data compression (Unix and Gzip) for distribution
- provide access to non-science data collections by a limited number of attributes and values
- provide Science Data Server scalability/operability improvements
- support FTP Pull Subscription
- provide machine-to-machine gateway support between SIPS and ECS for data orders
- provide capability for operator deletion of granules
- support EOS Data and Operations System (EDOS) backup by distribution of replacement data to EDOS on D3 tapes and ingesting of backup data on D3 tapes from EDOS
- archive and distribution request management by multi-host scheduling
- ingest browse and metadata from IGS tapes
- allow users to request a data processing request associated with a DAR at the time of submittal

A more detailed overview of the Release 6A ECS may be found in the Release 6A Segment/Design Specifications for the ECS Project, 305-CD-600-001.

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

ECS (EOSDIS Core System) is a complex system comprising of many subsystems and components running on multiple heterogeneous host machines. The coordination of all the subsystems for SSI&T is an arduous, time consuming, error prone task. In order to improve our effectiveness and efficiency, an easy-to-use GUI tool, “ECS Assistant,” has been developed to facilitate ECS SSI&T activities. Its origin as a development tool provides "fallback" functionality for other tools, such as HP OpenView. Because of high overhead use of system resources, ECS Assistant has been scaled back in functionality.

Currently, the ECS Assistant has lost its capability to monitor ECS. This capability now exist with GUI Whazzup. ECS Assistant continues to have the ability to Monitor Subsystem functions and is mainly used for doing ECS Assistant Subsystem Installs (E.A.S.I.) and staging ESDT/DLL's into the directories, CUSTOM/data/ESS and CUSTOM/lib/ESS respectively. Database Review capability still exists however. The ESDT Manager Installation/Deletion functions are no longer available, the ability to startup and shutdown subsystem servers and the use of the ECS Logfile Viewer have been taken away in Release 6A.05.

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

- To **start up or shut down all servers** at the same time, a Script is used that accesses a list of subsystem servers.
- To **start up and shut down servers individually** using a Script established within each subsystem.
- To **graphically monitor the server up/down status** with the **Whazzup GUI**
- To **view** ESDTs for SSI&T
- To **review various databases** used in the ECS system by using **an ISQL Browsers** established in each subsystem.

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

- **Section one** explains how to use **Scripts** to facilitate and manage the subsystems and their servers, including **server start up and shut down.**
- **Section two** contains an ESDT monitor function which includes reviewing the Science Data Server database through the **DB Viewer** GUI provided by ECS Assistant

26.10.1 Using Scripts to Start Up / Shut Down Servers

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

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

26.10.1.1All Servers Start Up/Shut Down

1 Script name: **rcmd_start_mode TS1, rcmd_kill_mode TS1**

2 Location: **/home/cmshared/bin/**

3 **HOSTLST** contains all the Subsystem Servers that will be operated upon.

26.10.1.2 Individual Servers Start Up/Shut Down

1 If you want to Start or Kill a server, you must be on the individual machine that supports the server in question.

2 Type **cd /home/cmshared/bin/** <**ENTER**>

3 Type **start_mode TS1** to start up a server or **kill_mode TS1** to shutdown a server**.** <**ENTER**>

4 Repeat steps 1-3 to start up or shut down other servers.

26.10.2 Bringing Up ECS Assistant

Assumptions:

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

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

26.10.2.1 To Run The ECS Assistant

1 At the UNIX Console or Terminal **setenv DISPLAY clientname:0.0** and then press the **Enter** key.

2 Create an xterm by typing**: xterm -n hostname &**

The ***hostname*** is the name of the machine on which the ECS Assistant is to be displayed, *i.e.*, the machine that your are using.

3 Log into one of the host machines used for SSIT, (Tested using **telnet p0acs03** for **SDSRV** and **p0ins02** for **IOS**, **ID:, PW:**

4 At the UNIX Console or Terminal type **setenv DISPLAY clientname:0.0** and then press the **Enter** key**.**

To verify the setting, type **echo $DISPLAY** and then press the **Enter key.**
setenv ECS_HOME/usr/ecs
setenv TK_LIBRARY/tools/lib/tk4.2
(Mount point called /tools must be mounted.)

5 If necessary, at the UNIX prompt on the host from which the ECS Assistant is to be run, type **cleartool setview *ViewName*** and then press the **Enter** key.

The ***ViewName*** is the ClearCase view to be used while the ECS Assistant is running in this session. For example, type: cleartool jdoe and then press the **Enter** key.

A ClearCase view is required only if the ECS Assistant needs to be able to "see" into a ClearCase VOB; a view is not necessary otherwise.

4 At the UNIX prompt, type **EA** and then press the **Enter** key.

- File **/tools/common/ea** must exist in the path. (This can be set in the **.cshrc** or **.kshrc** file)

EA is an alias for: **/tools/common/ea** is the path where ECS Assistant is installed.
This will invoke the ECS Assistant GUI with three push buttons for selecting the proper activities, as indicated in Figure 26.10.2.1-1.

Figure 26.10.2.1-1. ECS Assistant Main GUI

7 At the ECS Assistant GUI, click the **Subsystem Manager** pushbutton.

This will invoke the Subsystem Manager GUI, as indicated in Figure 26.10.2.2-2.

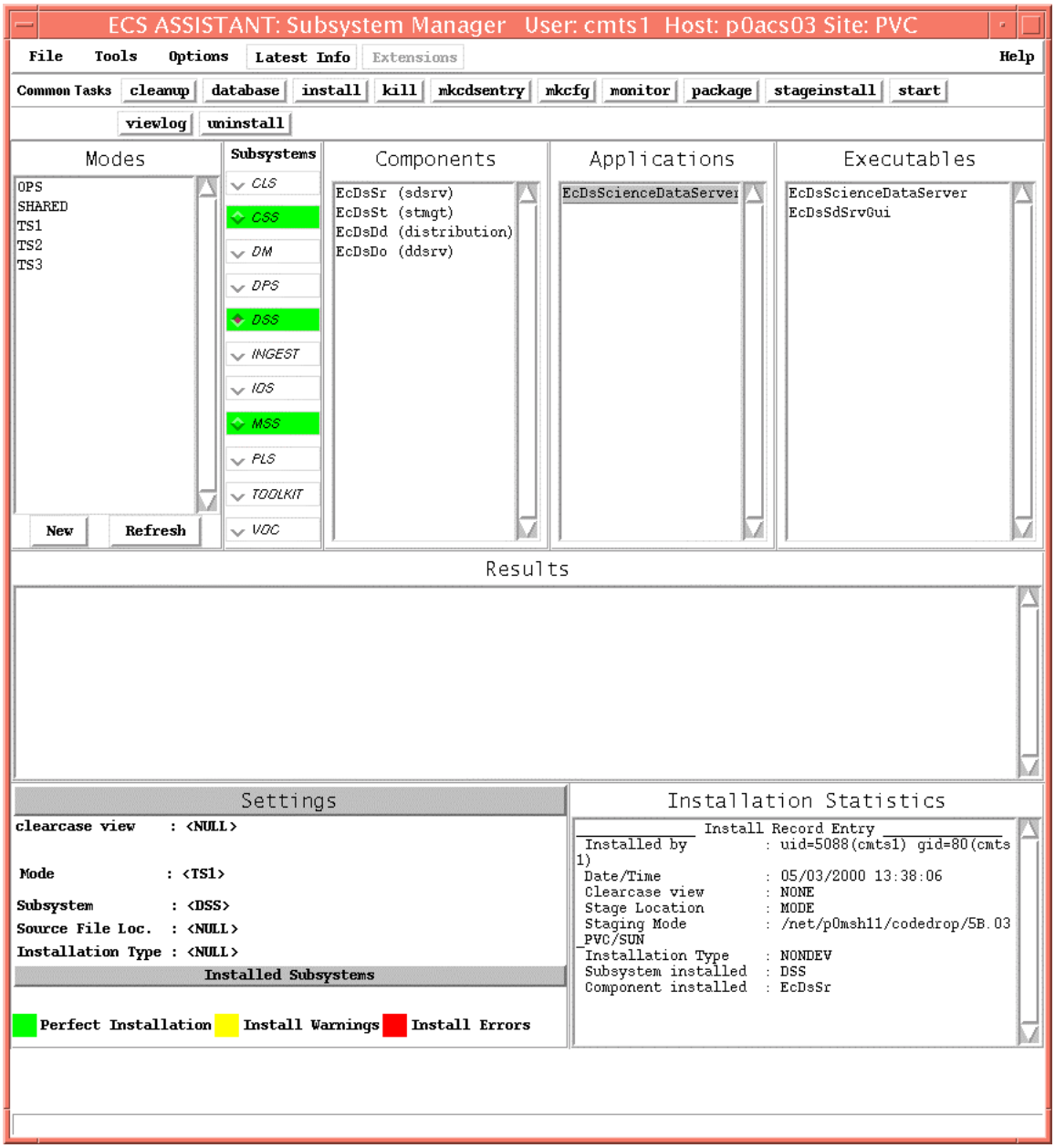


Figure 26.10.2.2-2. Subsystem Manager GUI

8 Select a mode by clicking a mode in the mode listing. The mode should be the one to be used for SSI&T.

Once the mode is selected, the color of the subsystem name list is changed.

9 Select a subsystem with the **Subsystem** radio button.

The component list for the selected subsystem will appear in the component window.

10 Select a component by clicking the component name under the component window.

The selected component will be highlighted.
The server list corresponding to that component will appear in the server window.

11 Select a server by clicking the server name from the server list under the servers window.

The server selected is highlighted.

12 Note that the functionality for Viewlog, Kill, Start and Monitor are not supported for Release 6A.05. Refer to the section titled: "Using Scripts to Start Up / Shut Down Servers."

13 To exit the Subsystem Manager GUI, select **File..Exit** in the menu bar of the Subsystem Manager GUI.

This will terminate the Subsystem Manager GUI.

26.10.3 Monitoring ECS using WHAZZUP Web GUI

To use the WHAZZUP GUI to monitor ECS make the following enteries to display the GUI.

Log into a server that has Web access.

Setenv DISPLAY ..:0.0

/tools/bin/ssh -l cmshared t1code1u.ecs.nasa.gov

Password:

setenv DISPLAY :0.0
ssh t1pls02

/tools/bin/netscape/ when the Netscape GUI is displayed, then select from the Bookmark: **whazzup.** A GUI such as the one depicted in Figure 26.10.3-1 should appear on the terminal screen.

Notice the other links listed in the bookmark List depicted in Figure 26.10.3-2, that go to sites that support ECS.

Figure 26.10.3-1. WHAZZUP ECS Monitor GUI

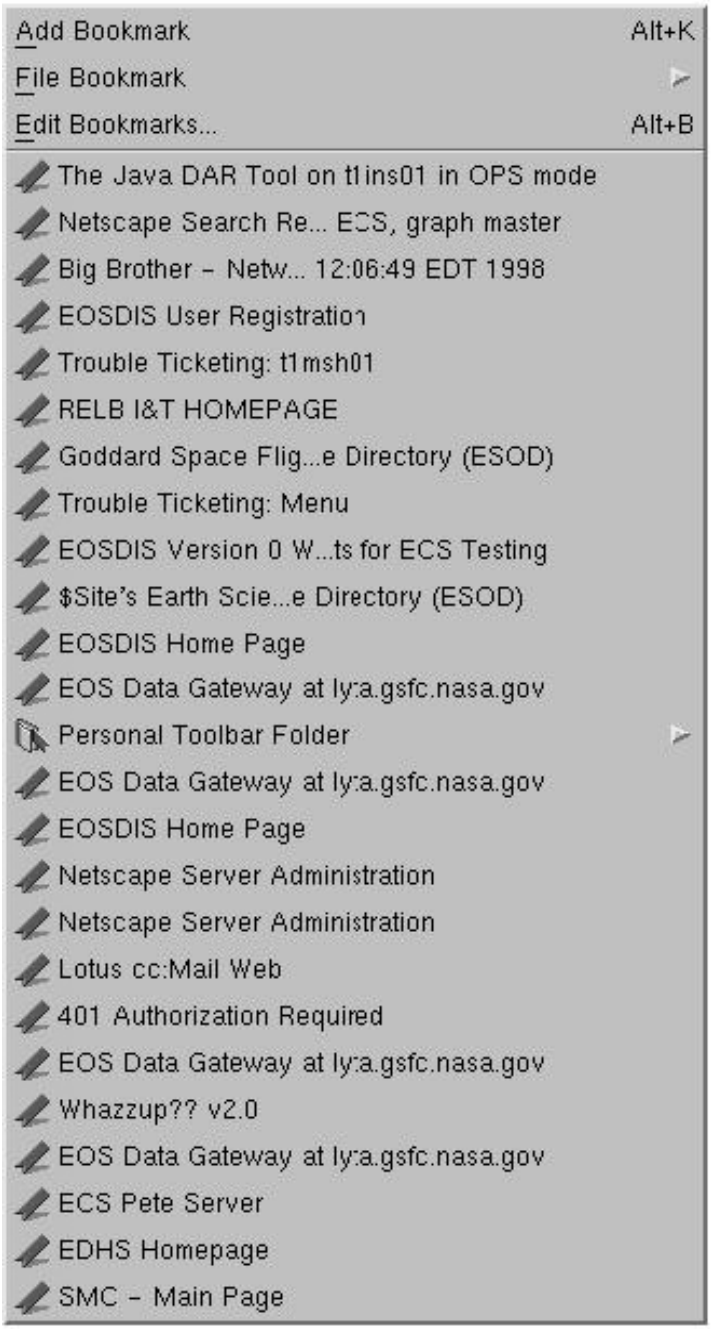


Figure 26.10.3-2. WHAZZUP Bookmark List

26.10.4 Using ECS Assistant to View ECS Science Data Server Database

ESDTs and their granules can be viewed in the 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.

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

Assumptions:

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

To start up the ECS monitor GUI, execute the procedure steps that follow:

1 Follow Section Bringing Up ECS Assistant to invoke the ECS Assistant GUI.
- The ECS Assistant GUI will be launched.

2 At the ECS Assistant GUI, select ESDT Manager GUI by clicking the ESDT Manager.
- The ESDT manager GUI will appear. See Fig. 26.10.4-1.

3 At the ECS ESDT Manager GUI, select the DB Viewer by clicking the **DB Viewer** button. See Figure 19.
- The Database Login GUI will appear as shown in Figure 26.10.4-2.
- Fill in the fields to point to the specific database for the mode used.
- Click Login to open the DB Viewer.
- The DB Viewer GUI will appear as shown in Figure 26.10.4-3.
- ESDTs are listed in the Collections window.

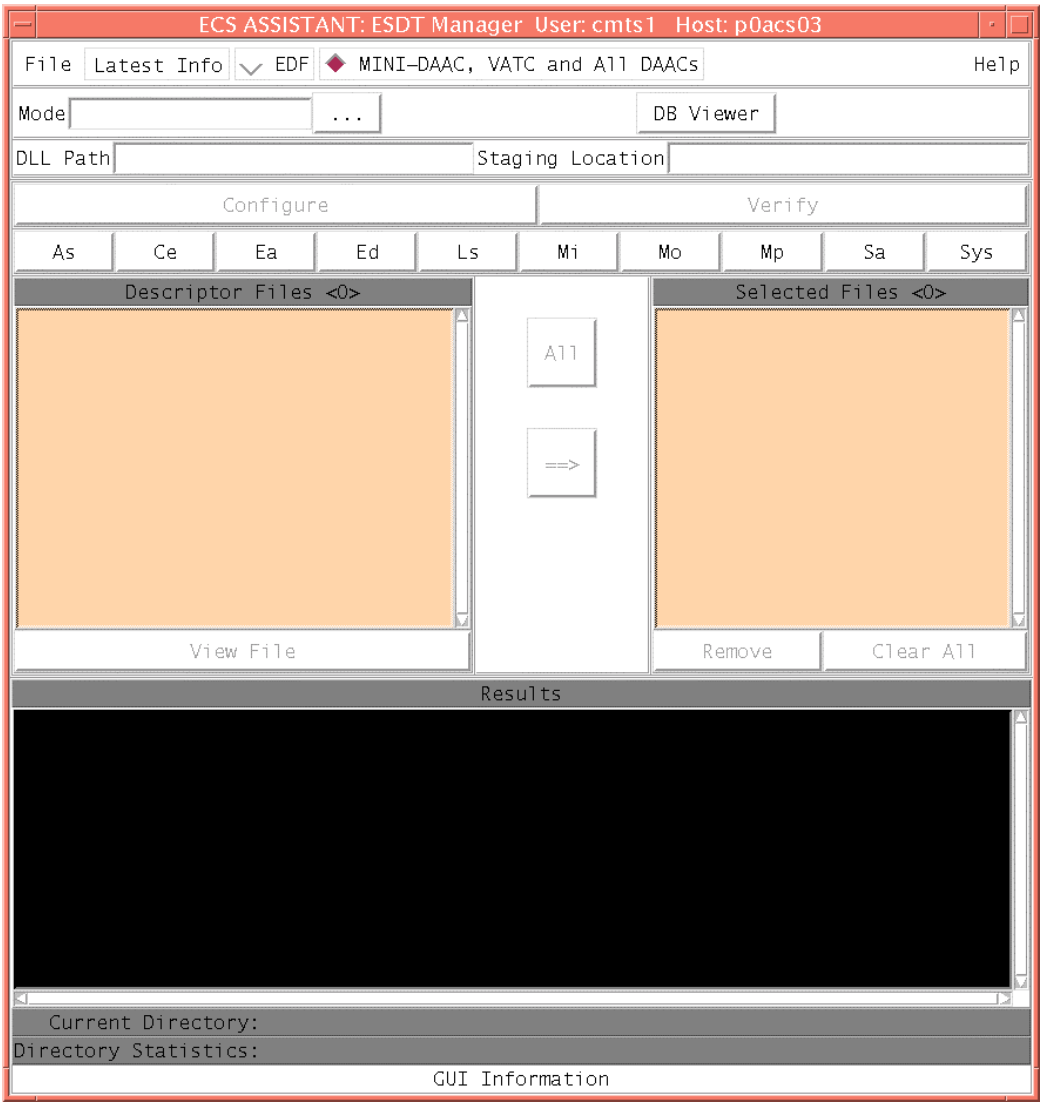


Figure 26.10.4-1. ESDT Manager GUI

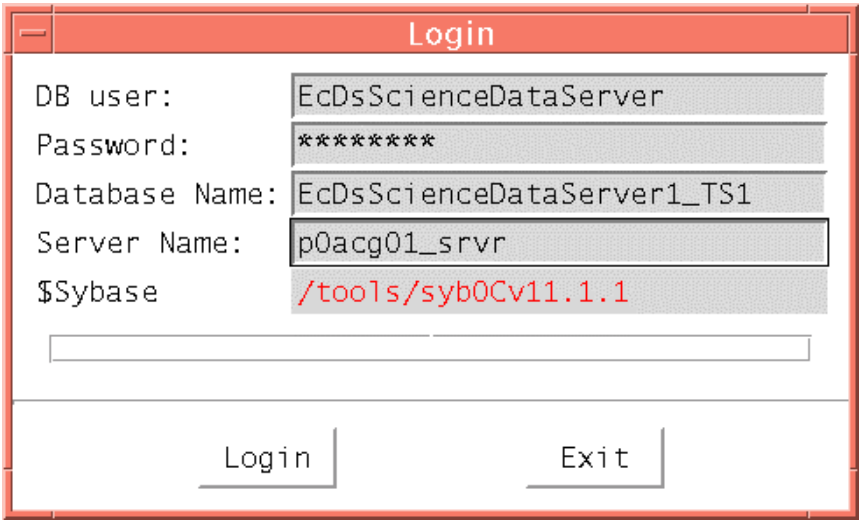


Figure 26.10.4-2. Database Login GUI

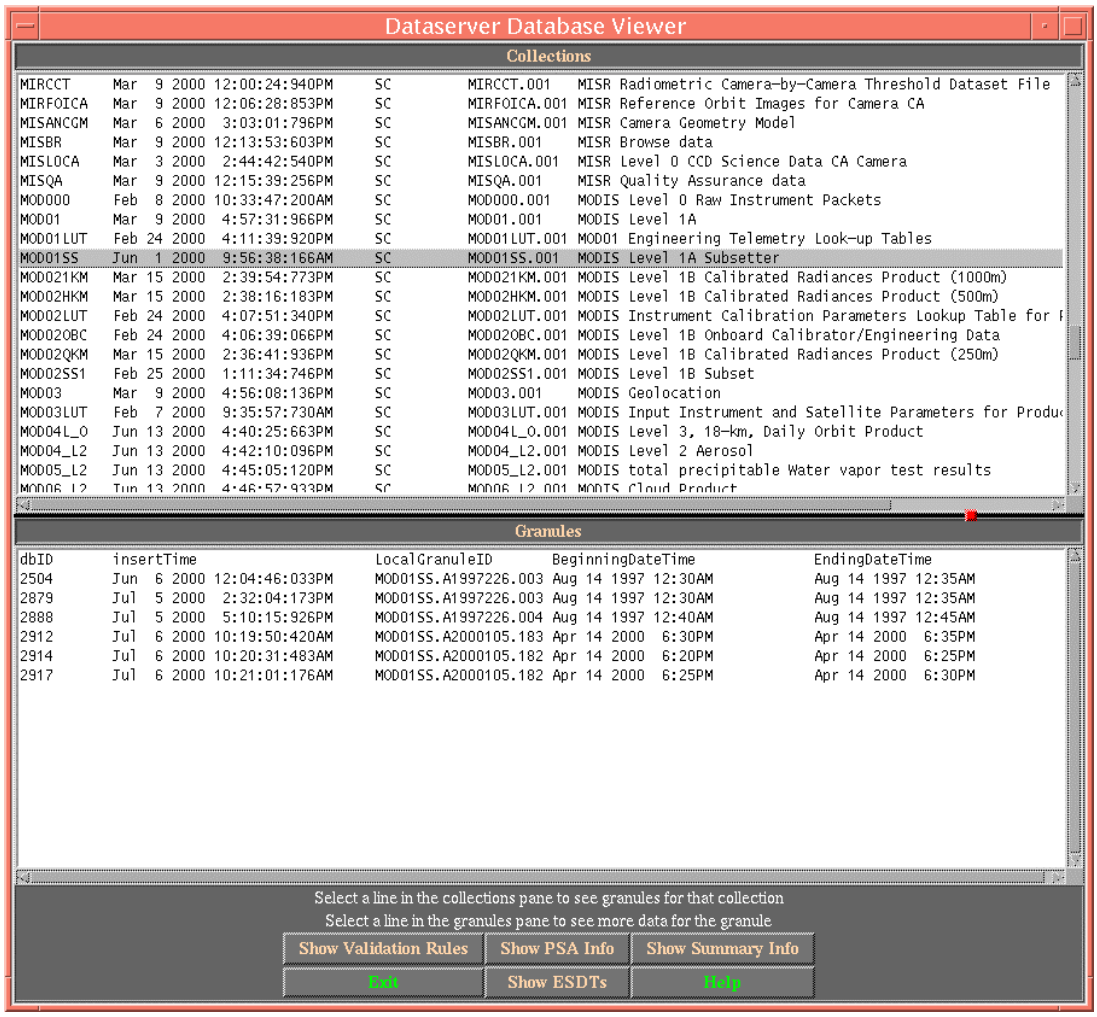


Figure 26.10.4-3. DB Viewer GUI

4 To view the inserted granules for a selected ESDT, first select an ESDT by clicking its short name in the Collections window.

- The selected ESDT is highlighted.
- Granule information for that ESDT, if there is any, will be listed in the Granules window.

5 To exit, click the **EXIT** button.

- This will end the DB Viewer GUI.

26.10.4.1 Using ECS Assistant to View ECS Science Data Server Database

1. Invoke the ECS Assistant GUI.

2. At the ECS Assistant GUI, invoke the **ESDT Manager**.
3. At the ECS ESDT Manager GUI, select the **DB Viewer** to open the login window for the database.
4. Fill in the Login field and open **DB Viewer** for the selected database.
5. View the inserted granules for a selected ESDT.

26.11. ESDT Management

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

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

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

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

This section will describe how to Check, Install, Remove or Update an ESDT.

26.11.1 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_srvr) 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 - g0acs3.

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*>**, press **Return**.

- The Science Data Server GUI will be presented.

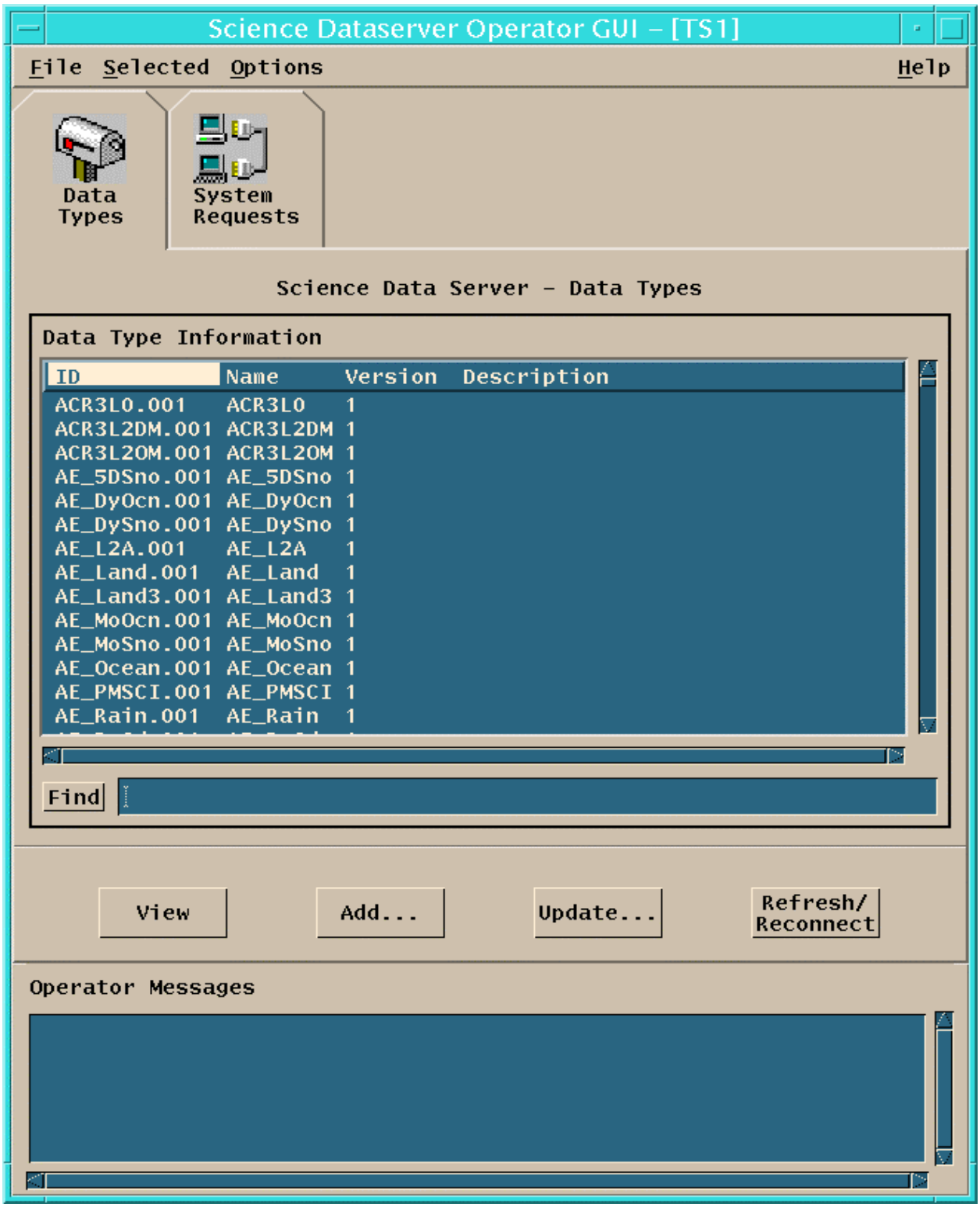


Figure 26.11.1-1. Science Data Server Operator GUI

4 Scroll thought 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 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.

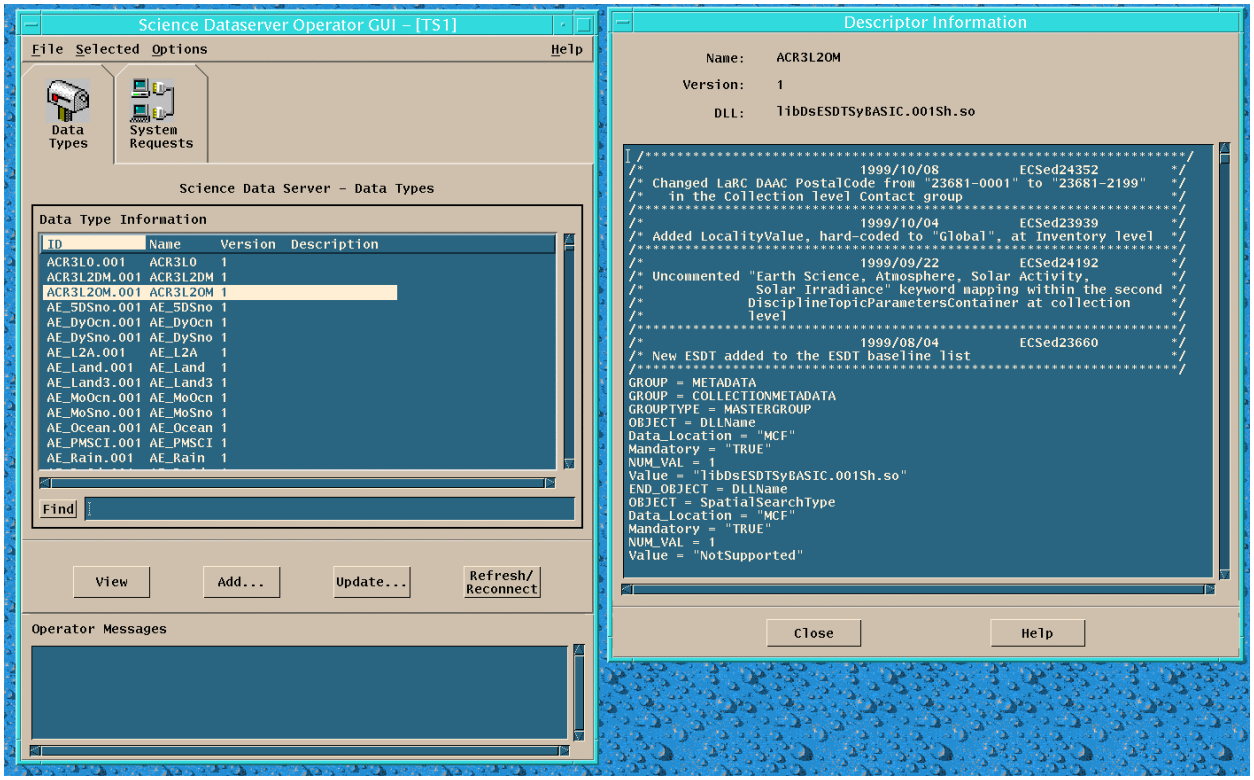


Figure 26.11.1-2. Viewing An ESDT Descriptor File

Table 26.11.1-1. 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 **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsSdSrvGuiStart <*MODE*>**	Press **Return**
4	Scroll through Data Type Information box	none
5	Select shortname/version	Click the **View** button

26.11.2 Removing ESDTs

If one wants to install a modified version of an ESDT, retaining the shortname and version number, there are two ways to go. The first way is to completely replace the descriptor file and corresponding ESDT data in the 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	**Disadvantages**
Can be done with any ESDT	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	**Disadvantages**
No removal scripts need to be run Granule references are preserved.	Only certain ODL structures supported. Science Data Server must be brought up with a StartTemperature=maintenance. This means it is unusable by anything else until it is recycled with an operational StartTemperature. Many other conditions must be met.

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, EcIoDbDeleteCollection, resides in

…/dbms/IOS on the IOS server platform. A separate set of procedural steps will be given for each scenario.

26.11.2.1 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.

1. 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 : ***XXXXXXX*** ← use appropriate value
- Please enter DCE password : ***XXXXXXX*** ← 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 : XXXXXXX
Please enter DCE password : XXXXXXX
----- 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 environamental 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.

1. At the UNIX prompt ,
type **setenv DSQUERY p0ins02_srvr,** press **Return** .

2. At the UNIX prompt ,
type **setenv DBNAME EcDmDictService_TS1,** press **Return** .

3. At the UNIX prompt,
type **setenv DBUSERNAME **********,** press **Return** .

4. At the UNIX prompt,
type **setenv DBPASSWD *******,** press **Return** .

1. 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.

4 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 26.11.2.1-1. ESDT Removal Using ContributionDriverStart - Quick-Step Procedures

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

26.11.2.2 ESDT Removal Scenario 2 - Using EcIoDbDeleteCollection

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 environamental 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.

4 At the UNIX prompt ,
type **setenv DSQUERY p0ins02_srvr,** press **Return** .

5 At the UNIX prompt ,
type **setenv DBNAME EcDmDictService_TS1,** press **Return** .

6 At the UNIX prompt,
type **setenv DBUSERNAME **********,** press **Return** .

7 At the UNIX prompt,
type **setenv DBPASSWD *******,** press **Return** .

- 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:

8 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 26.11.2.1-1. ESDT Removal Using ContributionDriverStart - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
	Removal From Advertising Server DB	
1	Logon to (IOS) Advertising Server platform	Log onto IOS host
2	type **cd /usr/ecs/<*MODE*>/CUSTOM/dbms/IOS**	Press **Return**
3	type **EcIoDbDeleteCollection *mode USERNAME PASSWORD SERVER DATABASE ShortName VersionID***	Press **Return**
	Removal From Data Dictionary Server DB	
1	Logon to (DMS) Advertising Server platform	Log onto DMS host
2	type **cd /usr/ecs/<*MODE*>/CUSTOM/dbms/DMS**	Press **Return**
3a	type **setenv DSQUERY *sybase_ddict_srvr***	Press **Return**
3b	type **setenv DBNAME *EcDmDictService_dbname***	Press **Return**
3c	type **setenv DBUSERNAME *************	Press **Return**
3d	type **setenv DBPASSWD *************	Press **Return**
4	type **DmDbCleanCollection *ShortName VersionID***	Press **Return**
	Removal From Subscription Server DB	
1	Logon to (CSS/IDG) SubscriptionServer platform	Log onto CSS/IDG host
2	type **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **dbDeleteEvents.csh *mode ShortName VersionID UserName Passwd***	Press **Return**
	Removal From Science Data Server DB	
1	Logon to (DSS) Science Data Server platform	Log onto DSS/SDSRV host
2	type **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsSrRmesdt mode DescriptFileName1 …**	Press **Return**

26.11.3 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.

26.11.3.1 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:

4.	**Platform**	**Server**
5.	p0acs03	/usr/ecs/TS1/CUSTOM/bin/DSS/EcDsScienceDataServer
6.	p0ins01	/usr/ecs/TS1/CUSTOM/bin/CSS/EcSbSubServer
7.	p0ins02	/usr/ecs/TS1/CUSTOM/bin/IOS/EcIoAdServer
8.	p0ins02	/usr/ecs/TS1/CUSTOM/bin/DMS/EcDmDictServer

9. If the ESDT to be added already exists, it will be removed from the four involved databases
10. 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**

- This brings up the Science Operator Dataserver Gui GUI. Refer to figure 26.11.3.1-1

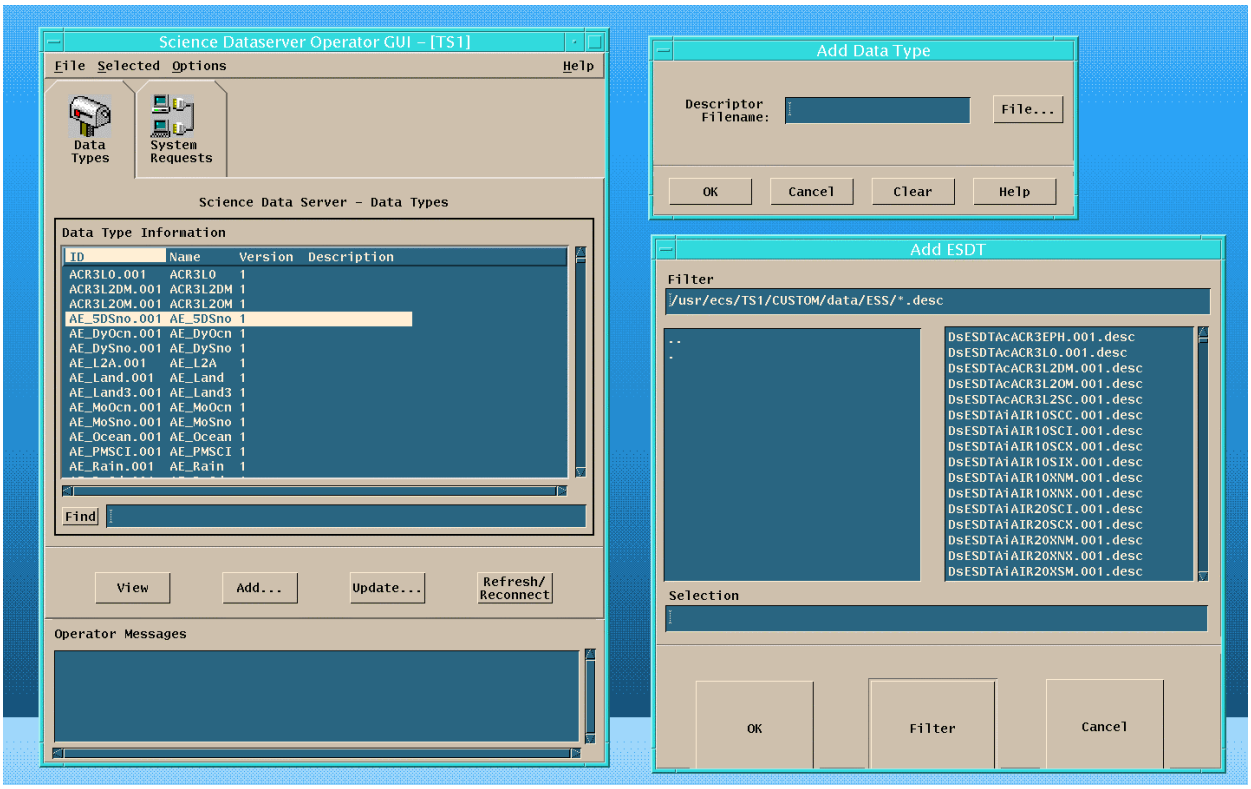


Figure 26.11.3.1-1. Adding An ESDT - The GUIs Involved

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 26.11.3.1-1. 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 **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsSdSrvGuiStart <*MODE*>**	Press **Return**
	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 **OK** in the "Add ESDT" GUI	none
8	Click **OK** in the "Add Data Type" GUI	none

26.11.4 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
9 attributes (exceptions: AdditionalAttributeName, AdditionalAttributeType,
10 AnalysisShortName, CampaignShortName, InstrumentShortName,
11 PlatformShortName, SensorCharacteristicName, SensorCharacteristicType,

12 SensorShortName, ShortName, VersionID, and type)

13 Change a mandatory attribute to optional

14 Modify parameters in existing services

One thing an ESDT Update explicitly can't do:

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

26.11.4.1 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:

Platform	**Server**
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 26.11.4.1-1

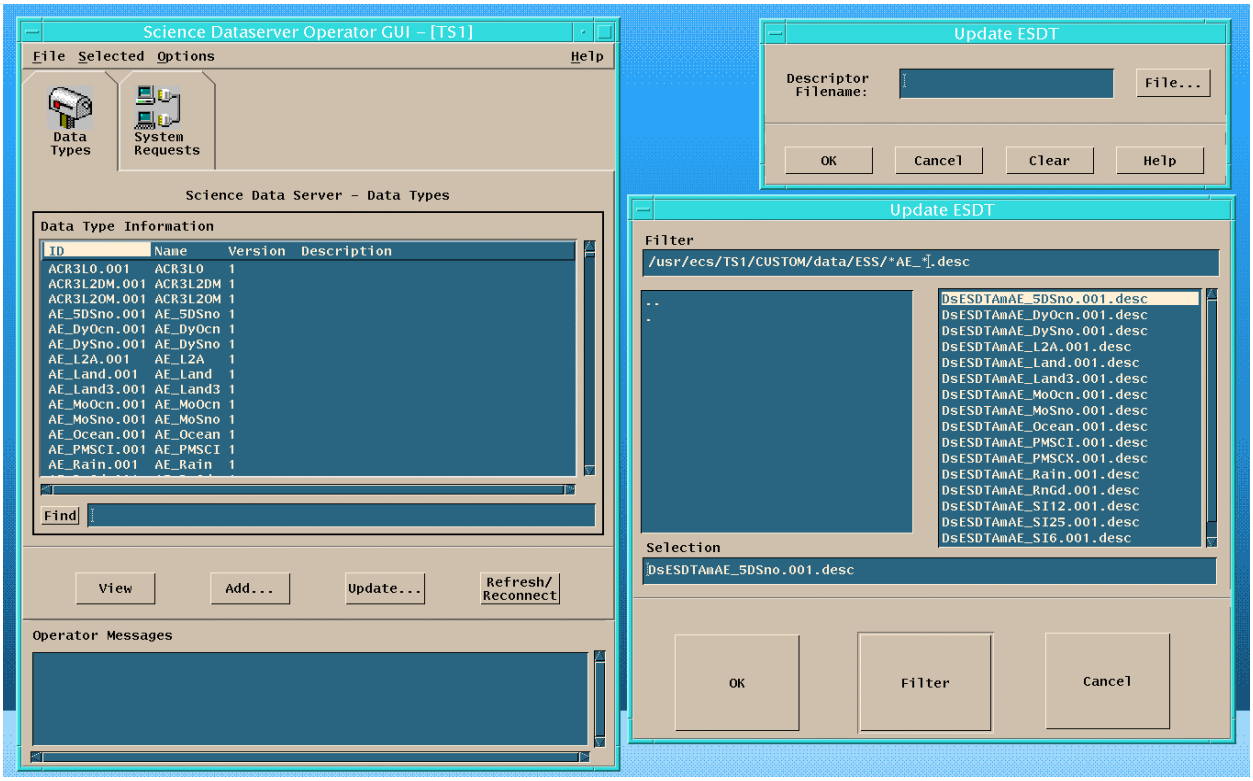


Figure 26.11.4.1-1. Updating An ESDT - The GUIs Involved

4 Click the "Update…" button on the Science Dataserver Operator Gui GUI.
- This brings up the small "Update ESDT" GUI. Refer to figure 26.11.4.1-1

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 26.11.4.1-1

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 Updatedd 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 26.11.3.1-1. 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 **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsSdSrvGuiStart <*MODE*>**	Press **Return**
	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 **OK** in the large "Update ESDT" GUI	none
8	Click **OK** in the small "Update ESDT " GUI	none

26.11.5 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 additonal 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-Rev*xx*.xls" .

26.11.5.1 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 26.11.5.1-1

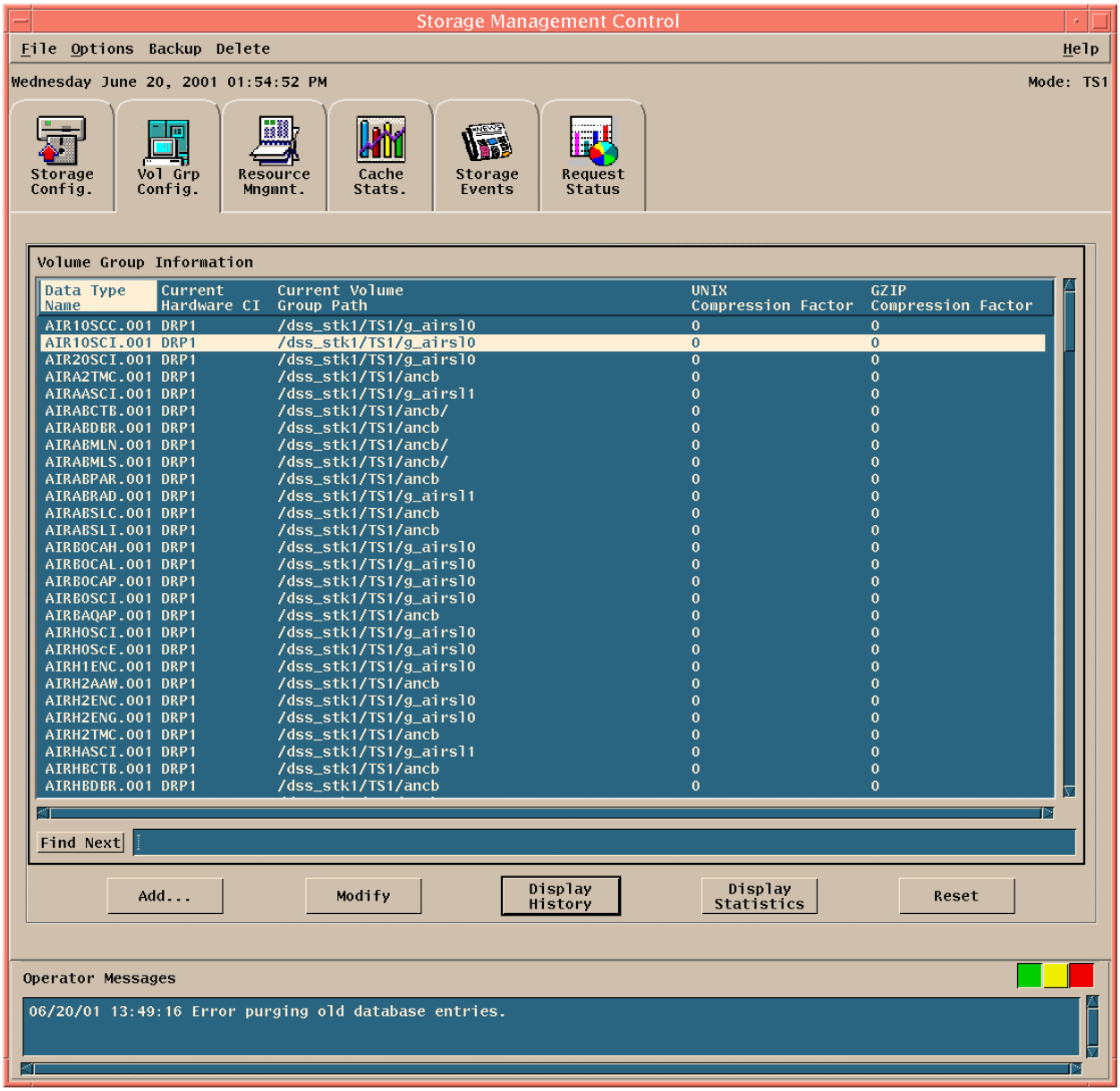


Figure 26.11.5.1-1. Storage Management GUI

4 Click the "Vol Grp Config" tab on the GUI

1. This brings up the "Volume Group Information" pane which is what figure 26.11.5.1-1 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 26.11.5.1-2

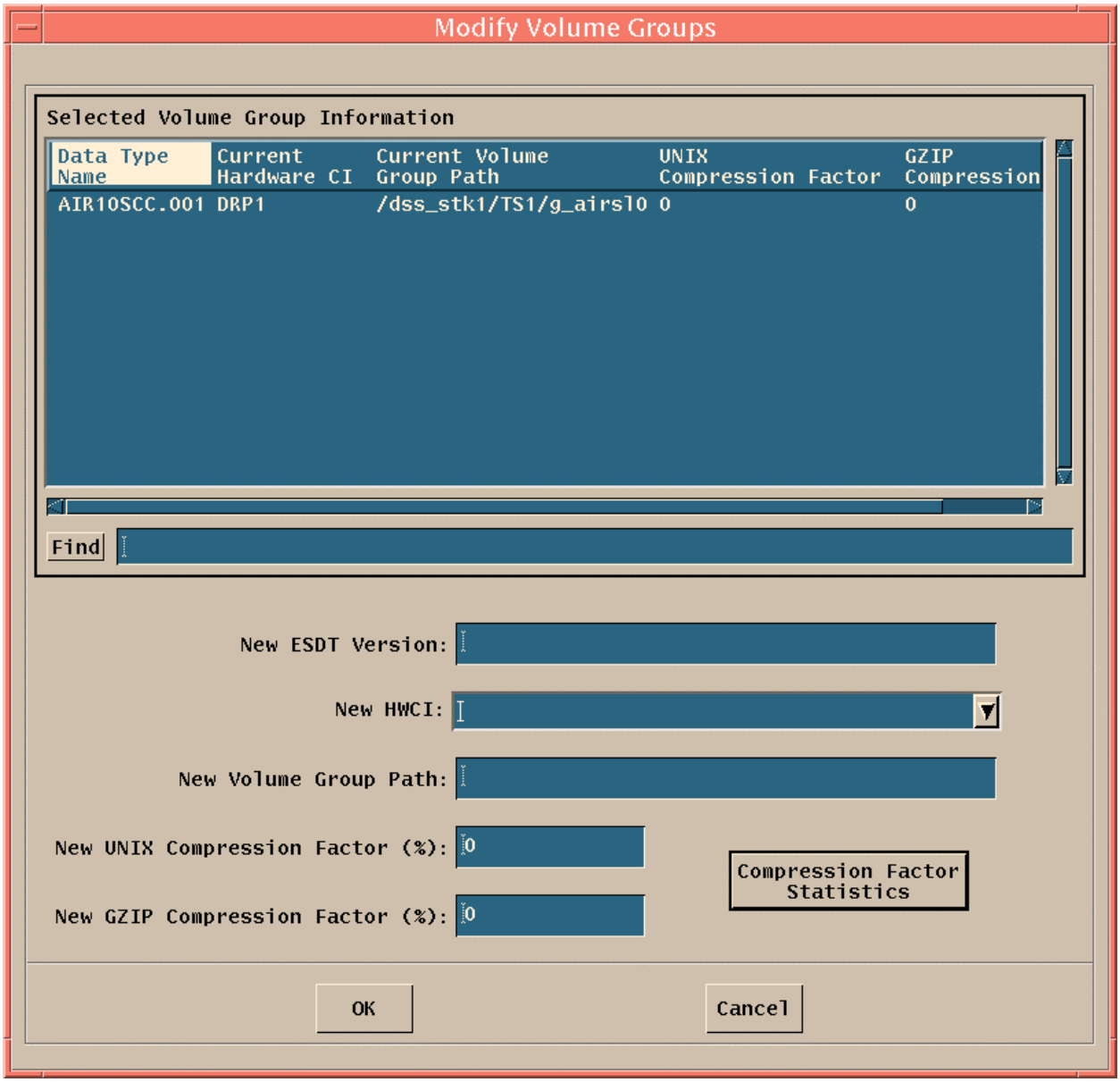


Figure 26.11.5.1-2. Modify Volume Groups GUI

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 26.11.5.1-3 .

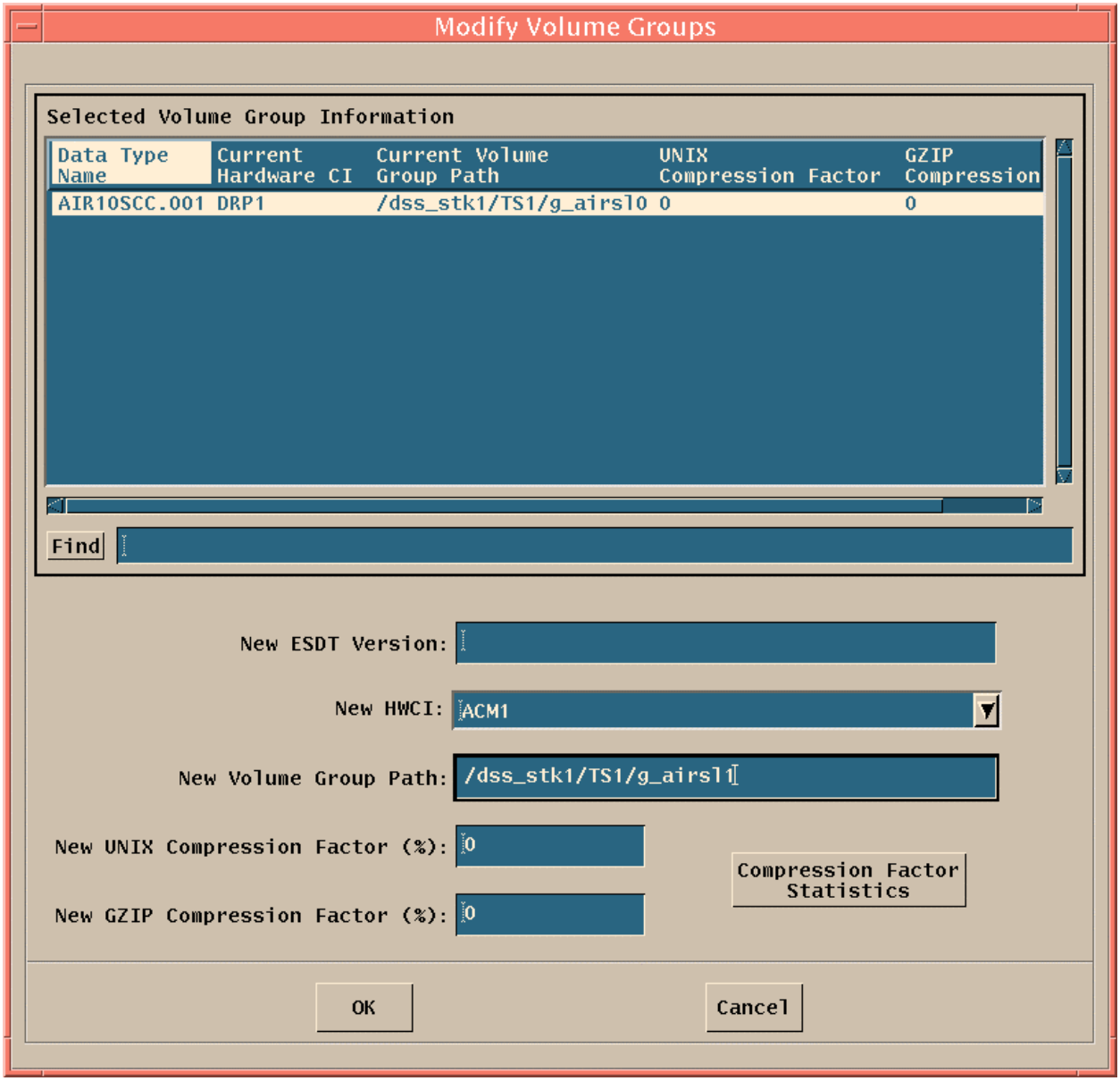


Figure 26.11.5.1-3. Modify Volume Groups GUI with entries

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 26.11.5.1-1. 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 **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsStmgtGuiStart <*MODE*>**	Press **Return**
	Stmgt Control GUI appears	none
4	Click the "**Vol Grp Config"** tab	none
5	Click on the ***ShortName.VersionID*** to be modified	none
6	Click the **Modify** button	none
	The Modify Volume Groups GUI will appear	
7	Type in the new information	none
8	Click **OK** button	none

26.11.5.2 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_srvr .

To enter information for an ESDT not already entered into the Storage Magaement'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 26.11.5.

4 Click the "Vol Grp Config" tab on the GUI

- This brings up the "Volume Group Information" pane which is what figure 26.11.5.1-1 actually illustrates.

5 Click the "Add…" button.

- The Add Volume Group GUI will appear

See figure 18.5.2-1

•

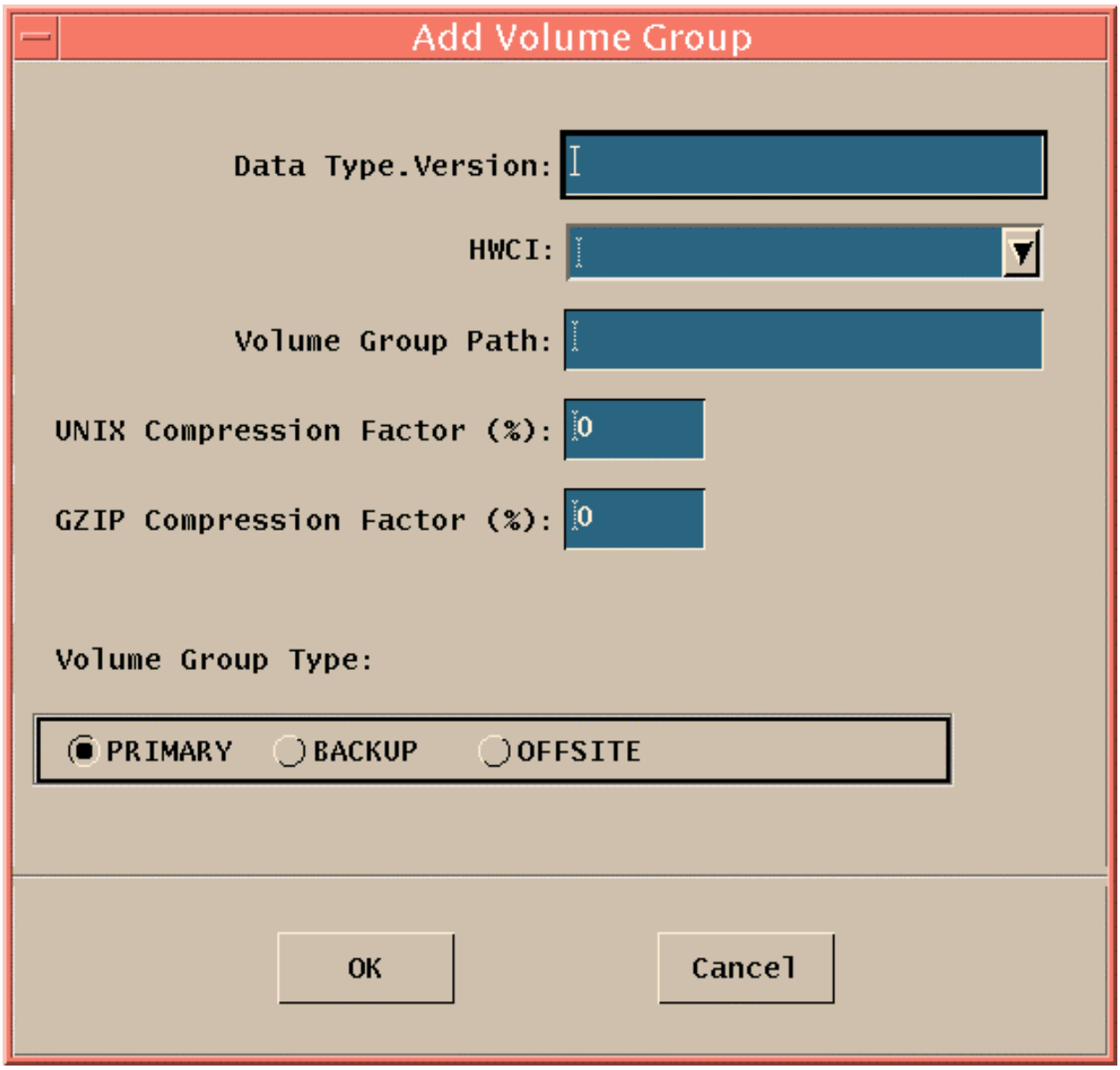


Figure 26.11.5.2-1. Add Volume Group GUI

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 26.11.5.2-1. 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 **cd /usr/ecs/<*MODE*>/CUSTOM/utilities**	Press **Return**
3	type **EcDsStmgtGuiStart <*MODE*>**	Press **Return**
4	Click the "**Vol Grp Config"** tab	none
5	Click the "**Add…"** button	none
6	Enter all the requisite information	none

26.12 Production Planning Considerations

1 During normal operations it is expected that the Production Planner will not have to add PRs to the PDPS database very frequently. The frequency of this activity is, to some extent, determined by the SCF responsible for the science software.

- The PR is a template request to generate a particular data product and results in a production run of the associated SCF-provided PGE.
- PR specifies a range (temporal, orbit, or tile) over which the data products are to be produced or the PGEs are to be scheduled.
- PR might request that the data product be produced for only a single day's data.
- PR might request that data products be produced for every opportunity of input data for several months, resulting in several hundred jobs being planned and run as the input data become available.
- Early in a mission the SCF may prefer to request processing for a short time period only (e.g., a week or less).
- At that time the SCF is gaining an understanding of the on-orbit behavior of the instrument, the resulting data, and the interaction of the science processing software with real data.
- SCF reviews the quality of the products and notifies the Production Planner of the need for any changes to the PR (e.g., discontinue the PR, change time ranges, or modify input parameters).
- When the SCF has developed a good understanding of the instrument's behavior, the team may be comfortable requesting processing for months at a time.
- DAAC operations may have operational reasons for wanting to issue processing requests for a more limited time period.

2 The Production Planner has to balance the various considerations when determining whether or not to create or update a PR.

Planning decisions are made on the basis of locally defined planning strategies for supporting the SCFs' data processing needs. The production planning tools are intended to be flexible enough in their design to support the particular planning and scheduling cycles of the operations organization at each DAAC.

Before planning production the Production Planner must coordinate with the Resource Planner to resolve all resource allocation issues. The Resource Planner notifies the Production Planner of the resources available for use in processing. Furthermore, the Production Planner may well have direct access to the Resource Plan.

The Production Planner prepares monthly and weekly production plans. In addition, the Production Planner develops a daily production schedule from the most current weekly plan. However, the first step in the planning process is creating production requests using the Production Request Editor.

26.12.1 DPREP Considerations

DPREP (data preprocessing) is a set of three PGEs that are supplied by ECS, unlike most PGEs, which are provided by the Science Computing Facilities that ECS supports. DPREP consists of the following three PGEs:

- EcDpPrAm1EdosEphAttDPREP_PGE (Step 1).
- EcDpPrAm1FddAttitudeDPREP_PGE (Step 2).
- EcDpPrAm1FddEphemerisDPREP_PGE (Step 3).

The PGEs run separately and in a particular sequence.
Three files describe the PGEs and how to run them:

- "DPREP_README"
- "HowToCreateDprepTarFile"
- "HowtoRunDPREP"

The files are installed on the science processor hosts (e.g., e0spg01, g0spg01, l0spg01, n0spg03) in the **/usr/ecs/*MODE*/CUSTOM/data/DPS** directory.
The DPREP PGEs process Level Zero (L0) Terra (AM-1) spacecraft data (e.g., ESDT AM1ANC) provided by EDOS. The output files/granules of the DPREP PGEs are subsequently used in the processing of data from various instruments on the satellite. They provide the following types of ancillary (non-science) data:

- Ephemeris
- Spacecraft location: ephemeris (or orbit) data include: latitude, longitude, and height.
- Attitude
- Orientation of the satellite, including yaw, pitch, and roll angles; and angular rates about three axes.
- There are two profiles for DPREP PGEs:
- Profile 1 runs routinely at the DAACs using previous DPREP output in addition to new Terra ancillary (e.g., AM1ANC) data.
- Profile 2 (the boot-up procedure) takes in the Terra ancillary data only and is run under two sets of conditions:
- First run of DPREP (because there is no previous output) to initialize DPREP processing.
- Following any long period of time during which EDOS L0 ancillary data are unavailable. (Short gaps in the ephemeris data are filled by EcDpPrAm1EdosEphemerisRepair, one of the executables in the EcDpPrAm1EdosEphAttDPREP_PGE.)

In order to run Profile 2 successfully following a long period of data unavailability, DPREP must be told where to resume orbit counting. The initial orbit number in the Step 1 process control file (PCF), must be set to the orbit number corresponding to the timestamp at which data availability resumes.
Until an automated process can be implemented, whenever there is a telemetry drop-out, a member of the DAAC science support team takes the following actions:

- Calls the Flight Operations Team (FOT).

- Asks for the on-line engineer.
- Requests the orbit number that coincides with the start time of the first L0 ancillary data set that follows the data drop-out.
- Sets the orbit number in the Step 1 PCF.

Then Profile 2 can be run successfully. Afterward, routine operations can be resumed using Profile 1 PGEs.

26.12.2 DPREP Considerations Post Terra Launch

As previously mentioned DPREP (data preprocessing) consists of sets of PGEs that use a statistical approach to convert Level 0 (L0) attitude and ephemeris ancillary data for a particular satellite (e.g., Terra, Aqua) into SDP Toolkit native binary format without altering or modifying the scientific content of the data sets.
DPREP PGEs are supplied by ECS, unlike most PGEs, which are provided by the Science Computing Facilities that ECS supports. Release 5B DPREP supports Terra operations and Aqua SSI&T.

26.12.2.1 Terra DPREP

Terra DPREP consists of the following three PGEs:

- AM1Eph or Step 1 (EcDpPrAm1EdosEphAttDPREP_PGE), a ksh script that serves as a driver for the following three executables:
  - EcDpPrAm1EdosAncillary
  - EcDpPrAm1EdosEphemerisRepair
  - EcDpPrAm1ToolkitToHdf
- FddAtt or Step 2 (EcDpPrAm1FddAttitudeDPREP_PGE).
- RepEph or Step 3 (EcDpPrAm1FddEphemerisDPREP_PGE).

There are several sources of information on the Terra DPREP PGEs and how to run them:

- 184-TP-001-001, Terra Spacecraft Ephemeris and Attitude Data Preprocessing.
- 611-CD-510-001, Mission Operation Procedures for the ECS Project, Chapter 11.
- Two files installed on the science processor hosts (e.g., e0spg01, g0spg01, l0spg01) in the /usr/ecs/*MODE*/CUSTOM/data/DPS directory.
  - "DPREP_README"
  - "HowtoRunDPREP"

The Terra DPREP PGEs process Level 0 Terra (AM-1) spacecraft data (e.g., ESDT AM1ANC) provided by EDOS. The output files/granules of the DPREP PGEs are subsequently used in the processing of data from various instruments on the satellite. They provide the following types of ancillary (non-science) data:

- Ephemeris.
  - Spacecraft location: ephemeris (or orbit) data include: latitude, longitude, and height.
- Attitude.
  - Orientation of the satellite, including yaw, pitch, and roll angles; and angular rates about three axes.

26.12.2.2 AM1Eph (Step 1 DPREP: EDOS Level 0 Ancillary Data)

EcDpPrAm1EdosAncillary reads in an AM-1 L0 (EDOS) ancillary data set (ESDT AM1ANC). It also reads another set of AM-1 L0 (EDOS) ancillary data set. The second set of L0 data is required to ensure that incomplete orbits in the first data set get complete orbit metadata records. The only data that is extracted from the second data set is the descending node time and longitude. EcDpPrAm1EdosAncillary also reads in ephemeris and attitude data (ESDT AM1EPHN0 and ESDT AM1ATTN0). These would be the last ephemeris/attitude data sets generated from a previous run of the PGE.
If EcDpPrAm1EdosAncillary signals that a short gap was detected, EcDpPrAm1EdosEphemerisRepair reads the scratch file created by EcDpPrAm1EdosAncillary, fills the gap, and writes a gap-filled native-format ephemeris file.
EcDpPrAm1ToolkitToHdf takes the native format ephemeris file and produces a corresponding HDF file and a metadata file.
The PGE produces Toolkit- and HDF-format attitude (ESDTs AM1ATTN0 and AM1ATTH0) and ephemeris (ESDTs AM1EPHN0 and AM1EPHH0) data sets.

NOTE: The Level 0 (input) data sets represent two-hour periods of time but processing cannot be performed on the most recently available two-hour data set. Step 1 processing needs to look forward in the time stream in order to complete orbit metadata processing.

26.12.2.3 FddAtt (Step 2 DPREP: Definitive Attitude Data)

EcDpPrAm1FddAttitudeDPREP_PGE reads in both the current Flight Dynamics Division (FDD) attitude data set and the next FDD attitude data set. It also reads in the attitude data set it produced with its last run. The output of the process is a native-format attitude file (ESDT AM1ATTNF) and an HDF-format attitude file (ESDT AM1ATTHF). A metadata file is produced for each data file.

26.12.2.4 RepEph (Step 3 DPREP: Repaired Ephemeris Data)

If Step 1 finds too many missing data points in the ephemeris data (e.g., AM1EPHH0 and AM1EPHN0 granules have gaps of greater than 60 seconds), it signals that there are problems with the data. The Production Planner receives an e-mail message indicating the problems in the preprocessed Terra ephemeris granules and notifies the FDD (e.g., by telephone) that a repaired ephemeris file (AM1EPHF) is needed for the time span of the granule that has the gap.
When the repaired ephemeris file has been ingested, EcDpPrAm1FddEphemerisDPREP_PGE reads in the repaired ephemeris data set and the EOS_AM1 ephemeris data in toolkit native format. It produces replacement ephemeris data sets (AM1EPHH0 and AM1EPHN0).
Operationally, Steps 1 and 2 are scheduled daily and run independently of one another. Step 3 is scheduled and run on an as-needed basis.

26.12.2.5 Terra DPREP Profiles

As previously mentioned DPREP processing has data requirements beyond the current two-hour segment. Data from the preceding and following segments are used for performing consistency checks on the ephemeris and attitude data streams when the data streams bridge segment boundaries. However, there is no guarantee that data from the preceding and following segments will always be available. Consequently, four data processing profiles have been developed for each of the three DPREP steps to accommodate the various permutations of data availability:

- Profile 1 is used when data are available from the preceding, current, and following segments.
- Profile 2 is used when data are available from the current and following segments only.
- Profile 3 is used when data are available from the preceding and current segments only.
- Profile 4 is used when data are available from the current segment only.

Profile 1 is used for nominal DPREP operation. It is the profile of each DPREP step that is run on a routine basis.
Profile 2 (no preceding data, but following data is available) initializes DPREP's processing of a given step's ephemeris and/or attitude data stream. When Profile 2 has been run on a data segment, Profile 1 (preceding and following data available) assumes processing responsibility on all data segments thereafter until data dropout or mission end is encountered.
Profile 3 (preceding data available, but no following data) processes the data segment that immediately precedes data dropout and, therefore, terminates processing on a given step's ephemeris and/or attitude data stream.
Profile 4 is used for processing isolated data segments and is not likely to be scheduled operationally.
In the big picture of the mission, DPREP processing on the very first data segment would require running Profile 2. The next data segments would be processed using Profile 1 processes. The very last data segment of the mission would be processed using Profile 3.

26.12.3 Aqua DPREP

There is a slide presentation at this URL below that more clearly shows the files associated with Aqua processing. http://dmserver.gsfc.nasa.gov/ecsdev/relb/pdps/dprep/step1/sld001.htm

For Release 5B Aqua DPREP consists of the following PGEs:

- PM1DefEph or PM1 Definitive Orbit PGE (EcDpPrPM1DefOrbitDPREP_PGE).
- PM1PreEph or PM1 Predictive Orbit PGE (EcDpPrPM1PreOrbitDPREP_PGE).

One source of information on the Aqua DPREP PGEs and how to run them is the "HowtoRunPm1DPREP" file installed on the science processor hosts (e.g., e0spg01, g0spg01, l0spg01) in the /usr/ecs/*MODE*/CUSTOM/data/DPS directory.

26.12.3.1 PM1DefEph and PM1PreEph (EDOS Level 0 Ancillary Data)

The Aqua PM1DefEph and PM1PreEph PGEs process Aqua ancillary data (i.e., PM1EPHD, definitive ephemeris data and PM1EPHP, predictive ephemeris data) supplied by EDOS. The output files/granules of the Aqua DPREP PGEs provide satellite ephemeris data and are subsequently used in the processing of data from various instruments on the Aqua satellite.

The PM1 Definitive Orbit PGE reads in both the current PM1EPHD definitive ephemeris data set for Aqua and a previous PM1EPHND preprocessed Aqua platform definitive ephemeris data set in native format. The outputs of the process are preprocessed Aqua platform definitive ephemeris data sets in native format (PM1EPHND) and HDF format (PM1EPHHD).
The PM1 Predictive Orbit PGE reads in both the current PM1EPHP predictive ephemeris data set for Aqua and a previous PM1EPHND preprocessed Aqua platform definitive ephemeris data set in native format. The outputs of the process are preprocessed Aqua platform predictive ephemeris data sets in native format (PM1EPHNP) and HDF format (PM1EPHHP).

26.12.3.2 Aqua DPREP Profiles

Because there is no guarantee that data from the preceding and following segments will always be available, two data processing profiles have been developed for each of the Aqua DPREP PGEs to accommodate variations in data availability:

- Profile 1 is used when both current and previous data sets are available.
- Profile 2 is used the current data set only is available.

Profile 1 is used for nominal Aqua DPREP operation. It is the profile that is run on a routine basis.
Profile 2 (no previous or next data sets) initializes DPREP's processing of an ephemeris data stream. When Profile 2 has been run on a data segment, Profile 1 assumes processing responsibility.

26.12.3.2 Additional References for Aqua DPREP

This document is an additional reference to the HowTo document that
located at
/ecs/formal/PDPS/DPS/data/HowToRunPm1DPREP
This is because some of the information, and the ODL's changes from
baseline to baseline, and from time to time. Please use this reference
along with the document at clearcase directory to run the test.

--- IMPORTANT ---

PM1 Predictive Ephemeris and Attitude are No Longer needed to run.

ESDTs needed to be installed on the Science Data Server:

PM1EPHNP
PM1EPHHD
PM1ATTHQ
PM1ATTNQ
PM1ATTHR
PM1ATTNR
PMCOGBAD
PM1EPHD

PM1EPHP
PM1EPHNP
PM1EPHHD
PM1EPHND
PM1EPHHP

Check For Environment Setup

Before running DPREP it is a good idea to do the following as these are
The most common cause of DPREP failure:

- Compare the DPREP executables. Do a diff on each science processor
from the directory:
/usr/ecs/<mode>/CUSTOM/bin/DPS/<exeName>

against the directory:
/ecs/formal/PDPS/DPS/bin/irix65/<exeName> (if running DROPxxx_irix65)
- or -
/ecs/formal/STAGE/PDPS/DPS/bin/irix65/<exeName> (if running
current_test)

If these are not the same then the science processors affected MUST be
reinstalled.

-- - Compare the ODL files in **/usr/ecs/<mode>/data/DPS/ODL** with the
baseline. If there are differences, then SSIT workstation MUST be reinstalled.

- Compare the PGS_101 file in /usr/ecs/<mode>/data with the baseline
version. If there are differences, reinstall the science processor machines.

- To check the DPREP version number of a DPREP executable enter:
what <exeName>

Creating DPREP Tar Files

If tar files are not available for registering the three PM-1 DPREP
PGEs, follow the instructions found in the **HowToCreateDprepTarFiles** file to generate
the necessary tar files.

Registering DPREP PGEs

IN STRING 2 ONLY, ALL .tar and .tar.met files have to be
ftp'ed over on the SSIT machine to

/usr/ecs/<MODE>/CUSTOM/ssit/f2spg01/data OR
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg02/data

Remember which of the above two was used because they are two different directories and that path should be used instead of **/usr/ecs/<MODE>/CUSTOM/data/DPS** in the clearcase Howto.

IN STRING 1 ftp to **/usr/ecs/<MODE>/CUSTOM/data/DPS** and use it.

Inserting test data

IN STRING 2 ONLY, All data and data.met files have to be copied over on the SSIT machine **from /ecs/formal/PDPS/DPS/data** to
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg01/data OR
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg02/data

Remember which of the above two was used because they are two different directories and that path should be used instead of **/usr/ecs/<MODE>/CUSTOM/data/DPS** in the clearcase Howto.

IN STRING 1 copy to **/usr/ecs/<MODE>/CUSTOM/data/DPS** and use it.

** Inserting PGE parameter **For the following PGE/PROFILE ID combinations, the PGE parameter Initial Orbit Number (Logical ID 998) MUST be given a number (ex, 10) .

PGE	PROFILE
PM1DefEph	2, 4
PM1PreEph	2, 4

To enter the PGE parameter, select PGE Parameter in PREditor, and select above PGEs, and enter the number in the Override Parameter box on the bottom of the screen.

26.12.3.2 Additional References for Terra DPREP

This document is an additional reference to the HowTo document that located at
/ecs/formal/PDPS/DPS/data/HowToRunAm1DPREP
This is because some of the information, and the ODL's changes from baseline to baseline, and from time to time. Please use this reference along with the document at clearcase directory to run the test.

/*********************************/

ESDTs used in the test

AM1ANC
AM1EPHN0
AM1EPHH0
AM1EPHF
AM1ATTN0
AM1ATTH0
AM1ATTF
AM1ATTNF
AM1ATTHF

VERSION ID

001

Check for Environment

- Compare the DPREP executables. Do a diff on each science processor from the directory:
/usr/ecs/<mode>/CUSTOM/bin/DPS/<exeName>

against the directory:
/ecs/formal/PDPS/DPS/bin/irix65/<exeName> (if running DROPxxx_irix65)
- or -
/ecs/formal/STAGE/PDPS/DPS/bin/irix65/<exeName> (if running current_test)

If these are not the same then the science processors affected MUST be reinstalled.

- Compare the ODL files in **/usr/ecs/<mode>/data/DPS/ODL** with the baseline.
If there are differences, then SSIT workstation MUST be reinstalled.

- Compare the PGS_101 file in /usr/ecs/<mode>/data with the baseline version. If there are differences, reinstall the science processor machines.

- To check the DPREP version number of a DPREP executable enter:
what <exeName>

Creating DPREP Tar Files

If tar files are not available for registering the three AM-1 DPREP

PGEs, follow the instructions found in the **HowToCreateDprepTarFiles** file to generate The necessary tar files.

Registering DPREP PGEs

IN STRING 2 ONLY, ALL .tar and .tar.met files have to be
ftp'ed over on the SSIT machine to
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg01/data OR
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg02/data

Remember which of the above two was used because they are two different
directories and that path should be used instead of
/usr/ecs/<MODE>/CUSTOM/data/DPS in the clearcase Howto.

IN STRING 1 ftp to **/usr/ecs/<MODE>/CUSTOM/data/DPS** and use it.

ALSO, there are 4 hour timers in the test. To avoid that problem,
a script **/home/PDPS/TestingTools/DpPrMkNewAm1Odl** to make ODLs with a
two minute delay can be used. Log in as the owner of the ODL files in
/usr/ecs/<MODE>/CUSTOM/data/DPS/ODL on the SSIT machine and issue
the command IN THE ODL directory. It will change only the necessary
files
after saving a copy as .odl.save

so run

/home/PDPS/TestingTools/DpPrMkNewAm1Odl <VER_NUM>

AFTER YOUR TEST again log in as the owner of the ODL files, go to
/usr/ecs/<MODE>/CUSTOM/data/DPS/ODL and run
/home/PDPS/TestingTools/DpPrResetDPREPOdl <VER_NUM>

** Where VER_NUM is the version number of the PGE files. To obtain the
version number,
** you can look at the Clearcase HowTo Doc.

Inserting test data

IN STRING 2 ONLY, All data and data.met files have to be
copied over on the SSIT machine from **/ecs/formal/PDPS/DPS/data** to
PGS Toolkit**/usr/ecs/<MODE>/CUSTOM/ssit/f2spg01/data** OR
/usr/ecs/<MODE>/CUSTOM/ssit/f2spg02/data

Remember which of the above two was used because they are two different
directories and that path should be used instead of

/usr/ecs/<MODE>/CUSTOM/data/DPS in the clearcase Howto.

IN STRING 1 copy to **/usr/ecs/<MODE>/CUSTOM/data/DPS** and use it.

** Inserting PGE parameter **

For the following PGE/PROFILe ID combinations, the PGE parameter Initial Orbit Number
(Logical ID 998) MUST be given a number (ex, 10) .

PGE	PROFILE
AM1Eph	2, 4
RepEph	2, 4

To enter the PGE parameter, select PGE Parameter in PREditor, and select above PGEs, and enter the number in the Override Parameter box on the bottom of the screen.

26.13 PGE Registration and Test Data Preparation

The integration of science software with 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 (both input and output) need to be placed on the Data Server. These steps must be accomplished before the science software can be run and tested within the ECS.

The following procedures describe how to register a new PGE with ECS. This involves updating the PDPS database with information needed to plan, schedule, and run the PGE. The first step in the PGE registration process is to determine which ESDTs are needed for the PGE. You must Verify that an ESDT metadata ODL file exists for each ESDT or generate an ODL file. The next step in the process is to create a PGE metadata ODL file using the delivered PCF. Finally, additional operational information (resource requirements and runtime statistics) must be input into the PDPS database. This is the last step in the PGE registration process. The order in which these procedures are done is important and should be done as indicated. Please reference Appendix C. for Examples of PGE and ESDT ODL Files for Each Instrument Team**.**

26.13.1 PGE ODL Preparation

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

---

1 From the SSIT Manager, click on the **Tools** menu, then choose **PDPS Database** and then **PCF ODL Template**.

- An xterm with title "SSIT: Science Metadata ODL Template Creation" will be displayed.

2 At the program prompt **Configuration Filename (enter for default: ../../cfg/EcDpAtCreatODLTemplate.CFG)?**

- Press **Return** for the default configuration file

3 At the program prompt **ECS mode of operations?,** type ***mode***, press **Return,** or just press **Return** if the default shown is correct.

- The ***mode*** refers to the database used and will typically be **OPS** or **TS1**.

4 At the program prompt **PCF (Process Control file) to generate template from (including full path)?**, type ***PCFpathname*/*PCFfilename***, press **Return**.

- The ***PCFpathname*** is the full path name to the location of the PCF. If not specified, the directory from which the SSIT Manager was run will be assumed.
- The ***PCFfilename*** is the file name of the PCF.

5 At the program prompt **PGE name (max 10 characters)?**, type ***PGEname***, press **Return**.

- The ***PGEname*** is the name of the PGE that will be registered.

6 At the program prompt **PGE version (max 10 characters)?**, type ***PGEversion***, press **Return** or just press **Return** if the default shown is correct.

- The ***PGEversion*** is the version of the PGE that will be registered.

7 At the prompt **PGE Profile ID (0 for Null, max 999)?,** type 1 or any valid profile ID.

- After a brief time, the message "Successfully created ODL template file" should be displayed if the task was successful.
- The program will output a file with the filename **PGE_*PGEname#PGEversion#ProfileID*.tpl**.
- For example, if the PGE name was **PGE35**, and the version and profile ID were both **1** this output file will be named **PGE_PGE35#1#01.tpl**.

8 At the program prompt **Directory for output template file (including full path)?**, type **ODLtplPathname,** press **Return.**

- The **ODLtplPathname** is the full path to the directory for the ouput template file, e.g., **/usr/ecs/TS1/CUSTOM/data/DPS.**

9 At the program prompt **Hit return to run again, 'q <return>' to quit:**, press **Return** to repeat process with another **PCF** or type **q** and press **Return** to quit.

- The xterm will disappear.

10 At a UNIX prompt on an AIT Sun, type **cd *ODLtplPathname***, press **Return.**

- The ***SSITrunPathname*** is the full path to the directory from which the SSIT Manager was run, for example **/usr/ecs/TS1/CUSTOM/bin/DPS**. This will be the directory where the file **PGE_*PGEname#PGEversion#ProfileID*.tpl** will reside.

11 At a UNIX prompt on the AIT Sun, type **cp PGE_*PGEname#PGEversion#ProfileID*.tpl PGE_*PGEname#PGEversion#ProfileID*.odl**, press **Return.**

- The **PGE_*PGEname#PGEversion#ProfileID*.tpl** is the file name of the ODL template file created in step 8.
- The **PGE_*PGEname#PGEversion#ProfileID*.odl** is the file name of a copy which can be safely edited. This file name convention must be used.

12 At a UNIX prompt on the AIT Sun, type **mv PGE_*PGEname#PGEversion#ProfileID*.odl /usr/ecs/<mode>/CUSTOM/data/DPS/ODL**

- This will place the ODL file in the directory where the executable that populates the PDPS database will read from. **PGE_*PGEname#PGEversion#ProfileID*.odl** is the file name of the copy created in step 11.

13 At a UNIX prompt on the AIT Sun, change the directory to the one in step above and type **vi PGE_*PGEname#PGEversion#ProfileID*.odl**, press **Return.**

- The **PGE_*PGEname#PGEversion#ProfileID*.odl** is the file name of the copy created in step 11.

- Any text editor may be used such as *emacs*. For example**, emacs PGE_PGE35#1#01.odl,** press **Return.**

14 In the file, add required metadata to the ODL template.

- For an explanation of what metadata is required, see file /usr/ecs/<mode>/CUSTOM/data/DPS/PGE_ODL.template.
- Note that the ShortNames typed into this file must each have a corresponding PDPS ESDT metadata ODL file.
- All objects corresponding to output ESDTs will automatically have the SCIENCE_GROUP and YIELD set during the generation of PGE ODL.
- All objects corresponding to output ESDTs will have an attribute "ASSOCIATED_MCF_ID". Place here the Logical Unit Number (LUN) listed in the PCF for the associated MCF listing.
- All objects corresponding to static input ESDTs must have the SCIENCE_GROUP set. Objects corresponding to *dynamic* input ESDTs should NOT have the SCIENCE_GROUP set.
- See Appendix E for an example of PCF and corresponding PGE ODL and ESDT ODL files.

15 Save the changes made to the ODL template file and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.

For other editors, refer to that editor's documentation

26.13.2 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 At a UNIX prompt on an AIT Sun, type **ls /usr/ecs/<mode>/CUSTOM/data/DPS/ODL/ESDT_*ShortName#Version*.odl**, press **Return**.

- The **ESDT_*ShortName#Version*.odl** is the file name of the ESDT ODL file you are looking for where ***ShortName*** is the ESDT's ShortName and ***Version*** is the ESDT version. If a file for the desired ESDT is listed, then it has already been prepared and this procedure can be exited now.
- For example, if the desired ESDT has the ShortName MOD03 and version 001, type **ls /usr/ecs/TS1/S/CUSTOM/data/DPS/ODL/ESDT_MOD03#001.odl,** press **Return**.
- If the desired file is *not* listed, continue on to step 3.

3 At a UNIX prompt on the AIT Sun, type **cd *WorkingPathname***, press **Return.**

- The ***WorkingPathname*** is the full path name to a working directory for which the user has write permissions.
- For example, **cd /home/jdoe/working/**, press **Return**.

4 At a UNIX prompt on the AIT Sun, type **cp /usr/ecs/<mode>/CUSTOM/data/DPS/ESDT_ODL.template ESDT_*ShortName#Version*.odl**, press **Return**.

- For <mode> enter the mode you are working in, for example **OPS** or **TS1**.
- The **ESDT_*ShortName#Version*.odl** is the file name of the ESDT ODL file to be created.
- This command copies a template ESDT ODL file to the ESDT ODL file to be created. The template is well commented.
- For example, type **cp /usr/ecs/<mode>/CUSTOM/data/DPS/ESDT_ODL.template ESDT_MOD03#001.odl,** press **Return**.
- The **ESDT_*ShortName#Version*.odl** file naming convention *must* be observed.

5 At a UNIX prompt on the AIT Sun, type **vi ESDT_*ShortName#Version*.odl**, press **Return**.

- The **ESDT_*ShortName#Version*.odl** represents the file name of the ESDT ODL template file created in step 4.
- Any text editor may be used such as *emacs*. For example, **emacs ESDT_MOD03#001.odl**, press **Return**.

6 In the file, add required metadata to the ODL template.

- Use the internal documentation contained in the ODL file (from the original template) to aid in populating with metadata.
- Note that the ShortName specified within the file must match the ShortName of the file name itself.
- In addition, the ShortNames used in the PDPS PGE metadata ODL file must match the ShortNames in these files.

7 Save the changes made to the ESDT metadata ODL file and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
- For other editors, refer to that editor's documentation.

8 Next type **mv ESDT_*ShortName#Version.odl* /usr/ecs/<mode>/CUSTOM/data/DPS/ODL.**

- This will place the just created ESDT ODL file in the directory where PDPS will read it from.

9 Repeat steps 1 through 8 for each ESDT required by a particular PGE. When all ESDT metadata ODL files have been completed, continue on to next section.

26.13.3 Update PDPS/SSIT Database with PGE Science Metadata

In order to update the PDPS Database with PGE metadata, the ESDT metadata ODL files must first be prepared for each ESDT required by the PGE. This section describes how to perform the next step, running the SSIT Science Update program.

Assumptions:

1. The SSIT Manager is running.
2. The directory used for containing the PDPS PGE metadata ODL files. Nominally, this is **/usr/ecs/<mode>/CUSTOM/data/DPS/ODL.**

Updating the PDPS Database with PGE Metadata

The following is a list of tools, procedures and or assumptions:

The directory used for containing the PDPS ESDT metadata ODL files can be accessed by the following commands:

---

1 telnet to (AITTL/DPS) **p0ais01** or a machine that matches the SSIT Manager host.

2 login: **ID**, password:

3 ***setenv DISPLAY…….:0.0***

4 The directory used for containing the PDPS ESDT metadata ODL files. is **/usr/ecs/<mode>/CUSTOM/data/DPS/ODL**

5 From the SSIT Manager, click on the **Tools** menu, then choose **PDPS Database** and then **SSIT Science Metadata Update**.

- An xterm with title "SSIT: Science Metadata Database Update" will be displayed.

6 At the program prompt **Configuration Filename (enter for default: ../../cfg/EcDpAtRegisterPGE.CFG)?**

- Press **Return**.

7 At the program prompt **ECS mode of operation?**, type ***mode***, press **Return** or just press **Return** if the default shown is correct.

- The ***mode*** refers to the database used and will typically be **OPS** or **TS1.**

8 At the program prompt **PGE name (max 10 characters)?**, type ***PGEname***, press **Return**.

- The ***PGEname*** is the name of the PGE that will be registered. This name must match the PGE name specified.

9 At the program prompt **PGE version (max 10 characters1)?**, type ***PGEversion***, press **Return** or just press **Return** if the default shown is correct.

- The ***PGEversion*** is the version of the PGE that will be registered. This version must match the PGE version specified.

10 At the program prompt **PGE Profile ID (0-999, 0 means null)?** Type in a valid profile ID and press **Return,** or if already listed just press **Return.**

- The PDPS database will then be updated with the information contained in the file **PGE_*PGEname#PGEversion#ProfileID*.odl**

11 At the program prompt **Hit return to run again, q <return> to quit:,** press **Return** to update the PDPS database with another PGE ODL metadata file or type **q** and press **Return** to quit.

- If you make a mistake entering any values, press **Return** here; your previous entries are restored as defaults and you won't have to retype them.
- NOTE: If you make mistakes while editing the PGE and ESDT ODL files, you can run the ODL checker (Tools → PDPS Database → Check ODL) via the SSIT manager to locate any errors.

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

Sample of ESDT.odl files being established in ECS

home/emcleod/MODIS/STORE/PGE07
p0ais01{emcleod}10: ls
ESDT_MD10L2#001.odl MOD_PR10 pge_cfg
ESDT_MD35L2#001.odl MOD_PR10.mk scf_cfg
ESDT_MOD02H#001.odl PGE07.mk script
ESDT_MOD03#001.odl doc
p0ais01{emcleod}11: cp ESDT_MD10L2#001.odl ESDT_MD35L2#001.odl
ESDT_MOD02H#001.odl ESDT_MOD03#001.odl /usr/ecs/OPS/CUSTOM/data/DPS/ODL/

26.13.3.1 AlternativeTool for SSIT Metadata Update:

Source the buildrc file for the mode in which you are working (*source .buildrc)*.
/usr/ecs/<MODE>/CUSTOM/utilities, Note that this only has to be done once per login. Then **(*cd* /usr/ecs/<MODE>/CUSTOM/bin/DPS)**

(The tool can also be executed by being in the /**usr/ecs/<MODE>/CUSTOM/bin/DPS** and executing **EcDpAtDefinePGE..)**

Shell script prompts user for information.

1 Enter in the location of the configuration file **(../../cfg/EcDpAtRegisterPGE.CFG).**
2 Filename Enter the MODE of operation (<MODE>).
3 Enter name of PGE (it must match what is in the PGE ODL file).
4 Enter the version of the PGE (it must match what is in the PGE ODL file).
5 Enter the Profile ID (it must match what is in the PGE ODL file). Note that the ODL file for the PGE must have the of: PGE_<PGE NAME>#<PGE VERSION>#<PROFILE ID>.

- Each ODL file is displayed as it is processed. A good status message should be displayed as a result. Information about the PGE (inputs and outputs, Production Rules, etc) should be entered in the Database.

26.13.3.2 Examples of PGE and ESDT ODL Files for Each Instrument Team

This section is taken from the latest **Green Book 162-TD-001-007** and are listed in Appendix C. Depicted are examples of ODL files in SSI&T activities. Then, examples of specific ODL files are listed by instrument (ASTER, MISR or MODIS).

Template ODL Files

There are five Template ODL files listed therein. The specific or tailored ODL files listed were derived from these templates by appropriate editing and filling-in of values. The three 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
PATHMAP_ODL.template
TILE_ODL.template

Example of a successful PDPS Science Metadata Update:

PDPS/SSIT SCIENCE Metadata Database Update **
Configuration filename? (enter for default: ../../cfg/EcDpAtRegisterPGE.CFG)
ECS Mode of operations? (enter for default: OPS)
OPS
PGE name (max 10 characters)?
PGE07
PGE version (max 10 characters)?
001
PGE Profile ID (0-999, 0 means null)? (enter for default: 1)
1
Warning: Could not open message catalog "oodce.cat"
EcDpAtRegisterPGE: Process Framework: ConfigFile
../../cfg/EcDpAtRegisterPGE.CFG ecs_mode OPS
' PGE profile id = '1' ...
Do you wish to overwrite the previous PGE PGE07((y)es or (n)o):
y
FILES PROCESSED
: PGE SCIENCE ODL file =/usr/ecs//OPS/CUSTOM/data/DPS/ODL/PGE_PGE07#0#001.odl
ESDT SCIENCE ODL file = /usr/ecs//OPS/CUSTOM/data/DPS/ODL/ESDT_MOD02H#001.odl
ESDT SCIENCE ODL file = /usr/ecs//OPS/CUSTOM/data/DPS/ODL/ESDT_MD35L2#001.odl
ESDT SCIENCE ODL file = /usr/ecs//OPS/CUSTOM/data/DPS/ODL/ESDT_MOD03#001.odl
ESDT SCIENCE ODL file = /usr/ecs//OPS/CUSTOM/data/DPS/ODL/ESDT_MD10L2#001.odl
********** Update of PDPS/SSIT database with PDPS SCIENCE metadata SUCCESSFUL **********
Hit return to run again, 'q <return>' to quit:

26.13.4 Operational Metadata

The SSIT version of the PDPS database is initialized and updated with SSIT Operational Metadata so that the Planning and Processing Subsystem can schedule and run PGEs. Here, PDPS Operational Metadata refers to PGE information 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. 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 26.13.2.

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.

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

---

1 From the SSIT Manager, click on the **Tools** menu, then choose **PDPS Database** and then **SSIT Opnl Metadata Update**.

- The PDPS/SSIT Database Update GUI, **"dpAtOpDbGui"** will be displayed.

2 Click on the radio button labeled **NEW PGE** in the lower left quadrant.

- The PGE that you are working on should appear in the subwindow labeled **PGE Names** along with its version number in the subwindow labeled **PGE Versions.**

3 In the subwindow labeled **PGE Names**, click on a PGE name. Then in the subwindow labeled **PGE Versions**, click on the PGE version for that PGE. Then click on the button labeled **EDIT**.

- The PGE name and version will be highlighted when you click on them.
- The page tabs **PROFILE**, **RUNTIME**, and **ESDT** will change from gray (indicating disabled) to black (indicating enabled).
- To see the contents of PGE Metadata, click on the button labeled **DISPLAY** and then click on the button labeled **DONE.**
- If the PGE name and/or version does not appear in the lists, it means that updating of PDPS database with PGE metadata was not successful.

4 Click on the **PROFILE** page tab.

- The Profile page will be displayed.

5 In the fields under the label **Performance Statistics**, enter the information specified.

- In the field labeled **Wall clock time**, enter the amount of wall clock time it takes for one execution of the PGE, in seconds. The tab **PROFILE** will change from black (indicating enabled) to red (indicating database needs to be updated by APPLY button).

- In the field labeled **CPU time (user)**, enter the so-called *user* time of the PGE, in seconds. This value should come from profiling the PGE .
- In the field labeled **Max memory used**, enter the maximum amount of memory used by the PGE, in megabytes (MB). This value should come from profiling the PGE .
- In the fields labeled **Block input ops** and **Block output ops**, enter the integer number of block inputs and block outputs, respectively. These values should come from profiling the PGE .
- In the field labeled **Swaps**, enter the integer number of page swaps from the PGE. This value should come from profiling the PGE .
- In the field labeled **Page faults**, enter the integer number of page faults from the PGE. This value should come from profiling the PGE .

6 In the fields under the label **Resource Requirements**, enter the information specified.

- In the field labeled **Runtime Directory Disk Space**, enter the maximum amount of disk used by the PGE during execution, in megabytes (MB), for staging MCF files, system PCF file, Profile file and generating 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.

26.13.5 SSIT Operational Metadata Update GUI

The SSIT version of the PDPS database is initialized and updated with SSIT Operational Metadata so that the Planning and Processing Subsystem can schedule and run PGEs. Here,

PDPS Operational Metadata refers to PGE information 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 (Figure 26.13.5-1). The program then writes the data directly to the SSIT version of the PDPS database. The SSIT Operational Metadata Update GUI is used to view or update the following operational parameters for a particular PGE:

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

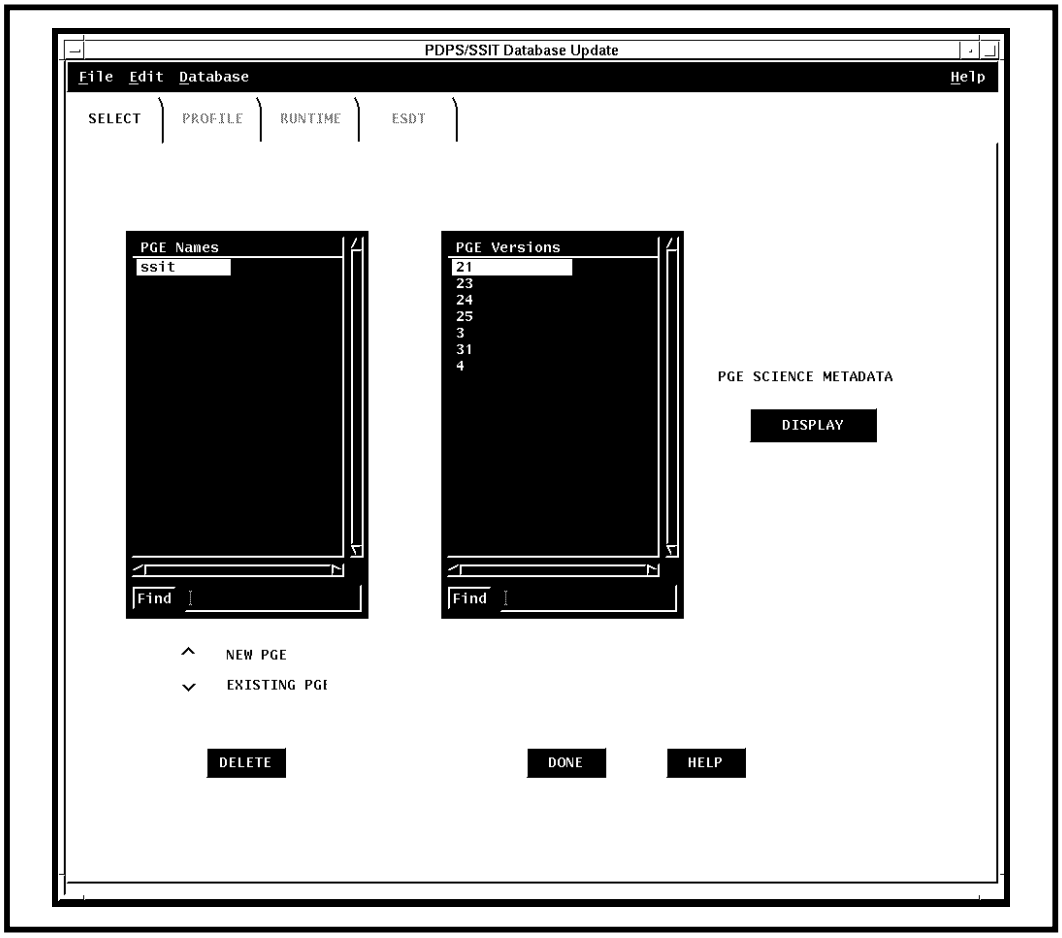


Figure 26.13.5-1. SSIT Database Operational Metadata Update GUI – SELECT view

26.13.6 Test Data Preparation and Insertion of Data Granules

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

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

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

26.13.6.1 Generating a Metadata Configuration File (Source MCF)

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

Assumptions:

- The SSIT Manager is running.
- ESDT's are installed onto the **Science Data Server.**

To Generate the Metadata Configuration File (Source MCF) for the input and output ESDT's, execute the steps that follow.

---

1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Get MCF.**

- An xterm in which EcDpAtGetMCFis running will be displayed as SSIT: Acquire MCF..
- Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **EcDpAtGetMCF.sh**, press **Return**.

2 At the program prompt **Configuration Filename (default *defaultConfigFile*)?**

- Type in **../../cfg/** ***defaultConfigFile*** and press **Return**.
- The ***defaultConfigFile*** will be replaced by the full path name and file name of the default configuration file. The file name will be **EcDpAtGetMCF.CFG** where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.

3 At the program prompt **ECS mode of operation (enter for default: def*aultMode*)?**, type ***mode***, press **Return** or just press **Return** if the default shown is correct.

- The ***mode*** refers to the database used and will typically be **TS1**.

4 At the program prompt **ESDT Short Name?**, type ***ESDT ShortName***, press **Return**.

- The ***ESDTShortName*** is the name of the ESDT that the EcDpAtGetMCF tool will use to generate the MCF.

5 At the program prompt **ESDT Version?**, type ***ESDTversion***, press **Return** or just press **Return** if the default shown is correct.

- The ***ESDTversion*** is the version of the ESDT.

6 At the program prompt **Directory to receive MCF (must be full path)?**, type ***MCFpathname***, press **Return**.

- The ***MCFpathname*** is the full path name to the location where the source MCF will be placed. For example, /home/jdoe/ssit.

7 To the final prompt **Hit return to run again, 'q <return> to quit:**, press **Return** to generate another Source MCF or type **q** and press **Return** to quit.

- If you make a mistake entering any values, press **Return** here; your previous entries are restored as defaults and you won't have to retype them.

Example of a successful installation of a Source MCF:

Configuration filename? (enter for default: ../../cfg/EcDpAtGetMCF.CFG)
ECS Mode of operations?
OPS
ESDT Short Name?
MOD03EM
ESDT Version?
0
Directory to receive MCF? (must be full path)
/home/emcleod/MCF/
Warning: Could not open message catalog "oodce.cat"
EcDpAtGetMCF: Process Framework: ConfigFile ../../cfg/EcDpAtGetMCF.CFG
ecs_mode
OPS
incomplete group entries in the configfile,using default G1
Request for MCF successful for:
ESDT name = 'MOD03EM'
ESDT version = '0'
directory = '/home/emcleod/MCF/'
Hit return to run again, 'q <return>' to quit:

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

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

In standalone or isolation testing of a PGE, the inputs needed by it will not have been Inserted by a previous PGE in the chain. This Insertion must be done manually. A Target MCF file for a

corresponding data granule is required to run a standalone PGE. This way a dynamic data granule can be Inserted to the Science Data Server as if a PGE had produced it.

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

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

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.

26.13.7.1 Creating a Metadata File for a Static Granule

---

1 At the UNIX prompt on the AIT Sun, type cd WorkingPathname, then press the **Enter** key.

- Example: **cd /usr/ecs/{MODE}/CUSTOM/data/DPS/ODL/**
- The ***WorkingPathname*** *is* the full path name of the working directory containing the template metadata ODL file.

2 At the UNIX prompt on the AIT Sun, type **cp *StaticODLmet.tpl filename*.met**, then press the **Enter** key.

- The ***StaticODLmet.tpl*** is the file name of the template Target MCF.
- The ***filename*.met** is the file name of the Target MCF for this static file. The file name extension must be **.met**.
- This command will copy the template Target MCF to ***filename*.met**. For example, type **cp StaticODLmet.tpl CER11T.mcf.met**, then press the **Enter** key.

3 At a UNIX prompt on the AIT Sun, type **vi *filename*.met**, then press the **Enter** key.

- This command invokes the *vi* editor and reads in the Target MCF created above.

4 Edit the Target MCF with the specific information for the static data granule to be Inserted. The following guidelines should be followed when editing on the template MCF:

- The value for the ShortName object should be filled out with proper instrument name.
- The value for the Version ID object should be filled out with the proper version number.
- In the **INFORMATIONCONTENTCONTAINER** object enter the following:
  - The value for the **PARAMETERNAME** object of the class **"1"** should be filled out with the name of static data file.
  - The value for the **PARAMETERVALUE** object of the class **"2"** should be filled out based on the following guideline:

- If the data granule is a coefficient file, a **"C"** followed by a numerical number **n (n=1,2,….)** will be used. Here n stands for the number of the coefficient file.
- If the data granule is a MCF file, a "M" followed by a numerical number **n (n=1,2,….)** will be used. Here n stands for the number of the MCF file.

5 Save the changes made to the Target MCF (***filename*.met**) and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, then press the **Enter** key.

26.13.8 Inserting Static Data Granules Into The Data Server

Inserting a Static Data File:

- The following Servers/Services must be up and operational:
- **Science Data Server, Storage Management.**
- The following must have occurred between those Servers/Services:
- The ESDT of the static file must have been installed at the Data Server.

What the user must do before trying SSIT functionality:

- Create a metadata file for the static file to insert. To do this, an MCF (See "Getting an MCF in this section") must be gotten from the Data Server for the ESDT of the file to insert. Mandatory fields are filled into the MCF, creating a metadata file.
- If the tool is NOT run from the SSIT Manager then go to the executables directory **(cd /usr/ecs/<MODE>/CUSTOM/utilities)**
- Source the buildrc file for the mode in which you are working **(source .buildrc).** Note that this only has to be done once per login.
- If the tool is NOT run from the SSIT Manager then go to the executables directory (**cd /usr/ecs/<MODE>/CUSTOM/bin/DPS)**

From the **SSIT Manager** choose **Tools** menu and then **Data Server** submenu. Choose **Insert Static File.**

The tool can also be executed by being in the /**usr/ecs/<MODE>/CUSTOM/bin/DPS and executing EcDpAtInsertStatic**

Shell script prompts user for information.

1 Enter in the location of the DpAtInsertStaticFile configuration file **(../../cfg/EcDpAtInsertStaticFile.CFG).**

2 Enter the MODE of operation **(<*MODE*>).** At the program prompt **mode (default ops)?**, or press **Enter** to take default.

3 Enter the short name of the ESDT (for the static file). This value is in the pdps database under the PlDataTypeMaster table and must be in the PGE ODL file.

- At the program prompt **ESDT name?** type ***ESDTShortName***, then press the **Enter** key. For example type: **MOD02LUT.**

4 Enter the version of the ESDT for the static file. This value is also in the pdps database under the PlDataTypeMaster table and must be in the PGE ODL file.

- At the program prompt **PGE version (default 1)?**, type ***PGEVersion***, then press the **Enter** key.
- The ***PGEVersion*** must match exactly the PGE version entered into the PDPS for this PGE.

5 Enter the science group for this static (this will be from the ODL created during Populating the PGE information in the Database).

- At the program prompt **Science group for Static file(one of{C,L,D,O} followed by a 4 digit number)?**, type ***ScienceGroupID***, then press the **Enter** key.
- 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 which share the same ESDT. For instance, for a coefficient file, use **C*n***, where number ***n*** could be **0, 1, 2…;** this number *n* needs to be matched with the number *n* in the PGE_PGEName#Version.odl file. For an MCF. For example, type **C001**, press **Return.**
- The Science Group ID must match what was edited into the PGE metadata ODL file for that PCF entry.

6 At the program prompt **Is there more than one data file for this Static (Y = Yes, N = No)? (enter for default: N).** If there is only one data file, press **Return** and go to next step. If there are more than one data files, type **Y,** press **Return** and go to step 10.

7 At the program prompt **Single Static Filename to Insert (including FULL path)?**, type ***pathname*/*GranuleFileName***, press **Return**

- The ***pathname*/*GranuleFileName*** is the full path name and file name of the static data granule to be Inserted. For example, type **/home/MODIS/PGE10/MOD_PR28/coeff/emissivity.dat**, press **Return**.

8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path).** Type ***pathname*/*GranuleFileName.met*)?**, press **Return**.

- The ***pathname*/*GranuleFileName.met*** is the full path name and file name of the .met file for the associated static data granule to be Inserted. For example, type **/home/MODIS/PGE10/MOD_PR28/MOD28LUT.met** press **Return**.

9 At the program prompt **Directory where all data files and .met file exist (FULL path)?** Type ***pathname*** press **Return**.

- where ***pathname*** is the full path of the directory where all data files and .met file exist.
- Note for a multifile granule, the data files and .met file should be placed in the same working directory.

10 At the program prompt **Name of MFG file (enter to end list)?** Type in the ***GranuleFileName,*** one at a time and press **Return.** To end the list press **Return.**

- Where ***GranuleFileName*** is the names of the multifile granules.

11 At the program prompt **Associated ASCII Metadata Filename to Insert?** Type ***GranuleFileName.met***, press **Return.**

- Where ***GranuleFileName.met*** is the name of one **.met file** that is used with all data granules in the even of a multifile granule.

- The dynamic data granule will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.

12 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.

- If continuing, repeat steps 2 through 9.

26.13.9 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 ESDT's have been installed on the Data Server.
2. The Target MCF for this data granule has been created for the Insert.

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

1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Insert Test Dynamic**.

- An 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. For example, **TS1** or **OPS.** Press **Return.**

4 At the program prompt **ESDT short name for the file(s) to insert?** type ***ESDTShortName***, press **Return**

- The ***ESDTShortName*** is the ShortName of the ESDT descriptor file corresponding to this granule to be inserted. For example, type **MOD021KM** press **Return**.

5 At the program prompt **ESDT Version for the file(s) to insert?** Type in the ESDT version and press **Return.**

6 At the program prompt **Is there more than one data file to this Dynamic Granule (Y = Yes, N = No)? (enter for default: N)?** If there are no multifiles for this ESDT, press **Return** and go to step 7. If there are more than one file for this granule go to step 9.

7 At the program prompt **Single Filename to Insert? (including FULL path)** type ***pathname/GranuleFilename***, press **Return.**

- The ***pathname/GranuleFileName*** is the full path name and file name of the data granule to be inserted. For example, type **/home/MODIS/PGE10/MOD021KM.A1996217.0014.002.hdf**, press **Return**.

8 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path)** , Type ***pathname/GranuleFileName.met*** and press **Return**.

- ***pathname*** is full name of the path and ***GranuleFileName.met*** is the name of the associated .met file. For example, **/home/MODIS/PGE10/MOD021KM.met**
- The dynamic data granule will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.

9 At the program prompt **Directory where all data files and .met file exist (FULL path)?** Type ***pathname*** press **Return**.

where ***pathname*** is the full path of the directory where all data files and .met file exist.
Note for a multifile granule, the data files and .met file should be placed in the same working directory.

10 At the program prompt **Name of MFG file (enter to end list)?** Type in the ***GranuleFileName,*** one at a time and press **Return.** To end the list press **Return.**

where ***GranuleFileName*** is the names of the multifile granules.

11 At the program prompt **Associated ASCII Metadata Filename to Insert?** Type ***GranuleFileName.met***, press **Return.**

where ***GranuleFileName.met*** is the name of one .met file that is used with all data granules in the even of a multifile granule.

- The dynamic data granule will be inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.

12 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.

- If continuing, repeat steps 2 through 8.

Example of a successful insertion of a Dynamic Input Data Granule into the Data Servers:

PGE Test Dynamic Input File Insertion **
Configuration filename? (enter for default:
../../cfg/EcDpAtInsertTestFile.CFG)
ECS Mode of operations? (enter for default: OPS)
OPS
ESDT name
MOD02H
ESDT Version (enter for default: 1)
0
Staged Filename to Insert? (including FULL path)
/home/emcleod/MCF/MOD02HKM.A1997217.1730.002.hdf
Associated ASCII Metadata Filename to Insert? (including FULL path)
/home/emcleod/MCF/MOD02H.met
Warning: Could not open message catalog "oodce.cat"
EcDpAtInsertTestFile: Process Framework: ConfigFile
../../cfg/EcDpAtInsertTestFile.CFG ecs_mode OPS
incomplete group entries in the configfile,using default G1

Trying to make a request to [MDC:DSSDSRV]
incomplete group entries in the configfile, using default
Trying to make a request to [MDC:DSSDSRV]
incomplete group entries in the configfile, using default
Insert to Data Server successful:
ESDT Version = '0'
staged file = '/home/emcleod/MCF/MOD02HKM.A1997217.1730.002.hdf'
metadata file = '/home/emcleod/MCF/MOD02H.met'
Inserted at UR:
'UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[MDC:DSSDSRV]:16:SC:M
OD02H:1757'
Hit return to run again, 'q <return>' to quit:

26.13.10 Science Server Archive Package (SSAP)

The SSAP is used to provide a record of the science software, documentation, and other related files stored at the DAAC. The SSIT SSAP GUI provides a method for grouping required data about a PGE.

The SSAP is not to be confused with the Delivered Algorithm Package (DAP) received from the SCF. Much of what is in the DAP will make it into the SSAP. The key difference is that SSAP data is prepared after initial testing of the science software and will include data that reflects site integration as well as fixes required for performance at the DAAC.

The SSAP is made up of 2 different data types. The first data type is the Algorithm Package which contains metadata (name of the PGE, name of the instrument, date accepted, etc...) about the SSAP. The second data type is the source code, documentation, and test data which will be stored as a SSAP, with its own metadata in addition to the files. SSAP components such a source code will be tared to retain the directory structure.

The executables and static files are stored separately from the SSAP and will have their own data types (ESDTs).

Before continuing on we recommend that a review be made of the latest SSAP documention contained in the Release 5B/6A internal Interface Control Document for the ECS Project 313-CD-510-002. Here you will find step by step procedures that cover the various Thread Components you may encounter.

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.

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

1. The SSIT Manager is running.
2. The PGE has been successfully built with the SCF and DAAC version of the Toolkit.

26.13.10.1 Creating a SSAP

The following Servers/Services must be up and operational:

Science Data Server, Storage Management.

The following must have occurred between those Servers/Services:

NONE.

What the user must do before trying SSIT functionality:

1 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). 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.
The SSAP GUI will be displayed.

2 Click on the **Create** to create a new SSAP.

- The **Create SSAP** window appears. If no OK button is visible, resize the window such that the OK button is visible.

3 Enter the name of the SSAP in the first field . Enter SSAP version in the second field. Note that version has a limit of 20 characters.

4 Click OK and the window disappears

3 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.

5 To set up the SSAP components, click on the **File List** tab.

- 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.

6 To select a file in the left column, click on the **File Type** button, highlight a file (or files) and click on the **Add** arrow button to add the files..

The files selected to be added will be displayed in the right column.

T o change directories (and thus add files from other directories to the SSAP component), click on the listing in the window on the bottom left of the GUI. The ".." is to go up one directory level. A single click with move to the directory chosen and change the display to show the directories under the new current directory. Note that the list of files in the upper left window changes to show the files within the current directory.

7 To add metadata for the new SSAP, select the **Metadata** tab.

The Metadata window will be displayed.

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:

8 To change the default information, click on the **Edit Assoc Collections** button.

The **Edit Associated Collections** window displays a list of associated collections and fields for the entry of new ShortNames and Versions.

9 Enter a ShortName, and version (of the 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.

10 Enter the version (of the installed ESDT).

Then select the **OK** button. Select **Done** to close the window.

11 To save the updated metadata, click **Save** on the **Metadata** tab**.**

12 To get back to the **Main** tab**,** select the **Main** tab button.

13 To submit the new SSAP to the Data Server, select the **Submit** button.

When the SSAP has been submitted, the **SSAP Successfully inserted to the Data Server** prompt will appear.

The SSAP Editor is covered in more detail in the 313-CD-510-002 Release 5B ECS Internal Interface Control Document for the ECS Project, section 3.8.6, where Insert Thread Components are covered. Information on the SSAP Editor is also available in 609-CD-003-004 Operations Tools Manual for the ECS Project, section 4.5.1.2.9.6.

26.13.11 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

3 From the SSIT Manager choose *Tools* menu and then *Data Server* submenu. Choose *SSAP Editor*.

- The GUI starts (Figure. 26.13.11-1). 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 (Figure. 26.13.11-2). 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.

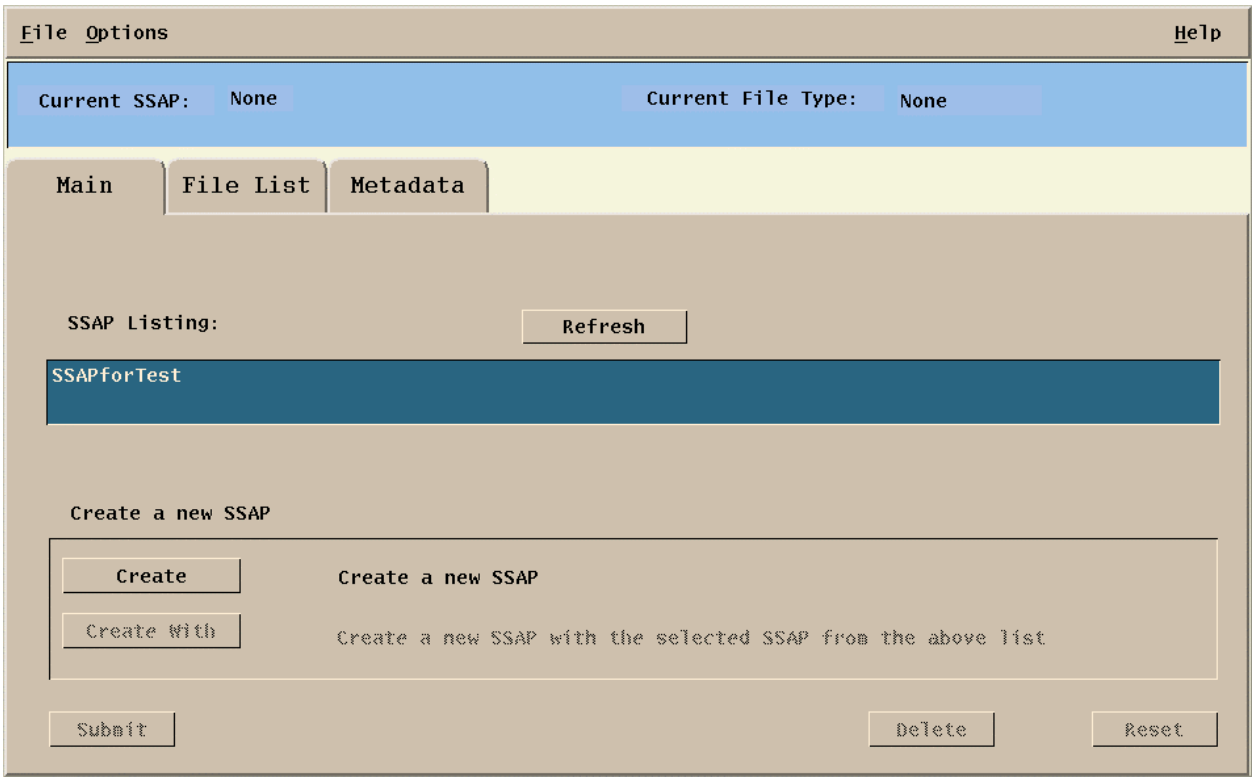


Figure 26.13.11-1. SSAP Main Gui.

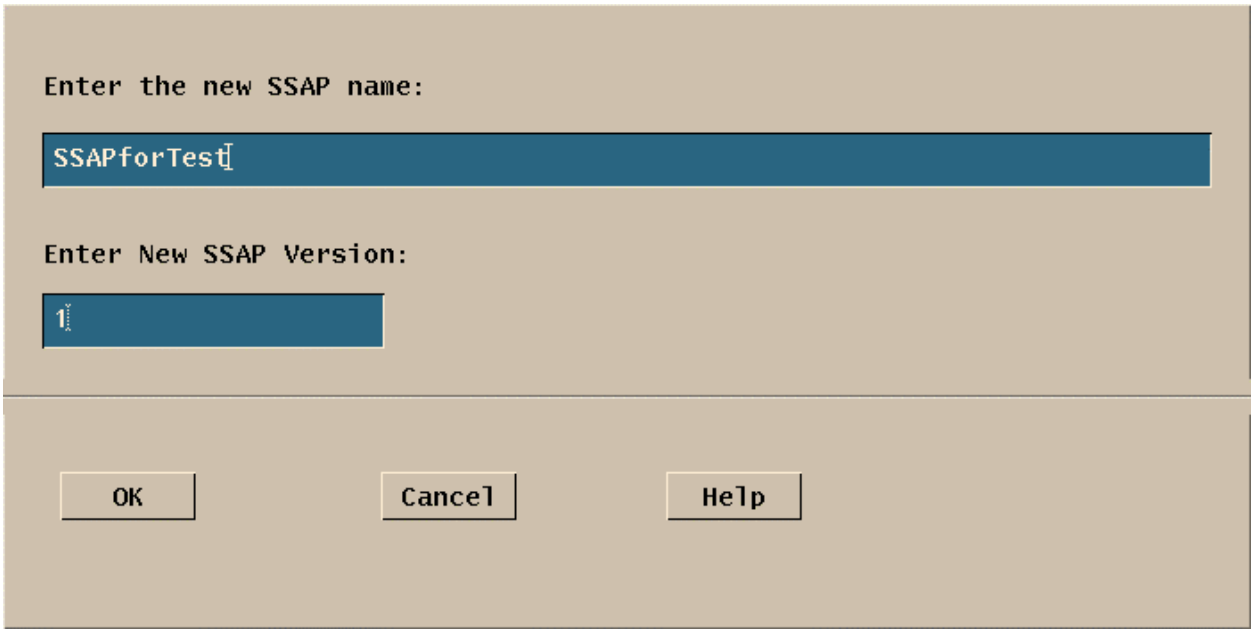


Figure 26.13.11-2. Window to define new SSAP

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 (Figure. 26.13.11-3) 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.

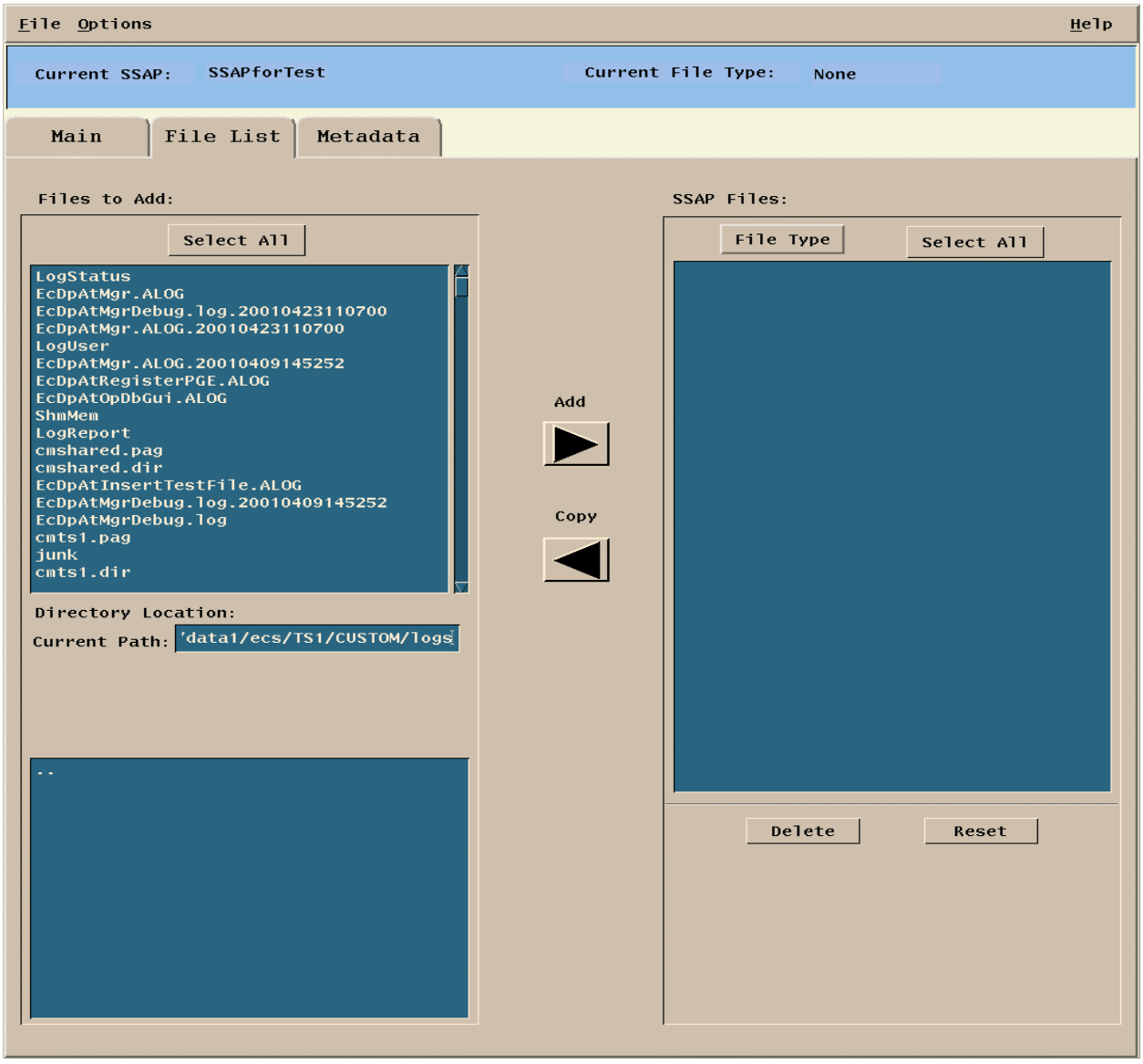


Figure 26.13.11-3. File list tab for file manipulation in defining new SSAP

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 (Figure. 26.13.11-4) 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.

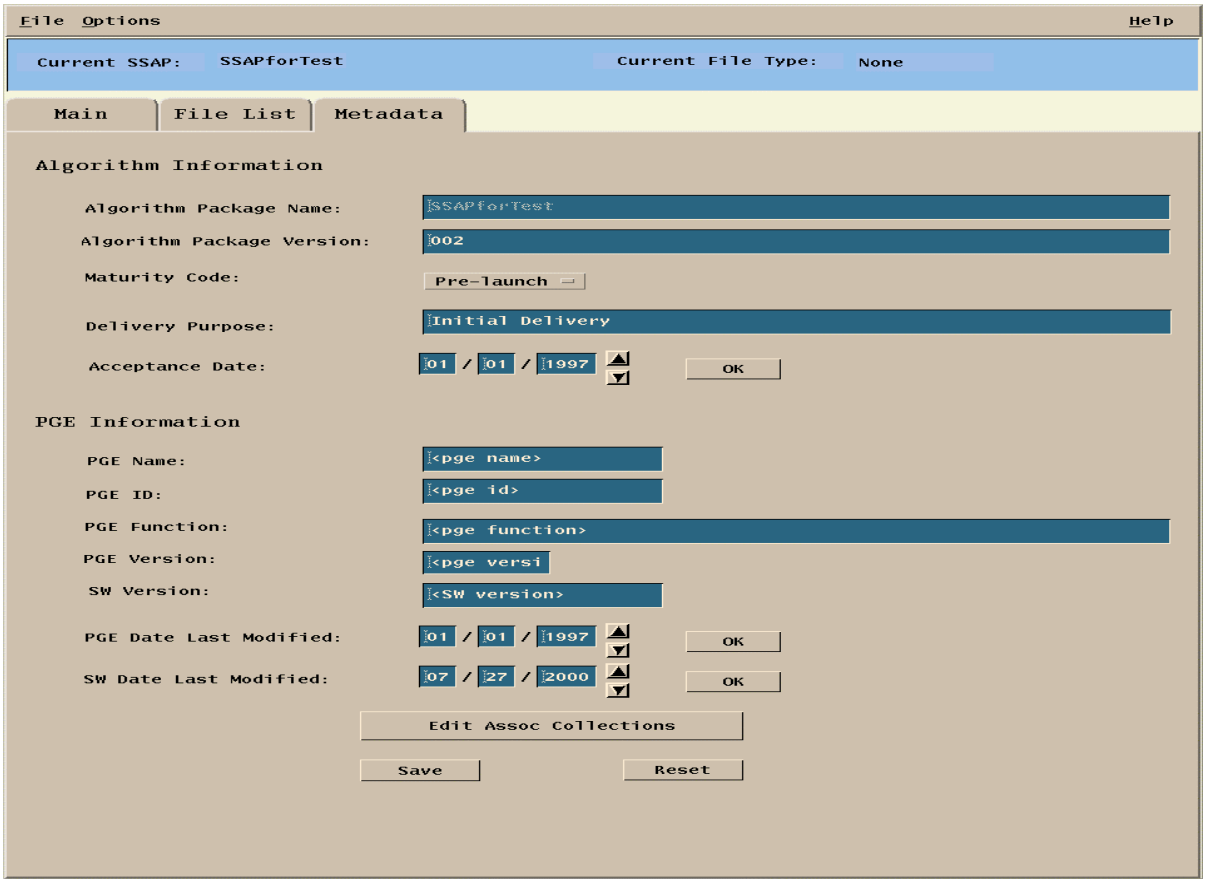


Figure 26.13.11-4. Metadata tab to enter metadata for new SSAP

14 Click the *Edit Assoc. Collections* button.

- The Edit Associated Collections window (Figure. 26.13.11-5) displays a list of associated collections and fields for the entry of new ShortNames and Versions (which make up an Associated Collection).

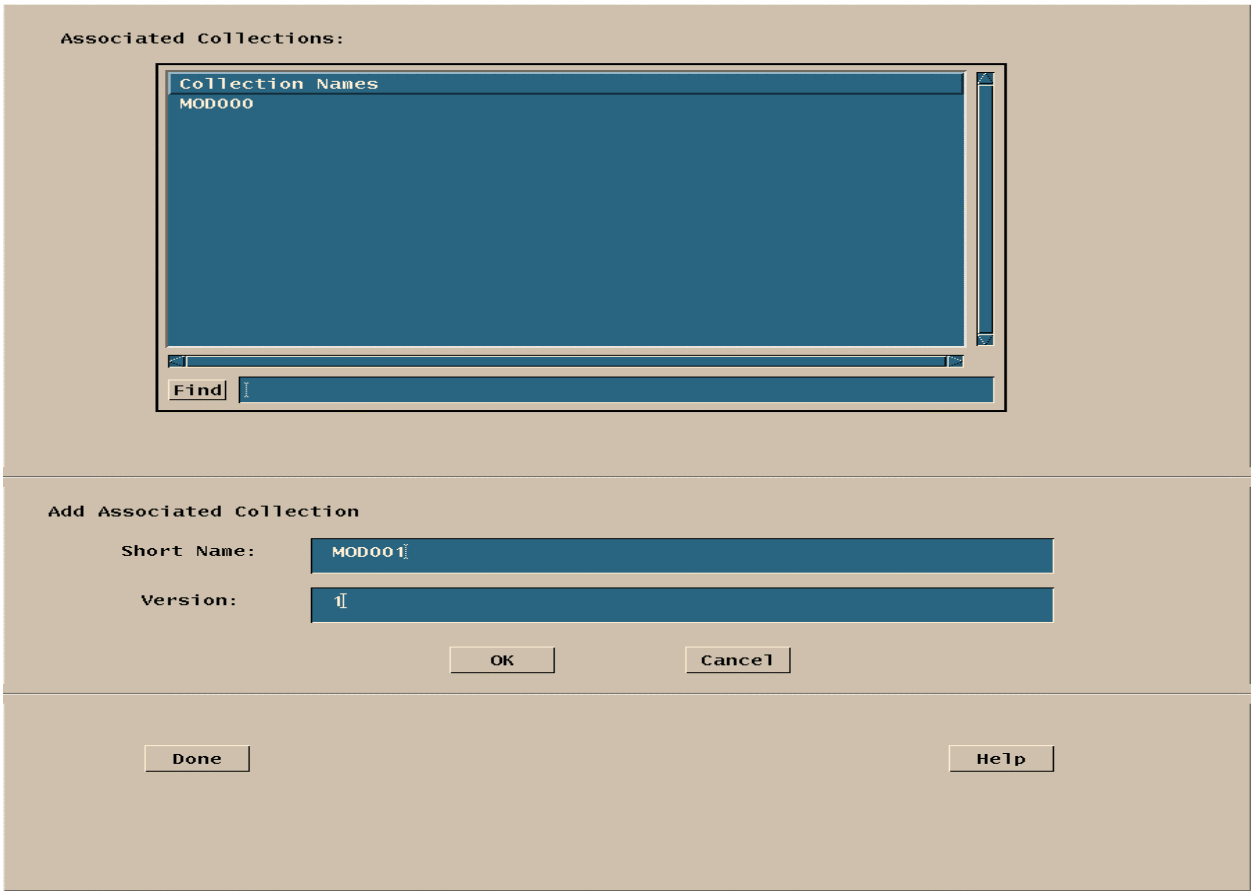


Figure 26.13.11-5. The Edit Associated Window

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".

26.13.12 Updating a Science Software Archive Package (SSAP)

The following Servers/Services must be up and operational:

Data Server, Storage Management.

The following must have occurred between those Servers/Services:

An SSAP must have already been inserted to the Data Server.

What the user must do before trying SSIT functionality:

The SSAP Editor has been used to insert an SSAP to the Data Server.

What must be done via SSIT tools:

If SSAP Editor is not running, use the directions from the first 2 paragraphs of (Creating an SSAP) to bring up the SSAP GUI. Note that the added SSAP should appear in the window of the Main tab.

If the SSAP Editor is already running, the added SSAP should appear in the window of the Main tab.

---

1 Click on **added SSAP** in the main display.

2 Click on the **Metadata tab** to update the SSAP.
The Metadata Tab displays the metadata for the SSAP. All fields will be set to the values entered when the SSAP was created, and the Algorithm Name field will be grayed out (because it may not be updated). If you want to create a new SSAP from the an existing one, go back to the Main tab and hit the Create With button.

3 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).

4 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.

5 Before you leave the Metadata tab, click Save to save the updated metadata.

6 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.

7 Click on the File Type button to select the additional SSAP component to manipulate.
Choose one of the menu items.
Select a file (or files) from the left window to add to the component.
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.
Click Main to get back to the Main tab.
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 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."

26.13.12.1 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 (Figure.26.13.12-1). 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.

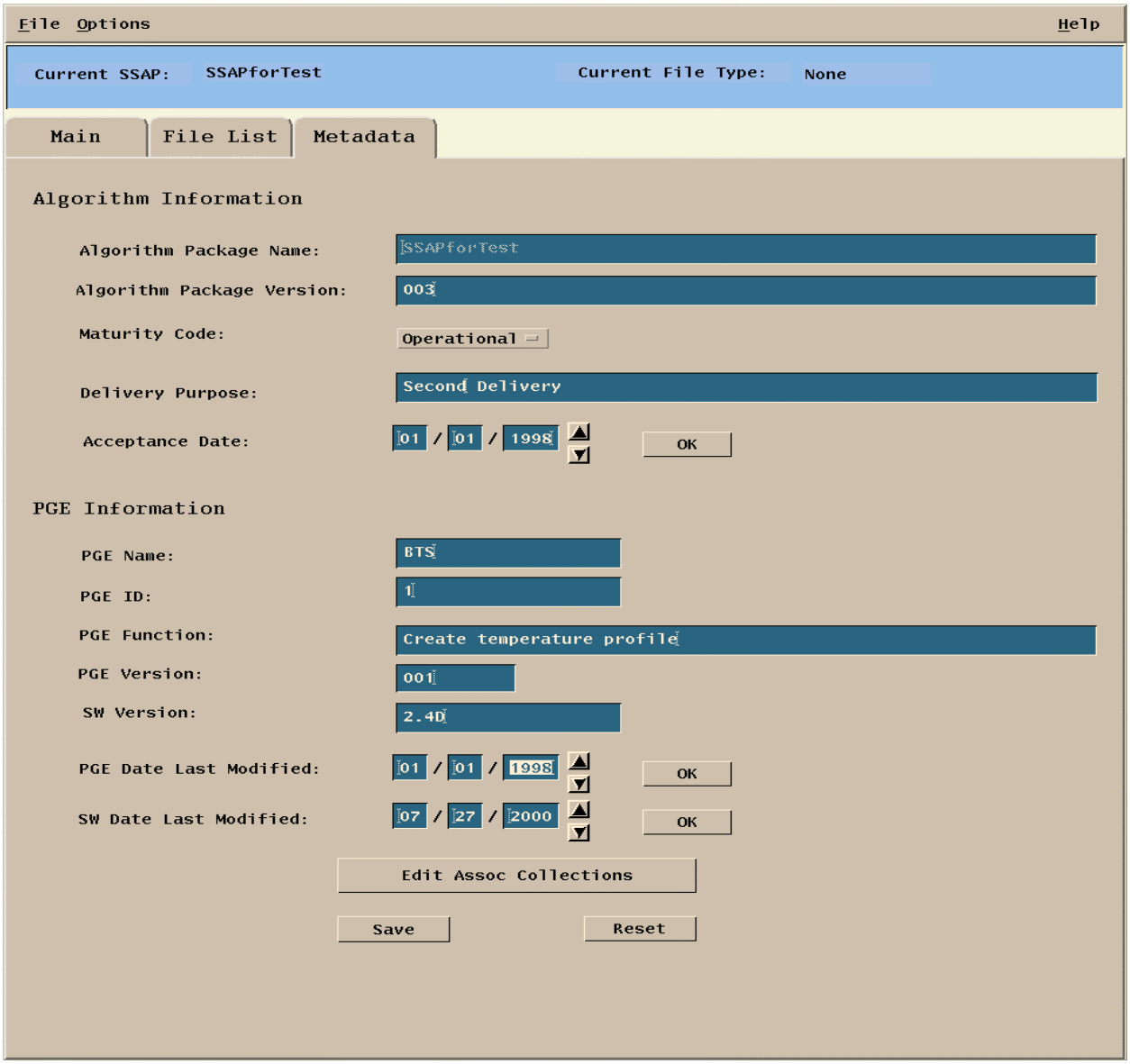


Figure 26.13.12-1. Metadata window to for updating metadata

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.

26.13.13 Placing the Science Software Executable (SSEP) on the 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.

26.13.13.1 Assembling a Science Software Executable Package (SSEP)

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

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

- The ***SSEPpathname*** is the full path name of a *new* directory which will contain all the files to be placed into the SSEP as well as the SSEP itself.
- It is recommended that ***SSEPpathame*** be named with a convention that indicates the PGE for which a SSEP will be created. For example, type **mkdir PGE35.ssep**, press **Return**.

2 At the UNIX prompt on the AIT Sun, type **cd *SSEPpathname***, press **Return**.

- The ***SSEPpathname*** is the directory name of the new directory created in step 1.

3 At the UNIX prompt on the AIT Sun, type **cp *pathname/file1 pathname/file2 … pathname/filen* .**, press **Return** (note the "dot" and then space at the end of the command).

- The ***pathname/file1, pathname/file2,…pathname/filen*** represents a list of path names and file names (delimited by spaces) to copy into the current directory, ***SSEPpathame*** (the "dot" represents the current directory and must be last in the command).
- For example, type **cp /data/MODIS/pge/PGE35.exe /data/MODIS/mcf/mod35.mcf /data/MODIS/MOD_13453 .**, press **Return** (note the space and then "dot" at the end of the command).
- The files copied into this directory should be the PGE executable, any shell scripts or other executables that are part of the PGE and SDP Toolkit message files.
- Files can be individually copied into the ***SSEPpathame*** directory. For example, type **cp /data/MODIS/pge/PGE35.exe .**, press **Return** (note the space and then "dot" at the end of the command). Repeat for each file needed in the SSEP for this PGE.

4 At the UNIX prompt on the AIT Sun, type **tar cvf *SSEPfilename*.tar ***, press **Return**.

- The ***SSEPfilename*.tar** is the file name for the SSEP tar file. The file name extension .tar is recommended but not required.

- The asterisk (*****) is a file name wildcard that represents all files in the current directory. This will place all files in the SSEP tar file.
- Once created, the contents of the SSEP tar file can be viewed by typing **tar tvf *SSEPfilename*.tar**, press **Return**.
- Do not apply compression (*e.g.* UNIX compress or gzip) to the tar file.

5 At the UNIX prompt on the AIT Sun, type **cp *filename.met.tpl filename.met,*** press **Return**.

- The ***filename.met.tpl*** is the file name of the template Target MCF for this SSEP. If a template is not available, see Appendix D or use one used for another SSEP.
- The ***filename.met*** is the file name of the Target MCF to be tailored for this SSEP.

6 At the UNIX prompt on the AIT Sun, type **vi *filename.met***, press **Return**.

- The ***filename.met*** is the Target MCF for this SSEP.
- This command invokes the *vi* editor. Edit the ***filename*.met** with the specific information for the SSEP to be inserted.
- The following guidelines should be followed when editing on the Target MCF (***filename*.met**):
  - The value for the VERSIONID object should be filled out with the proper PGE version. For example: "1" .
  - In the INFORMATIONCONTENTCONTAINER object,
    - The value for the PARAMETERNAME object of the class "1" should be filled out with the PGE name. For example: "BTS".
    - The value for the PARAMETERNAME object of the class "2" should be filled out with the PGE Science Software Version. For example: "1".
    - The value for the PARAMETERNAME object of the class "3" should be filled out with the Platform Name. For example: "IRIX".
    - The value for the PARAMETERNAME object of the class "4" should be filled out with the Platform Version. For example: "6.5".
    - The value for the PARAMETERNAME object of the class "5" should be filled out with the date to perform the Insertion. For example: "970319".
    - The value for the PARAMETERNAME object of the class "6" should be filled out with the time to perform the Insertion. For example: "14:45:00".

7 Save the changes made to the SSEP's Target MCF (***filename*.met**) and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.

For other editors, refer to that editor's documentation

SSEP pocedures continued

1 At the UNIX prompt on the AIT Sun, type **mkdir *SSEPpathname*** then press the **Enter** key. For example, type **mkdir MOD35.ssep, press Enter.**

The ***SSEPpathname*** is the full path name of a *new* directory which will contain all the files to be placed into the SSEP as well as the SSEP itself.

2 At the UNIX prompt on the AIT Sun, type **cd *SSEPpathname***, then press the **Enter** key.
The ***SSEPpathname*** is the directory name of the new directory created in step 1.

3 At the UNIX prompt on the AIT Sun, type **cp *pathname/file1 pathname/file2 … pathname/filen* .**, then press **Enter** (note the space then the "dot" at the end of the command).
The ***pathname*** is the location of the files. The ***file1, file2,…filen*** represents a list of file names (delimited by spaces) to copy into the current directory, ***SSEPpathame*** (the "dot" represents the current directory and must be last in the command). For example, type **cd /data/MODIS/pge/MOD35.pge /data/MODIS/mcf/MOD35.mcf /data/MODIS/MOD_13453 ., press Enter.** (note the space then the "dot" at the end of the command).
For the synthetic PGE, only the executable needs to be copied.

4 At the UNIX prompt on the AIT Sun, type **tar cvf *SSEPfilename*.tar ***, then press the **Enter** key.
The ***SSEPfilename*.tar** is the file name for the SSEP tar file.
The file name extension .tar is recommended but not required.
The asterisk (*****) is a file name wildcard that represents all files in the current directory which will place all files in the SSEP tar file.
Once created, the contents of the SSEP tar file can be viewed by typing **tar tvf *SSEPfilename*.tar**, then press the **Enter** key.

5 At the UNIX prompt on the AIT Sun, type **cp *filename.met.tpl filename.met,*** then press the **Enter** key.
The ***filename.met.tpl*** is the file name of the Target MCF for this SSEP.
For the synthetic PGE, the **met** file has already been renamed and modified for use by the student when the file was unpacked.

6 At the UNIX prompt on the AIT Sun, type **vi *filename.met***, then press the **Enter** key.
The ***filename.met.tpl*** is the Target MCF for this SSEP.

7 Edit the ***filename*.met** with the specific information for the SSEP to be inserted.
The value for the **VERSIONID** object should be filled out with the proper PGE version.
In the **INFORMATIONCONTENTCONTAINER** object enter the following:
The value for the **PARAMETERNAME** object of the **class "1"** should be filled out with the PGE name. The synthetic PGE should be **"userid".**
The value for the **PARAMETERNAME** object of the **class "2"** should be filled out with the PGE Science Software Version.
The value for the **PARAMETERNAME** object of the **class "3"** should be filled out with the Platform Name.
The value for the **PARAMETERNAME** object of the **class "4"** should be filled out with the Platform Version.
The value for the **PARAMETERNAME** object of the **class "5"** should be filled out with the date to perform the Insertion.
The value for the **PARAMETERNAME** object of the **class "6"** should be filled out with the time to perform the Insertion.

8 Save the changes made to the SSEP's Target MCF (***filename*.met**) and exit the editor.

The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, then press the **Enter** key.

26.13.13.2 Insert a Science Software Exec Package (SSEP) onto 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 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 PGEEXE has been installed on the Science Data Server.
2. A Target MCF (.met) for this PGEEXE tar file has been created for the Insert.

1. The PGEEXE tar file has been created .\
2. The following Servers/Services must be up an operational:

   Science Data Server, Storage Management.

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

1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Insert EXE TAR**.

- An xterm with title "SSIT: PGE Executable Tar File Insertion" will be displayed.

2 At the program prompt **Configuration filename? (enter for default: ../../EcDpAtInsertExeTarFile.CFG)**, press **Return**.

3 At the program prompt **ECS mode of operations?**, Type <***mode***> press **Return.**

- <***mode***> can either be **OPS** or **TS1**.

4 At the program prompt **Name of PGE?**, type ***PGEName***, press **Return**.

- The ***PGEName*** is the name of the PGE for which this static granule is being Inserted. For example, type **PGE01**, press **Return**.
- The ***PGEName*** must match exactly the PGE name entered into the PDPS for this PGE.

5 At the program prompt **Science software version of PGE?**, type ***SSWversion***, press **Return**.

- The ***SSWversion*** is the version of the science software which is being Inserted in this SSEP. Press **Return** to accept the default or enter in a version and press **Return**.

6 At the program prompt **Staged filename to insert (including Full path)?**, type ***pathname/SSEPFileName***, press **Return**

- The ***pathname/SSEPFileName*** is the full path name and file name of the SSEP tar file to be Inserted. For example, type **/data/MOD35/ssep/PGE35_1.tar**, press **Return**.
- The SSEP tar file must not be compressed (*e.g.* with UNIX compress or gzip).

7 At the program prompt **Associated ASCII metadata filename to insert (including Full Path)? *pathname/SSEPFileName.met*)?**, press **Return**.

- The default is the file name of the granule to insert with the .met file name extension. If the default is not correct, then the file name of this file must be entered.

8 At the program prompt **Top level shell filename within tar file?**, type ***ExecFileName***, press **Return**.

- The ***ExecFileName*** is the file name of the top level executable or script within the SSEP tar file. It should be the same as was entered into the PDPS/SSIT Database Update GUI.
- The SSEP will be Inserted to the Science Data Server.

9 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.

- If continuing, repeat steps 3 through 8.

Example of a successful insertion of a SSEP EXE TAR:
PGE Executable Tar File Insertion Script

Configuration filename? (enter for default:../../cfg/EcDpAtInsertExeTarFile.CFG)

ECS Mode of operations? (enter for default: OPS)

OPS

Name of PGE? (enter for default: PGE07)

PGE07

Science software version of PGE? (enter for default: 0)

0

Staged filename to insert (including FULL path)? (enter for default:

/home/emcleod/SSEP/MODPGE07.tar)

Associated ASCII metadata filename to insert (including FULL path)? (enter for

default: /**home/emcleod/SSEP/MOD_PR10.tar.met**)

Top level shell filename within tar file? (enter for default:

/home/emcleod/SSEP/MOD_PR10.exe)

MOD_PR10.exe (note: this entry is done a second time) Note: If you get **core dump**, execute using "dbx command: type in: **dbx filename .exe**. This will help isolate error message that caused core dump.

```
Warning: Could not open message catalog "oodce.cat"
EcDpAtInsertExeTarFile: Process Framework: ConfigFile
../../cfg/EcDpAtInsertExeTarFile.CFG  ecs_mode OPS
Performing INSERT.....
incomplete group entries in the configfile,using default G1
Trying to make a request to [MDC:DSSDSRV]
incomplete group entries in the configfile, using default
Trying to make a request to [MDC:DSSDSRV]
incomplete group entries in the configfile, using default
Insert to Data Server and PDPS database update successful for:
   PGE name = 'PGE07'
   Ssw version = '0'
   ESDT = 'PGEEXE'
   ESDT Version = '0'
   staged file = '/home/emcleod/SSEP/MODPGE07.tar'
   metadata file = '/home/emcleod/SSEP/MOD_PR10.tar.met'
   Top level shell name = 'MOD_PR10.exe'
Inserted at UR:
'UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[MDC:DSSDSRV]:14:LM:PGEEXE:1787'
Hit return to run again, 'q <return>' to quit:
```

26.14 PGE Planning Processing and Product Retrieval

This section provides the capabilities and details necessary to carryout SSI&T on the PDPS. A more complete guideline has been created that addresses PDPS in the Operational Precedures Manual of DID 611- Production Planning and Processing section and the ECS Project Training Material Volume 6: DID 625-CD-606-001, Production Planning and Processing.

26.14.1 Using the Production Request Editor

When standalone 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.

26.14.2 Invoking the Production Request

Currently, the Production Request Editor is invoked from a command line script. In the future, this will be done by clicking on the icon for the PR Editor on the ECS Desktop. Once the Production Request Editor is invoked, it brings up a screen with five tabs at the top for selection (as shown in Figure 26.14.3-1). 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.

********* Please be advised: Only one user per mode can be executing a DPR at a time. If more than one occurs the system will cancel the remaining DPR's. Problem discovered when chained PGE's failed to kickoff. *********

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 In any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **PLS host.**

- It is recommended that this procedure begin within a new command shell on a PLS Host.

2 Set the DISPLAY environment variable. At the UNIX prompt on the PLS host (e.g. **p0pls01**), type **setenv DISPLAY** ***terminal_id***, press **Return**.

3 Set the UNIX environment variable. At the UNIX prompt on the PLS host, change to the directory where the scripts are located (e.g. **cd /usr/ecs/TS1/CUSTOM/utilities),** then **Return.**

- Type **setenv MODE** ***mode*** (e.g. TS1).
- Type **source** ***environment_setup_file*** (e.g. **EcCoEnvCsh** for C shell users).

4 At the UNIX prompt on the **PLS host** (e.g. **p0pls01**), under the directory where the scripts are located (e.g. **cd /usr/ecs/TS1/CUSTOM/utilities**), type

EcPlPRE_IFStart ***mode application_id*** **&**, then press **Return**.

- The ***mode*** is the operations mode (e.g. **TS1**).
- The ***application_id*** is a numerical number (e.g. **1**).
- For example, type **EcPlPRE_IFStart** ***TS1 1*** **&**, press **Return**.

- Various messages from the Production Request Editor may appear in this window as it is running. For this reason, avoid using this window for other tasks until the Production Request Editor has terminated.

5 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 Defining a New Production Request.
- 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.**

6 When tasks are completed in the Production Request Editor GUI, click on the **File** menu, then choose **Exit**.

- The Production Request Editor will disappear.
- Refer to Section (Troubleshooting and General Investigation) if fail to bring up the Production Request Editor GUI.

Figure 26.14.3-1. Production Request Editor Introductory GUI

26.14.3 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. A PR will explode into one or more Data Processing Requests (DPR) depending upon the time interval involved. Each DPR corresponds to the execution of a PGE. 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:

- The user has selected the **PR Edit** tab from the Production Request Editor .
- The PGE involved in the Production Request has been registered in the PDPS database.

On workstation **x0pls##**, at the UNIX prompt in a terminal window, type as in step 1 below your **user id** and **password**.

NOTE: The **x** in the workstation name will be a letter designating your site:
g = GSFC, **m** = SMC, **l** = LaRC, **e** = EDC, **n** = NSIDC, **o** = ORNL, **a** = ASF, **j** = JPL, **p**=PVC; the ## will be an identifying two-digit number (e.g.,**g0pls02** indicates a Planning Subsystem (PLS) workstation at GSFC).

Prior to the rlogin, enter **setenv DISPLAY <local_workstation IP address>:0.0.** The **<ipaddress>** is the ip address of **x0pls##,** and **xterm** is required when entering this command on a Sun terminal.

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 **Figure.** 26.14.3-2.

2 In the field labeled **PR Name:**, enter **New** or verify that **New** is already entered as the default.

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 and then on the **OK** button.
- The selected PGE will then be used to populate **Satellite Name**, **Instrument Name**, **PGE Name**, and **PGE Version** fields of the main GUI.

4 In the field labeled **Priority:**, enter ***priority***.

- The ***priority*** is the priority to be assigned to this Production Request in the range **0** through **999** with **0** being the highest priority and **999** the lowest. For example, enter **40**.

5 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.

6 In the Production Request Editor GUI, enter ***StartDate*** and ***StartTime*** in fields labeled **Begin**, respectively.

- The ***StartDate*** and ***Starttime*** are the start date and time of the Production Request and should be entered in the mm/dd/yy and hh:mm:ss formats.

7 In the case of UTC Time duration, enter ***EndDate*** and ***EndTime*** in fields labeled **End**, respectively.

- The ***Enddate*** and ***EndTime*** are the end date and time of the Production Request and should be entered in the mm/dd/yy and hh:mm:ss formats.

8 In the case of Orbit duration, enter ***StartOrbit*** and ***EndOrbit*** in fields labeled **From** and **To**, respectively.

- The ***StartOrbit*** and ***EndOrbit*** are the orbit range of the Production Request.

9 Optionally, enter ***Comment*** in field labeled **Comment:**.

- This comment will be displayed whenever this Production Request is brought up and viewed.

10 When Production Request is complete, click on **File** menu and select **Save As…**.

- A GUI labeled **File Selection** 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. A second message box stating "Write to Database of Production Request ok"; again click **Ok**.

Note that you will not be allowed to enter a PR name that already exists. PR names that already exist will be displayed in the main window. The Production Request will then be saved under the name specified.

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

11 When tasks are completed with the Production Request Editor GUI, click on the **File** menu, then choose **Exit**.

- The Production Request Editor GUI will disappear.

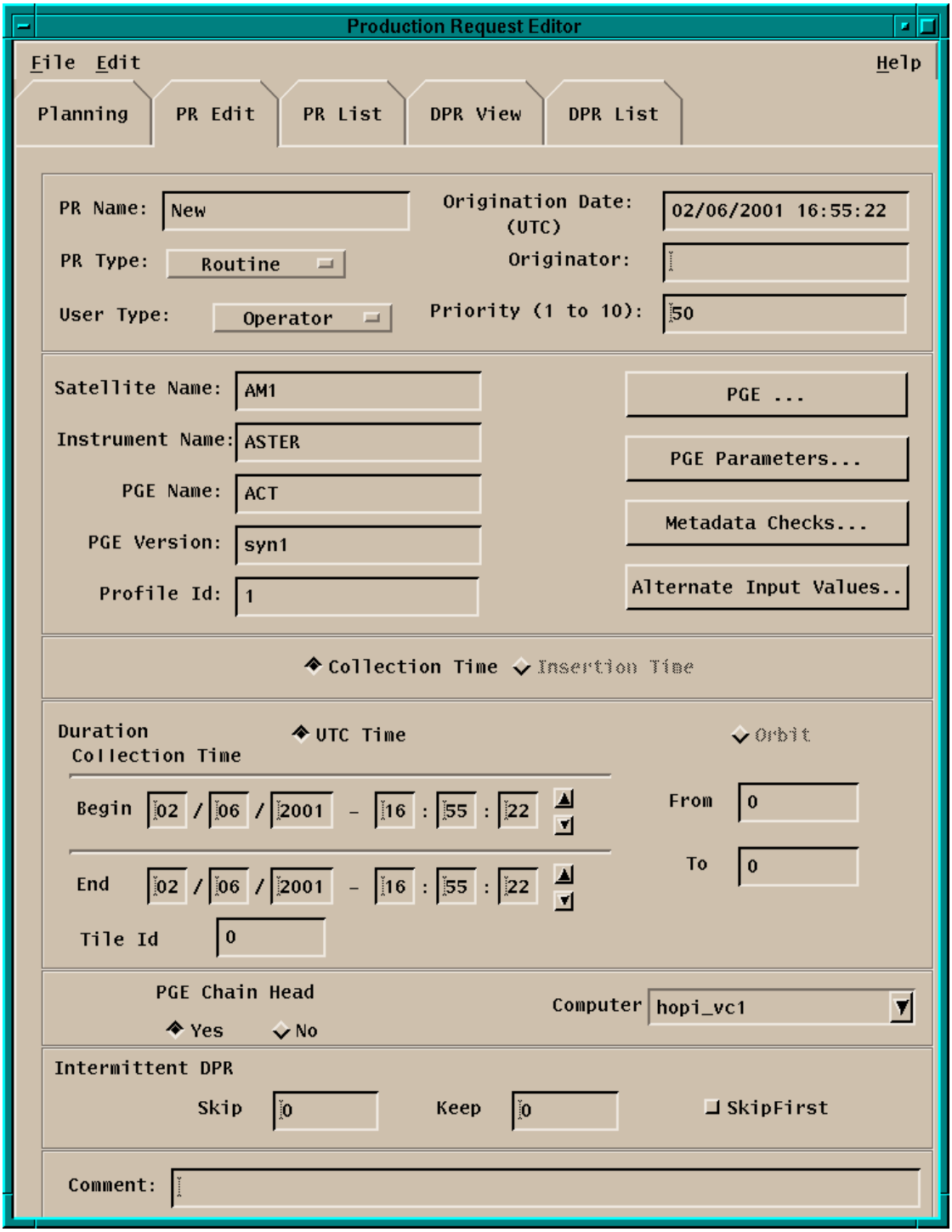


Figure 26.14.3-2. Production Request EditorGUI (Planning)

26.14.4 Processing

Once a candidate plan has been activated, each of the DPRs will result in subscriptions to the Data Server for the data needed. A request will go to the Data Server asking for notification when the required input data arrives.

Planning knows what data to request from the Data Server because the PDPS database stores this information as determined by the ESDT for each PGE. When the Data Server receives new data, it routinely checks to see if there are any outstanding subscriptions. If there are subscriptions, Planning will be notified. Once the input data required by a DPR becomes available, the DPR can be queued for processing.

Staging - The Data Processing Subsystem requests that the required input data, PGE (binary executables and shell scripts) and SDP Toolkit files be placed on a disk set aside for processing.

Process Control File (PCF) - establishes a linkage between logical Ids that the science software uses and the physical files that exist on the staging disk.

After the PGE has completed, the DPS will deallocate resources.

A Production History file will be created and will contain information concerning the conditions that the data products were generated by the PGE.

26.14.4.1 Viewing Production Requests

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

This procedure describes how to view PRs that have already been defined. 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 - 99) of the PR assigned when it was defined.
- Start - The start date and time of the PR.
- End - The end date and 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 user has selected the **PR List** tab from the Production Request Editor.

1 From the Production Request Editor GUI, click on the **PR List** tab.

- The PR List page will be displayed as shown in **Figure.** 26.14.4-1.

2 View the listed PRs. Optionally, find a PR by entering a search string in the field next to the **Find** button and then clicking on the **Find** button.

3 To modify a PR listed, click on the PR in the list and from the **File** menu select **Save As…**.

- In the **File Selection** GUI, replace the current PR name shown in the **Selection** field with a new PR name. Then click on the **OK** button.
- When modifying an existing PR, it must be saved under a new PR name.
- Next, click on the **PR Edit** tab. The PR Edit page will be displayed with fields populated from the existing PR name, but having the new PR name chosen above.
- See Section on using the PR Edit page and saving any changes made.

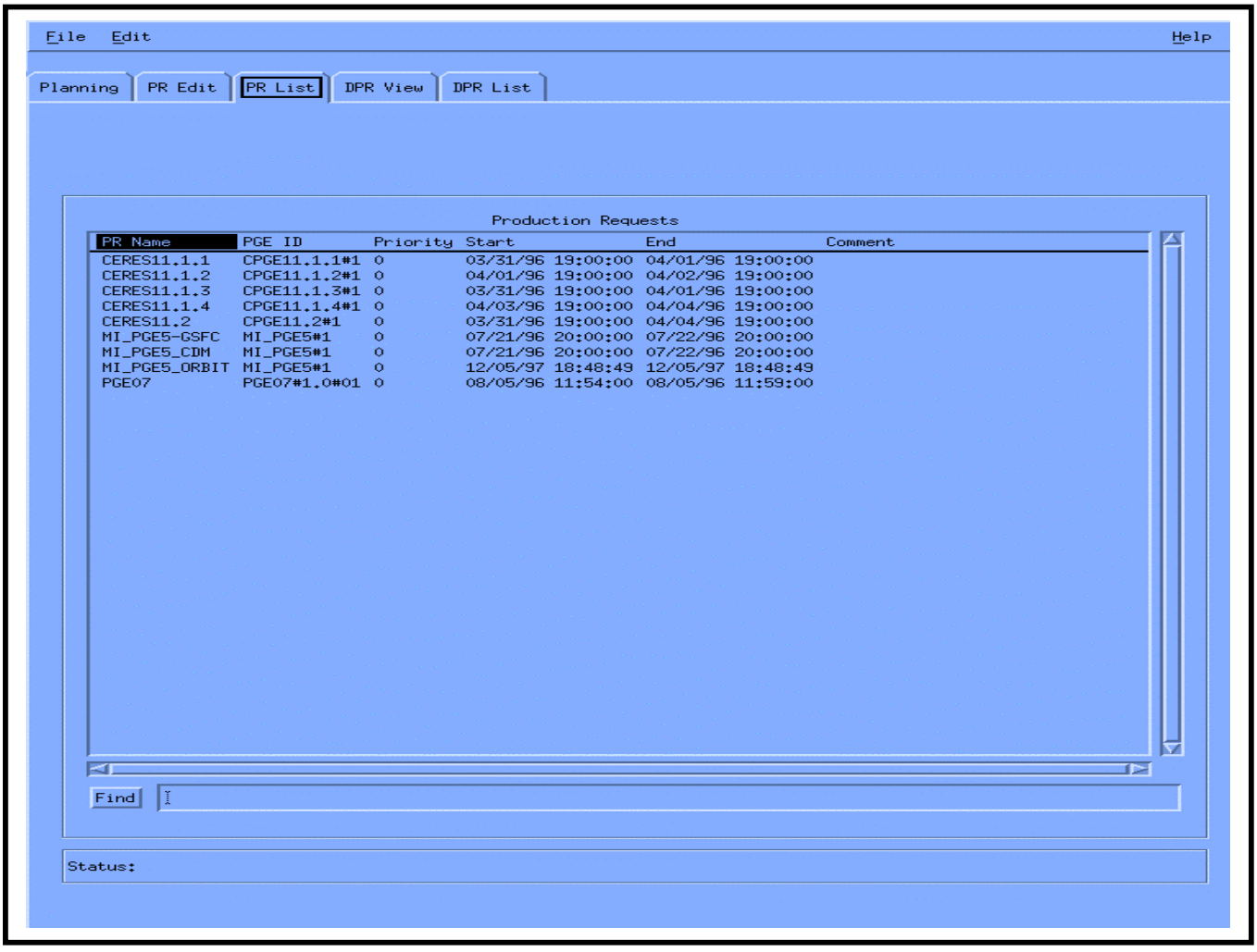


Figure 26.14.4-1. PR List GUI

26.14.4.2 Viewing Data Processing Requests

Clicking on the DPR View GUI tab displays a list of all DPRs for all PRs entered into the system.

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

1 From the PR Editor GUI, click on **DPR View** tab. The following information will be displayed:

- Data Processing Request Identification.
- PGE ID and its parameters.
- Request Data and Status.

26.14.4.3 Listing Data Processing Requests

Selection of one PR on the PR List by highlighting it and then clicking on the DPR List tab, brings up a detailed display of all DPRs associated with the selected PR. These may be examined in order to develop production plans and schedule jobs.

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

---

1 From the PR Editor GUI, click on **DPR List** tab**.** The following information is displayed:

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.

26.14.5 Using the Production Planning Workbench

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

26.14.5.1 Using the Planning Workbench to Run a PGE

Once a PGE has been fully registered, its test data files have been inserted to the Science Data Server, and a single Data Processing Request (DPR) has been generated, 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 UNIX environment variables have been set properly.
2. The required servers of the ECS System are up and running.
3. A DPR has been generated successfully.

1 **Telnet to (PDPS) p0pls01** or from the **SSIT Manager**, click on the **Tools** menu, then choose **Xterm**. Then **telnet** to a **PLN Host**.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **PLN Host.**
- It is recommended that this procedure begin within a new command shell on a PLN Host.

2 login: **ID, Password**:

3 ***setenv DISPLAY ….:0.0***

- Change the working directory to the location where the script for starting the Autosys is stored. (e.g**. cd /usr/ecs/<mode>/CUSTOM/utilities).**
- Type**: source EcCoEnvCsh**

4 At the UNIX prompt on an PLN Host, Type **EcPlAllStart <mode> <application_id>**, press **Return**.

- The ***mode*** is one of modes used in the ECS system (e.g. **TS1**.)
- The ***application_id*** is a numerical number. (e.g. **1**.)
- For example: **EcPlAllStart TS1 1, Return.**
- A Planning Workbench GUI will be appeared as shown in **Figure 26.14.6-1.**

5 In the Planning Workbench GUI, go to the subwindow labeled **Unscheduled** and click on a Production Request name.

- The Production Request name is the name under which the PR was saved.
- The PR name entry will be highlighted.

6 In the Planning Workbench GUI, click on the button next to the label **Schedule** (the button has an inverted triangle on it).

- The PR highlighted in step 3 will appear in the subwindow labeled **Scheduled**.

7 In the Planning Workbench GUI, click on the **Activate** button

- A small GUI labeled **Plan Activation** 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
- To terminate the processes of Planning Workbench GUI, type ***EcPlSlayAll mode***.

8 In the Plan Activation GUI, set the time in the time field forward to allow ample time for the PGE to run. Then click on the **Ok** button.

- All that is necessary is for there to be sufficient time for the PGE run. There is no penalty for allowing ***too*** much time.
- The Production Request thus planned will be submitted to processing and its progress can be monitored with AutoSys.

10 When tasks are completed with the Planning Workbench GUI, click on the **File** menu, then choose **Exit**.

- The Planning Workbench GUI will disappear.

Important Note **:** The creation of Production Requests and Plans requires close coordination among all who are using the same mode in SSIT. Otherwise, the submittal of a Plan may prevent a waiting DPR from starting. In particular, when submitting a new Plan in a mode being shared with others, one should:

- Ask everyone else if they have a DPR awaiting data as part of a chain.
- Check the PDPS database table PlDataProcessingRequest for any DPRs that are in state CQ_HOLD and include these in the new Production Request and Plan.
- Also, include any DPRs (except those marked SUCCESS) upon which the given DPR depends in the new Production Request and Plan.

26.14.6 Creating and Activating of a Production Plan

The Production Planner creates a plan for production data processing at the DAAC by selecting specific PRs whose DPRs are to be run. The planning tool provides a forecast of the start and completion times of the jobs based upon historical experience in running these PGEs. Through the planning tool, when the generated plan is "activated," the information included in the plan is transferred to the Data Processing subsystem and loaded into the Platinum AutoSys tool where production processing is managed.

The Production Planner creates the plan by selecting PRs from two lists of PRs, i.e., the list of available "Unscheduled" PRs and the list of "Scheduled" PRs. Using arrow buttons, the Production Planner moves the PRs between lists until the "Scheduled" list contains the desired set of PRs that define the new plan. Only one user can use the **Planning Work Bench** at a time. It is recommended for SSI&T that only one person do the planning for the group.

Before creating a new production plan the Production Planner must have available the following information:

Name of the plan.
Comments (if any).
PRs to be included in the new production plan.

1 Log into one of the **pln** sun workstations by typing: **username** then press the **Enter** key.

2 Enter the **password** then press the **Enter** key.

Prior to the remote login, enter **setenv DISPLAY** <local workstation IP address>**:0.0** where the local workstation IP address represents the IP address you where you are located**.**

You may need to setup the terminal so that the remote host is displayed on your screen (Sun machine). This is done by clicking on the **Application Manager** icon (the file drawer located at the bottom of the screen), followed by the **Desktop Tools** icon**,** followed by the **Terminal Console** icon

3 At a UNIX prompt type **cd** to the directory where the scripts are located. (e.g. **/usr/ecs/TS1/CUSTOM/utilities**).

4 At a UNIX prompt type **setenv DISPLAY *hostname*:0.0**

5 At a UNIX prompt on the **PLN host (e.g. p0pls01)**, type **EcPlPRE_IFStart mode 3** **Planning Workbench** GUI is displayed (Figure 26.14.6-1)

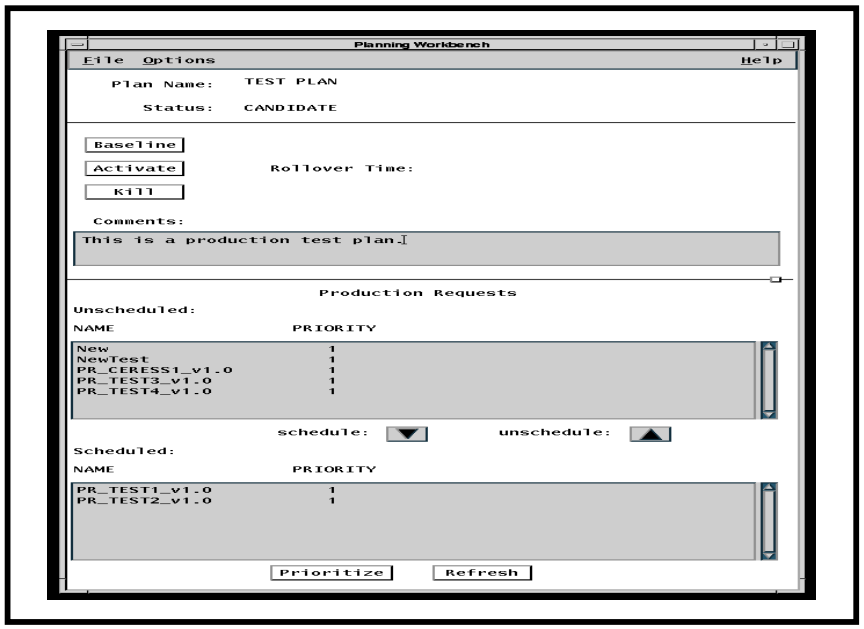


Figure 26.14.6-1. Planning Workbench GUI

Data concerning the currently active production plan are displayed.

If you want to "kill" (deactivate) the currently active production plan without activating a replacement, click on the **Kill** button.

Whenever you activate a plan (by clicking on the **Activate** button), you automatically "kill" the currently active plan.

6 Select **File → New** from the pull-down menu.
The "New" window appears.

7 Type a name for the new plan, then press the **Tab** key on the keyboard.

The **Planning Workbench** GUI is displayed.
The **Plan Name** is displayed.
The **Status** displayed is **Candidate**.

8 Type the desired date (in ***MM/DD/YY*** format), then press the **Tab** key on the keyboard to advance to the next field.

9 Type the desired time (in ***hh:mm*** format), then press the **Tab** key on the keyboard.
The **Rollover Time** is displayed.

10 Type any relevant comments (up to 255 characters) in the **Comments** field.

11 Move PRs between the **Unscheduled** and **Scheduled** lists as necessary by selecting (highlighting) the PR to be moved by clicking on the PR in the list from which it is to be moved then clicking on the up or down arrow button (as applicable) to move the PR to the other list. Highlighted PR disappears from one list and appears on the other.
The unscheduled and scheduled PR lists are scrollable.

12 When the **Scheduled** list accurately reflects the PRs to be scheduled in the production plan, select **File → Save** (or **File → Save As**) from the pull-down menu to save the new production plan.
The new production plan is saved.

13 If the new plan is to be activated immediately, click on the **Activate** button to activate the new plan.
The currently active plan is killed (deactivated) and the new plan is activated.
- The **Production Planning Timeline** GUI is displayed.

14 If the new production plan is to be used as a baseline plan, click on the **Baseline** button.
The **"New"** window appears.
The plan is recorded as well as the time of baselining so that it can be used in comparing future processing results with planned objectives.

15 If the production plan being displayed is active and should be deactivated, click on the **Kill** button. The **"New"** window appears.
The plan is deactivated without activating another plan.

26.14.6.1 Monitor Production in the (PLS)

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 seven 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

For example, for a scheduled PGE named MOPITT4, the AutoSys jobs making up that DPR would be:

MOPITT4#A

MOPITT4#S
MOPITT4#P
MOPITT4#E
MOPITT4#p
MOPITT4#I
MOPITT4#D

Table 26.14.6-1. 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 in the (PLS), execute the steps that follow:

1 In any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **DPS host**.

- It is recommended that this procedure begin within a new command shell on a DPS Host.
- Set the DISPLAY environment variable. At the UNIX prompt on the **DPS host (e.g. p0pls01)**, type **setenv DISPLAY *terminal_id***, press **Return**.

2 Set the UNIX environment variable,at the UNIX prompt on the **DPS host**, change to the directory where the scripts are located (e.g. **cd /usr/ecs/TS1/CUSTOM/utilities),** then

- Type **setenv MODE *mode*** (e.g. **TS1**).
- Type **source *environment_setup_file*** (e.g., **EcCoEnvCsh** for C shell users).

3 At the UNIX prompt on the DPS Host, under the directory where the scripts are located (e.g. **cd /usr/ecs/TS1/CUSTOM/utilities**), type **EcDpPrAutosysStart *mode Autosys_Instance* &**, then press **Return**. Note: where Autosys_Instance is the instance of autosys.

- The ***mode*** is one of modes used in the ECS system (e.g., **TS1**).
- The ***Autosys_Instance*** is a numerical number (e.g., **1or 2 if a DPR is already in progress in 1**).
- For example: **EcDpPrAutosysStart TS1 1, Return.**
- A GUI labeled **AutoSys** will be displayed.
- This GUI will contain eight buttons for invoking various tools available under AutoSys.

4 In the AutoSys GUI, click on the **Ops Console** button.

- A GUI labeled **AutoSys Job Activity Console** GUI will be displayed.
- The main subwindow of this GUI will contain a dynamically updated list of AutoSys jobs (seven jobs make up one DPR) currently scheduled.
- To disable dynamic updating of the main subwindow (which may be distracting), click on the **View** menu, then choose **Select Jobs**. A GUI labeled **Job Selection** will be displayed. 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 middle 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**. To kill the DPR, click on the **Delete** tab. ***(Warning: Be very careful to avoid deleting wrong DPR!)***
- Exit the **AutoSys Job Activity Console** by clicking on the **Exit** button.

5 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 (seven 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 (Staging) 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.

- To exit the **JobScape**, click on the **File** menu, then choose **Exit**.

6 In the AutoSys GUI, click on the **TimeScape** button.

- A GUI labeled **TimeScape** GUI will be displayed.
- The main subwindow labeled **Job Name** of this GUI will contain a dynamically updated list of AutoSys jobs (seven per DPR) currently scheduled.
- To disable dynamic updating of AutoSys jobs (which may be distracting), click on the **Freeze Frame** button.
- The color of each job indicates its status according to the legend on the left side of the GUI.
- The time line is shown on the right side of the GUI with time marked at the top. A red vertical line (dashed) indicates the current time.
- Exit the **TimeScape** GUI by clicking on the **File** menu and selecting **Exit**.

7 To quit AutoSys, in the AutoSys GUI, click on the **Exit** button.

- The AutoSys GUI will disappear.
- Refer to Section (Troubleshooting and General Investigation) if any job fails on AutoSys.

On workstation **x0pls##**, at the UNIX prompt in a terminal window, enter steps as in step 1 below.

NOTE: The **x** in the workstation name will be a letter designating your site: **g** = GSFC, **m** = SMC, **l** = LaRC, **e** = EDC, **n** = NSIDC, **o** = ORNL, **a** = ASF, **j** = JPL, **p**= PVC; the ## will be an identifying two-digit number (e.g.,**g0pls01** indicates a Planning and Data Processing subsystem(PLS) workstation at GSFC).

Prior to the rlogin, enter **setenv DISPLAY <local_workstation IP address>:0.0.** The **<ipaddress>** is the ip address of **x0pls##,** and **xterm** is required when entering this command on a Sun terminal.

26.14.7 Monitoring Production in PDPS Subsystem

Example of monitoring production by using these procedures:

1 **telnet to (PDPS) p0sps06** or from the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then **telnet** to a PLN Host, login: **ID**, **password:**

2 setenv DISPLAY ….:0.0

- Change the working directory to the location where the script for starting the Autosys is stored. (e.g**. cd /usr/ecs/<mode>/CUSTOM/utilities).**
- **setenv MODE <MODE>**
- **source EcCoEnvCsh**

3 At the UNIX prompt on the PLN Host, type: **EcDpPrAutosysStart *mode application_id* &**, then press **Return**.

- The ***mode*** is one of modes used in the ECS system (e.g., TS1).
- The ***application_id*** is a numerical number (e.g., 1).
- For example: **EcDpPrAutosysStart TS1 1, Return.**
- A GUI labeled **AutoSys** will be displayed.

- This GUI will contain eight buttons for invoking various tools available under AutoSys.

4 In the AutoSys GUI, click on the **Ops Console** button.

– The AutoSys GUI will be displayed.

5 Select the desired display and view the contents.

– Selections include:

1. Autosys Job Activity Ops Console
2. HostScape
3. TimeScape
4. JobScape

- The main subwindow of this GUI will contain a dynamically updated list of AutoSys jobs (seven jobs make up one DPR) currently scheduled.
- To disable dynamic updating of the main subwindow (which may be distracting), click on **Freeze Frame** in the small subwindow labeled **Show**.
- DPRs will be listed in the column labeled **Job Name** and their statuses (SUCCESS, FAILURE, TERMINATED) 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**.
- Exit the **AutoSys Job Activity Console** by clicking on the **Exit** 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 (Staging) 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

6 In the AutoSys GUI, click on the **TimeScape** button.

- A GUI labeled **TimeScape** GUI will be displayed.
- The main subwindow labeled **Job Name** of this GUI will contain a dynamically updated list of AutoSys jobs (seven per DPR) currently scheduled.
- To disable dynamic updating of AutoSys jobs (which may be distracting), click on the **Freeze Frame** button.
- The color of each job indicates its status according to the legend on the left side of the GUI.
- The time line is shown on the right side of the GUI with time marked at the top. A red vertical line (dashed) indicates the current time.
- Exit the **TimeScape** GUI by clicking on the **File** menu and selecting **Exit**.

7 To quit AutoSys, in the AutoSys GUI, click on the **Exit** button.

The AutoSys GUI will disappear.

8 Use the "**tail -f em.log**" to create a permanent record of a log file if debugging is necessary. Note: the xx.log file has to be created first: "**vi em.log**"

9 To look at the Data Base type: **setenv MODE <MODE>, cd utilities**

source EcCoEnvCsh, cd dbr, dbrowser-syb <MODE> 2 &

26.14.8 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. The Q/A Monitor is discussed in more detail in Section 15.

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 **telnet to (PDPS) p0sps06** or from the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then **telnet** to a **PLN Host**.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then **telnet** to the **PLN Host**.
- login: **ID, password:**

2 Set the DISPLAY environment variable: **setenv DISPLAY terminal_id, press Return.**

- Type **setenv <mode> (**e.g..,**TS1).**
- Type **source** environment_setup_file **(**e.g**., EcCoEnvCsh** for C shell users**).**
- Change the working directory to the location where the script for starting the Autosys is stored. (e.g**. cd /usr/ecs/<mode>/CUSTOM/utilities).**

3 At the UNIX prompt on the PLN Host, type: **EcDpPrQaMonitorGUIStart <MODE> <Q/A Monitor Instance>, example: TS1 1**, press **Return**.

- The ***mode*** is the operations mode.
- The **Q/A Monitor Instance** e is a numerical number. (e.g. **1**).
- The Q/A Monitor GUI will be displayed.
- Various messages from the Q/A Monitor will appear in this window as it is running.

4 In the Q/A Monitor subwindow labeled **Data Types,** select an ESDT from the list presented and click on it.

- Use the scroll bars if necessary to locate desired ESDT.
- Optionally, use the **Find** field and button to locate an ESDT.

5 In the Q/A Monitor, under the label **Data Granule Insert Date (mm/dd/yy)**, set the date range within which the search for granules of the ESDT selected will be conducted.

- The date range can be made arbitrarily large to select all granules of a particular collection (ESDT).
- The dates refer to date of granule Insert to the Science Data Server.

6 In the Q/A Monitor, click on the **Query** button.

- The results of the query will be displayed in the bottom window, labeled **Data Granules**.
- All granules having the ESDT selected in step 3 and having Insert times within the date range specified in step 4 will be listed in this window.

7 In the Q/A Monitor subwindow labeled **Data Granules**, click on one of the data granules listed to be examined.

- The data granule selected will be highlighted.

8 To retrieve the data granule's Production History file, click on the **Retrieve Prod History** 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 PLN 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 Prod History** button.
- If only the PH is desired, exit this procedure. To retrieve the data granule itself, continue on to step 8.

9 To retrieve the data granule, click on the **Retrieve Data Granule** button and note its file name (listed in the entry for the granule; you may have to scroll over to the right to see it).

- 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.

10 To examine the data granule, click on the **Visualize data** tab.

- The **Visualize data** page will be displayed.

11 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.

12 Click on the **Visualize** button.

- This action will invoke **EOSView** with the granule selected.
- The granule must be HDF or HDF-EOS format.

13 When tasks are completed with the Q/A Monitor GUI, click on the **File** menu, then choose **Exit**.

26.15 Postprocessing and General Investigation

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

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

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

26.15.1 Examining PGE Log Files

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

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

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

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

25.15.2 ECS Mechanisms for Capturing Information about a PGE's Execution

Note: This information was provided by ECS expressly for this document. It describes two tar files which are generated by the ECS Planning and Data Processing System (PDPS). One of these file types is generated for each execution of a PGE that is run within the ECS environment.

The particular file type that is generated depends on the success or failure of the PGE execution. The ECSDataGranule metadata contains a reference to the Production History.

25.15.2.1 ECS Production History (PH)

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

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

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

Other component files are:

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

The PCF file: Named PGEname#versionMMDDhhmm.Pcf – contains the actual instantiated PCF used when running the instance of the PGE (this dpr). Note that the PCF contains URs for all inputs to this execution of the PGE. There is one UR for each input granule. If a granule is a multi-file granule, the same UR will appear (repeat itself) for each inputs file of the granule.

The Production Log file: Named PGEname#versionMMDDhhmm.ProdLog – Contains the DPR ID, the PGEID, and resource usage information (same as in the .Log file). Resource usage information includes:

CPU time in application
CPU time in system
Physical memory
Max. resident set size
Avg. shared text size
Avg. shared data size
Avg. shared stack size
Page reclaims
Page faults
Swaps

Block input ops
Block output ops
Messages sent
Messages received
Signals received
Voluntary context switches
Involuntary context switches

The Profile file: Named PGEname#versionMMDDhhmm.Profile – Contains the environment variables defined during the execution of the PGE including the contents of the PATH environment variable.

The SDP Toolkit log files:

1) The Report log file: Named PGEname#versionMMDDhhmm.TkReport – This is the same log file that the SDP Toolkit generates when the PGE runs outside of ECS. (See SDP Toolkit documentation.)

2) The Status log file: Named PGEname#versionMMDDhhmm.TkStatus – This is the same status log file that the SDP Toolkit generates when the PGE runs outside of ECS (See SDP Toolkit documentation.)

3) The User log file: Named PGEname#versionMMDDhhmm.TkUser – This is the same user log file that the SDP Toolkit generates when the PGE runs outside of ECS (See SDP Toolkit documenation.)

25.15.2.2 ECS Failed PGE Tar File

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

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

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

The error log entries referencing a PH tar file are of the format:

:PH.version:dbid:1.BINARY where version is the ESDT version no. for the Production History ESDT, and dbid is the database ID, a unique identifier within SDSRV.

The error log entries referencing a FailedPGE tar file are of the format:

:LM: FAILEDPGE.version:dbid:1.BINARY where version is the ESDT version no. for the FailedPGE ESDT, and dbid is the database ID, a unique identifier within SDSRV

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

26.16 Examining PDPS-Related Scripts and Message Files

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

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

26.16.1 Examining AutoSys JIL Scripts

JILxxxxxxxxx is the Job Information Language (JIL) script that defines the DPR job to **AutoSys** and which must be submitted to the **AutoSys** Database before a DPR job can be run. The name of the file created is system-generated and begins with the characters 'JIL' followed by nine characters (e.g. JILAAAa0066c).

Sample file content:

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

To examine JILxxxxxxxx scripts on the AIT Sun, execute the procedure steps that follow:

1 At the UNIX prompt on an AIT Sun, type **cd *JILscriptPathname***, press **Return**.

- The ***JILscriptPathname*** is the full path name to the location of the JILxxxxxxxx scripts to be examined.

2 At the UNIX prompt on the AIT Sun, type **vi *JILscriptFilename***, press **Return**.

- The ***JILscriptFilename*** is the file name of the JILxxxxxxxx script to be examined.
- This brings up the file named ***JILscriptFilename*** in the *vi* editor.
- Any text editor may be used such as *emacs*. For example, **emacs *JILscriptFilename***, press **Return**.

26.16.2 Examining Application Log Files (ALOG)

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

Log files are generally named according to the convention:

***ApplicationName*.log**

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

Where an **SSIT Manager** application has been run using login **cmts1**, pw: **ecsu$er**, the log files will be found using path: **/usr/ecs/{MODE}/CUSTOM/logs/.**

Connectivity failures have been encountered when installing **ESDT's , MCF's** and **.met files**. The term bounce the servers has been widely used in conjunction with the effort to re-install or delete files. **Bounce** means to **shut down a server** and then **bring them back up** to rid the servers of unwanted or old bindings. The nature of what needs to be done is outlined as follows:

1 Install or Delete **ESDT's** - the **SDSRV** and **ADSRV** need to be bounced after installation or removal of **ESDT**'s to allow for a refresh of the **Connectivity** cell management.

2 For **PGE…..odl, MCF's** and **.met files,** bouncing the servers **SDSRV** and A**DSRV** need to done after installation and reinstallation.

3 This can be done by loging into **ECS Assistant** for each server. The login should be with generic **ID**: and **PW**:**,** and then press **Enter** Key**..**

26.17 PDPS Troubleshooting - The PGE Job has Failed

- The PGE Job has failed, but the DPR has not gone into "Failed-PGE" processing
- The Post-Execute Job has failed
- The PGE Job and Post-Execute Job have both failed, but the DPR has not gone into "Failed-PGE" processing
- The PGE Job has failed and the DPR has gone into "Failed-PGE" processing

26.17.1 The PGE Job has failed

This condition is indicated when the PGE job only is red in **AutoSys**. This is hard to do, because the **AutoSys** job definition for this job says to allow any exit code to indicate success. This is because we want the next job, the post-execute job, to continue
even if this job fails. This job will "succeed" even if the PGE Wrapper job, **EcDpPrRunPGE,** doesn't exist. This job can fail if **AutoSys** cannot see the machine machine.

26.17.2 The Post-Execute Job has failed

This condition is indicated when the Post-Execute Job only is red in **AutoSys**. This happens when the PGE job never ran or if for some other reason (such as a mount point problem) the Execution Manager job cannot read the log file created by **EcDpPrRunPGE.**
Check that **/usr/ecs/B302TS1/CUSTOM/bin/DPS/EcDpPrRunPGE** and **EcDpPrRusage** exist on the science processor and that they are not links.
Check that **/usr/ecs/DEV04/CUSTOM/data/DPS** on the science processor is mounted or linked to **/usr/ecs/DEV04/CUSTOM/data/DPS** on the queuing server machine.

26.17.3 The PGE Job and Post-Execute Job have both failed

This condition is indicated when both the PGE and Post-Execute Jobs are red in **AutoSys**, but no other jobs are red. This indicates that the Post-Execute job has read the log file created by **EcDpPrRunPGE** in the runtime directory and has found an exit status not equal to 0. However, it failed to de-stage the failed **pge tar file**.

26.17.4 The PGE Job has failed, the DPR has gone into "Failed-PGE" processing

This condition is indicated when the entire job box has turned red along with post-execute, de-staging and de-allocation jobs. A Failed PGE Tar File has been created and archived.
A PGE may fail for many reasons. Some of the possible causes are documented here:

- The PGE is the wrong architecture.

This happens when the PGE was miss-defined as **New32, Old32** or **64** from the SSIT Operational Metadata GUI. The PGE will core dump because of this problem. To fix this you need to go back to **the SSIT Operational Metadata GUI** and enter the correct architecture, then delete any **DPRs** created for that **PGE** and recreate them.

- One of the expected inputs for the PGE is missing

The first reason for this is that an expected input of the PGE is NOT defined in the **PGE ODL**. Check the error messages for something about a missing Logical Id and then check the **PGE ODL** for the expected **Logical Id.**
This can also happen when a miscommunication causes Subscription Manager to release a PGE despite it missing one (or more inputs). To find out if this is the case, verify that all inputs to the DPR have their availability flag set to 1 in PlDataGranuleShort and the corresponding entries in **PlDprData** have the accepted field set to 1.

- The leapseconds file is incorrect.

26.17.5 PDPS Troubleshooting - A single DPS job has failed or is hanging

The entire Job Box is hung
A DPS Allocation job is hanging

A DPS Allocation job has failed

A DPS Staging job is hanging
A DPS Staging job has failed
A DPS Preprocessing job is hanging
A DPS Preprocessing job has failed
A DPS PGE job is hanging
A DPS PGE job has failed
A DPS Post-processing job has failed
A DPS De-staging job is hanging
A DPS De-staging job has failed
A DPS De-allocation job has failed
All known problems are corrected and the re-start of the job fails again

Non-PDPS servers are down

Back to the PDPS Troubleshooting home page

26.17.5.1 The entire Job Box is Hung

This condition is determined by noting that the entire Job Box (all 8 job steps) are the same color, and it is either the one indicated for "Inactive" jobs or the one for "On Hold" jobs. Check the legend to the left on the Jobscape display for the job box color meanings.

- The AutoSys Event server or one of the AutoSys clients could be down. See
- Attempt to re-start the Job Box by selecting the top of the Job Box.

26.17.5.2 A DPS Allocation job is hanging

This condition is determined by noting that the Allocation job has turned green to indicate that it is running, but that it never turns red (failed) or blue (success). If you "tail" the DPR .err file (eg: "tail -f
/usr/ecs/OPS/CUSTOM/logs/MODPGE08#s28035000OPS.err") you see that nothing is happening, or that the job is in a retry loop.

- The Science Data Server (SDSRV) may be waiting for a request to Data Distribution (DDIST) to distribute the PGE tar file, but it can't because Storage Management (STMGT) is

down. Go to where the DDIST GUI is running. Refresh the GUI. Check to see if the requestor source is EcDpPrEM and that the state is "Suspended with Errors". If this is the case, then you will have to bounce STMGT. After this is done, select the request on the DDIST GUI and click on "resume".

- The Science Data Server (SDSRV) may be waiting for a request to Data Distribution (DDIST) to distribute the PGE tar file, but it can't because Storage Management can't FTP the file to the data directory on the science processor disk. Go to where the DDIST GUI is running. Refresh the GUI. Check to see if the requestor source is EcDpPrEM and that the state is "Suspended with Errors". Does the target directory exist? Can you FTP a file to the directory on the science processor? If the answer to these questions is no, then fix the problem, and resume the request.

- If you observe that the Allocation Job is in a retry loop, then the SDSRV may be down. See Non-PDPS servers are down. Note that the first retry is designed to fail, because the software is retrieving server-side information to refresh the client-side at this point.

- The request may be waiting on the archive to stage the file. If there are several other requests in progress, the PGE acquire may have to wait until one or more of them completes. Check the state in the DDIST GUI - if it is in "staging" state, then the request should eventually complete.

26.17.5.3 A DPS Allocation job has failed

This condition is determined by noting that the Allocation job has turned red.

- Look at the .ALOG file (in **/usr/ecs/{MODE}/CUSTOM/logs**). If it is there, then look for the following:

  A message of "Error: unable to update Machine in Autosys" means that DPS is unable to access the AutoSys database. The auto.profile **in /usr/ecs/MODE/CUSTOM/bin/DPS** has the wrong settings for AUTOSYS and AUTOUSER parameters. Although they may differ from DAAC to DAAC, the expected values are:

  AUTOSYS = /usr/ecs/MODE/COTS/autotreeb/autosys
  AUTOUSER = /usr/ecs/MODE/COTS/autotreeb/autouser

- To fix the problem, you either need to run the AutoSys Mkcfg again or go into the auto.profile file and change the values by hand.

- A message of "Unable to determine type of UR" means that the PGE tar file has not been inserted. To verify this is the problem check the PlResourceRequirement table in the PDPS database. There should be a non-null entry for the field exeTarUR. If that field is null, you need to go back to the SSIT procedure and insert the EXE Tar File. Then you should be able to re-start the job and watch it complete successfully.

If the .ALOG file is NOT present. Then do the following:

- Bring up the Autosys Ops Console and select the Allocation Job that has failed.

- Check the return code. A value of 122 means that owner of the job DOES NOT HAVE WRITE PERMISSION to the log files directory. You need to find out the user account that was used to bring up Autosys and verify that it is correct and should have write permission to the logs directory.

26.17.5.4 A DPS Staging job is hanging

See A DPS Allocation job is hanging

26.17.5.5 A DPS Staging job has failed

This condition is determined by noting that the Staging job has turned red. Look at the .ALOG file (**in /usr/ecs/{MODE}/CUSTOM/logs**) for the DPRID of the job that has failed.

- A message of "ESDT Acquire Failed for UR...." means that SDSRV had trouble processing one of the acquire requests. In this case re-starting the job should allow the acquire to succeed.

26.17.5.6 A DPS PreProcess job has failed

This condition is determined by noting that the Pre-Process job has turned red.

Look at the .ALOG file (in **/usr/ecs/{MODE}/CUSTOM/logs**) for the DPRID of the job that has failed. If it is there, then look for the following:

- A message of "NOFREECPUS" means that all of the Science Processor CPUs are busy and the PreProcess job went through its maximum number of retries to find an available CPU. You can just start the job again and it will work its way through its retries until a CPU is available.

Possible reasons for a job to run out of CPU resources:

- PGEs are taking longer to run than expected. DPS plans for execution times specified during SSIT, and if those times are exceeded by a large margin (by an executing PGE) it is possible that a PGE that is "ready to run" will be CPU starved.

26.17.5.7 A DPS PGE job is hung

This condition is determined by noting that the Execution job has turned orange or ossilates between orange and green.

- The AutoSys client is most likely down. See Checking the Status of AutoSys for how to verify AutoSys is up and happy.

26.17.5.8 A DPS PGE job has failed

This condition is determined by noting that the Execution job has turned red or the entire job box has turned red (failedPGE scenario).

See Troubleshooting - The PGE Job has Failed.

26.17.5.9 A DPS De-staging job has failed

The destaging job icon on the JobScape GUI will have turned red. Look at the .err log file.

- Typically, you will see a message such as "Error archiving metadata into catalog". You may also see some warning messages in the returned GlParameter list. You can disregard the warnings. If the problem occurred for an existing ESDT which has previously worked within the past day or two, then most likely STMGT is the culprit. Have someone from STMGT look at their log files, paying particular attention to changes/defects in their stored procedures.

- If you see the "Error archiving metadata into catalog" message and the ESDT is new or has recently been installed, then look at the .MCF file in the runtime directory. Get somebody from SDSRV to help you compare the values of the mandatory parameters in the metadata file with "valids" from the SDSRV database.

- "Error archiving metadata into catalog" may also be associated with a SDSRV temporary directory getting filled up.

  A message that indicates "Error archiving files" means that SDSRV is having trouble getting Storage Management to place the file(s) in the archive. Contact a Storage Management person:

- to Verify that the Archive (AMASS) is up and functional.

- Check through their logs to see why the request for archiving failed.

  It is possible that the mount point between the science processor and the Storage Management machine has been lost. Check to make sure that the file that is being de-staged can be seen on the Storage Management machine. The typical path for the mount point is:
  Name**}**
  /usr/ecs/{MODE}/CUSTOM/pdps/{science processor name}/data/DpPrRm/{Disk

- A message that indicates "Error modifying file usage" means that the numberOfUsage column in DpPrFile for a particular file is at 0 and the software is trying to decrement it. This column is an increment/decrement counter and is not normally decremented more times than it is incremented when under software control. However, if someone manually changes the database then the value may get out of sync and need to be manually reset to 1.

- If you see science data files in the disk partition, but no metadata files, then DDIST/STMGT is okay and SDSRV is not okay. Otherwise, suspect STMGT.
- When the problem is corrected, re-start the job from AutoSys.

26.17.5.10 All known problems are corrected and the re-start fails again

The retry information in DpPrRpcID may now be out of sync between one or more servers. Find the appropriate entry in the table by inspecting the readableTag column and remove the entry before trying to re-start the job again.

26.17.5.11 Non-PDPS servers are down

Always verify that the Science Data Server, Storage Management Servers and Data Distribution Servers are up.

1. Bring up ECS ASSIST
2. Select the correct mode
3. Click on the "monitor" button
4. Observe that the status for the following servers is "Listening":
   EcDmDictServer
   EcDpPrDeletion
   EcDpPrJobMgmt
   EcDsDistributionServer
   EcDsScienceDataServer (all instances)
   EcDsStArchiveServer
   EcDsStFtpDisServer
   EcDsStPullMonitorServer
   EcDsStStagingDiskServer
   EcDsStStagingMontitorServer
   EcIoAdServer
   EcPlSubMgr
   EcSbEventServer
   EcSbSubServer
6. If any of these servers are down, contact the MSS operator to bring them up.

26.17.6 PDPS Troubleshooting - Job Activation Fails from the

26.17.6.1 Planning Workbench

Error reported is "DPR Validation Failed"

1. Check to make sure that Performance data has been entered for the PGE
   1. Use the database browser or isql to get access to the PDPS database.

2. Look at the entries for PlPerformance.
3. For the PGE(s) that are schedule, verify there is a non-zero value for the entries in this table.
4. If entries are 0, then run the SSIT Operational Metadata GUI to enter correct performance values.
5. Delete the DPRs and then re-create them. Activation will succeed on the next attempt.

26.17.6.2 PDPS Troubleshooting - Input Data Problems

General description of the problem:

- The Production Request fails due to too many granules
- The Staging Job has failed due to too many granules

A Failure Due to Too Many Granules

A failure of this sort is caused by too many granules meeting the criteria for input granules for a particular DPR. At PGE registration, the number of granules we expect for each input ESDT is defined. We define the minimum number and the maximum number of granules we expect. If the number of granules found is not between the minimum and maximum number, the request fails. This will fail either during production request time, or when the PCF File is generated (Autosys PreProcessing step).

The Production Request fails due to too many granules

The Production Request fails. The ALOG displays an error message as follows:
Msg: PlPge::GetInputForDpr - Extr input to process DPR MoPGE01#2007081600OPS, for data type id MOD000#001, with logical id 599001. PlDataTypeReq has a scienceGroup of for this datatype. Expected 2 max inputs, but got 3. Priority : 2
Time : 07/09/99 17:10:52

The problem in this case was that the Production Request Editor queried the PDPS database for granules that would satisfy the data needs for this DPR and found 3 granules instead of 2 like it expected. When we inspected the PDPS database (PlDataGranuleShort) we found only two entries that satisfied the data needs for this DPR. After some investigation, the true problem was discovered.

The DPR was for 16:00:00 - 18:00:00. This particular DPR takes in the current input granule (16:00:00-18:00:00) and the previous input granule (14:00:00- 16:00:00). When it inspects the PDPS Database for data granules, it "pads" the timeframe with a (5A) configurable percentage or (4PY) a hardcoded percentage - 50%. This was 4PY, so the PRE added 50% of the granule's expected time to each side of the time-range of the granules. This means that the PRE queried the PDPS database for granules that had start and stop times within 15:00:00 - 19:00:00. In this case,

there was an invalid data granule with a start time of 15:30:00 - 15:59:59. This granule was found during the query and caused the PRE to find one too many granules and fail the PR.

The Workaround:
Delete the offending granule and recreate your production request.

The Fix:

For 5A, the padding that is added to each side of the range for the query is configurable. If the padding is decreased, these kinds of granules will not be found. In addition, there is a minimum size that a granule must be before the Planning system will recognize it as a valid granule. This is a percentage, and is hardcoded in 4PY to 20% (or so), but this is configurable in 5A. So in 5A we can set this value to 50% which will filter out all granules less than 1 hour long, and we can set the padding to 50% which will filter out granules that do not start before 15:00:00. Since our granules must be an hour long, and we can assume they do not overlap, we would not get any invalid granules such as this one.

The PreProcessing Job has Failed due to Too Many Granules
This condition is indicated when the Staging Job is red in AutoSys. This happens due to the same condition as noted above with the PRE, but the data granules came in after the Production Reqeust was generated. The error message appears in the PGE log files and comlains about not being able to generate the PCF file due to too many granules for a particular logical id.

The Workaround:
Delete the offending granule and the production request and recreate the production request.

The Fix:
See fix for the previous scenario.

26.17.6.3 PDPS Troubleshooting - Jobs are activated, but do not

26.17.6.3.1 Get started in AutoSys

The Job Management Server is down
The DPR is waiting in the AutoSys queue (never got released)
Subscription Server Problems
The DPR was released but failed due to a JIL failure
The DPR was released but failed due to a AutoSys ID failure
The DPR was released but failed to be received by Job Management Server
AutoSys is not functional
AutoSys is full

26.17.6.4 The Job Management Server is down

1.Bring up ECS ASSIST
2.Select the correct mode

3.Select DPS for the subsystem
4.Click on the "monitor" button
5.Observe that the status for the EcDpPrJobMgmt server is "UP"
6.If it is "down", then do steps 7 - 8
7.Click on the "start" button
8.Repeat steps 1-5

26.17.6.5 The DPR is waiting in the AutoSys queue (never got released)

The Job Management server may have never received a ReleaseDprJob command from the PLS Subscription Manager

1.Check the database table, DpPrCreationQueue, to see if the job is still waiting to into AutoSys. If it's in this table then it probably never got a ReleaseDprJob command get from the PLS Subscription Manager, unless AutoSys is full)
2.Check the Job Management error log file to see if the ReleaseDprJob command was sent
3.If it was, then there may have been a JIL (AutoSys Job Information Language) processor problem -- a JIL FAILURE
4.If you can't find any evidence that the command was sent to Job Management, then the PLS Subscription Manager didn't send the ReleaseDprJob command. It won't send this command if it doesn't think all of the DPR's required inputs have been received. Verify this for yourself as follows:

a. For regular DPRs (ie. one without optional inputs), check to see if all of the required inputs are present. First look at the PlDprData table and find all of the granule Ids with an ioFlag of 0 (an input granule) for this DPR. Then look at the UR column for each granule Id in PlDataGranule. If all of the input granules have URs (as opposed to granule Ids), then the Subscription Manager should have sent a ReleaseDprJob command to Job Management.

Look at the Subscription Manager log file to verify that it never, in fact, sent the ReleaseDprJob command.

b.While you're looking at the Subscription Manager log file, check to see if it got a subscription notification from the

Subscription Server for any dynamic data that the DPR needs. If you believe that all of the necessary input files for
the DPR have been inserted by another DPR, then there may be Subscription Server Problems
c.If there are no Subscription Server Problems, all of the input granules for the DPR have URs and/or Subscription Manager received notification for all dynamic granules, then something may be wrong with the Subscription Manager. Check with somebody in the PLS subsystem.

26.17.6.6 The DPR was released but failed due to a JIL failure

A "JIL Failure" means that the Job Management Server had some problem placing the DPR in AutoSys. The Job Interface Language processor rejected the create job command sent to it by the

Job Management Server. If you look at the completionState column for the DPR in the PDPS PlDataProcessingRequest table, you will see "JIL_FAILUR". There are 2 main reasons for this:

1.There is already a job with an identical name in AutoSys. Check this by going to the Ops Console, select View->Select Jobs and type a portion of the job name in the "Job Name" box, bracketed by the "*" or "%" wildcard character. If the job is already in AutoSys, it must be removed by using the Production Request Editor or by using the Job Management Server Client tool (selectable from the Ops Console). Never delete a job from AutoSys using the job definition GUI. This will corrupt the PDPS database.
2. The event processor is down - AutoSys is not functional.
3. The job had a problem when it was loaded into AutoSys and a malformed or mutant job box is the result. This is a job box which will stay dark blue (meaning that it was not activated) and will be missing one of the seven job steps. To correct this problem you must do the following:

- Delete the job from AutoSys by hand. To do this select the job from JobScape and right click. Select the Job Definition and then select Delete from the pop-up window. In general, it is bad practice to delete a job from Autosys using the Job Definition GUI. This can cause corruption in the PDPS database. But for this problem there is no other solution.
- Update the completionStatus of the DPR in the PDPS database for which the mutant job box was created. Youmust do this via isql and set the completionStatus = NULL (using the isql update command).
- Delete the DPR that maps to the job via the Production Request Editor. Note that you do not want to delete the entire Production Request, only the DPR that had the mutant Job Box. Any DPRs that depend on this DPR will also have to be deleted.
- Re-create this DPR and any subsequent DPRs via the Production Request Editor.

26.17.6.7 The DPR was released but failed due to a AutoSys ID failure

An "AutoSys ID" failure is indicated if the following messages appear in the Job Management ALOG file:

```
PID : 7668:MsgLink :0 meaningfulname :DpPrAutosysMapList::GetAutosysIDByDpr
 Msg: unable to find autosys id for dpr: ACT#syn1#004130123DEV02 Priority: 2 Tim
e : 03/09/99 11:33:51
PID : 7668:MsgLink :9 meaningfulname :CantFindAutoSysId
 Msg: Unable to find autosys id Priority: 2 Time : 03/09/99 11:33:51
PID : 7668:MsgLink :10 meaningfulname :DpPrSchedulerDObjSmainCreateFailed
 Msg: RqFailed=CreateDpr DprID=ACT#syn1#004130123DEV02 Priority: 2 Time : 03/09/99
11:33:51
```

An "AutoSys ID" Failure means that the Job Management server could not associate the AutoSys ID with the DPR that was activated. When the Job Management server is started it reads various tables in the PDPS database that provide the linkage between processing resources and AutoSys instance. If data is missing from these tables, or was added after the Job Management server was

started, then the error shown above can occur when any jobs are activated by the Planning Workbench.

The following actions should be taken when an "AutoSys ID" failure error is reported:

1. Verify that the **PlResource** table in the PDPS database has at least 1 entry for a processing string and at least one entry for an AutoSys Instance. If either of these are missing, then you need to re-do Resource Planning and add them via the Resource Editor GUI.
2. Verify that the **PlRscString** table in the PDPS database has at least 1 entry and that autosysIdKey matches the entry in the PlResource table. Again, if information is missing or wrong, you need to re-do Resource Planning.
3. Verify that the **DpPrAutosysMapList** table in the PDPS database has at least 1 entry and that **resourceString and autosysIdKey** matches the entry in the **PlRscString** table. Yet again, if information is missing or wrong, you need to re-do Resource Planning.
4. If Resource Planning has been done after the Job Management server was brought up, then bounce the server. Since the server reads this information at start up, any changes since it was brought up will NOT have taken affect.

26.17.6.8 The DPR was released but failed to be received by Job Management Server

(This section will be replaced when procedures are established for the replacement of DCE functions.)

In this case, the Planning Workbench thinks it successfully activated the DPR(s) but the Job Management Server had trouble receiving the notification.

Look for the following in the **EcDpPrJobMgmtDebug.log:**

Failed in CdsEntryRead

This indicates a problem with the communication and needs to be resolved by RTSC as a "problem with DCE". Things that you can do to confirm that this is a DCE problem:

- Run dce-verify on the machine where Job Management and Planning Workbench are executed. This should check the DCE communication service and find any errors.
- Check the Debug logs of other servers. If it is a global DCE problem, there should be errors in other server's logs such as "Invalid Bindings". When the above problem happened in the Functionality lab, all of the servers had multiple "Invalid Bindings" errors.

26.17.6.9 AutoSys is not functional

1. Set the AutoSys environment variables by sourcing something that looks like /data/autotreeb/autouser/AS1.autosys.csh,
   where AS1 is an autosys instance name.

2. type chk_auto_up and verify that you see the message: "Primary Event Processor is RUNNING on machine:
   machine-name"
3. If you don't see that a Primary Event Processor is running, check with your AutoSys administrator.

Note that if Autosys will not stay up **(Autosys administrator brings it up and it goes down right away)** the following could be occuring:

- It may be possible that too many events were queued up to AutoSys while it was down. If Autosys detects a certain number of events in a short time period, it brings itself down. The only way to handle this is to keep bringing Autosys back up. Each time it will work through a few of the events before it detects "too many" and shuts down. Eventually the events will be cleared out and Autosys will stay up.
- It may be the sql server is not up for AutoSys (this may be a different server than the one needed for the PDPS database). Look for the following error messages in the AutoSys log or when you attempt to bring up JobScape:
  Couldn't create DBPROCES
  Unable to get encoded and plaintext passwords for l0sps03_srvr:FMR

26.17.6.10 AutoSys is full

This is an unlikely problem, and would only occur under the following conditions:

- The number of job boxes in the AutoSys instance > DpPrAutoSysMaxJobs/8. Look in EcDpPrJobMgmt.CFG to get this number.
- The DPR completionState in PlDataProcessingRequest is CQ_RELEASE.

What this means is that the Job Management Server got the command from Subscription Manager to release the job, but that no more jobs can fit into AutoSys at present. Wait for a DPR to finish, so that the next waiting one can be put into AutoSys.

26.17.7 PDPS Troubleshooting - SDSRV Troubleshooting

How to examine the SDSRV Database

1. Log into the Science Data Server (SDSRV) machine
2. Bring up ECS Assist
3. Select "Subsystem Manager"
4. Select "ESDT Manager"
5. Select "DB Viewer"
6. Click on Login
7. A list of data types will appear. Click on the data type and information about all granules in the archive will be displayed.

26.17.7.1 PDPS Troubleshooting - Quick Tips

Job Management fails with a "JIL FAILURE" in the .ALOG file when trying to cancel a job in AutoSys

Use AutoSys Job Definition to see who owns the job.

Then select "Adv Features" to see if the user who is trying to delete the job is in the same group for "Edit Dfn" as the user who owns the job.

A quick fix is to give the job world "Edit Dfn", but generally, whoever starts the Job Management Server should be in the same group as the person using the tool to cancel the DPR.

26.18 DPREP

26.18.1 Introduction

This section contains information to schedule and run Terra DPREP.

1 Terra DPREP is made up of three PGE's that this document refers to and each are run separately. The PGE's are titled Step1 DPREP, Step 2 DPREPand Step 3 DPREP.

2 The input files come from INGEST. These files are depicted in the three step DPREP process in **Figure 26.18.1-1**

3 The output files generated from each of the DPREP PGE's contains Ancillary Attitude, and Ephemeris data that becomes new inputs to Instrument PGE's. These Instrument PGE's will then process its satellite data with similar time span files created by DPREP.

4 The DPREP registration process for each of the three PGE's creates in the Science Data Server Archive a subscription for each of the DPREP PGE's. PGE execution then takes place in the PDPS.

5 The SSI&T effort for DPREP PGE's is similar in effort to what would be required to register any other PGE.

26.18.2 SSI&T Activity for DPREP

The Level Zero datasets are received in 2 hour chunks. The file processes in Figure 26.18.2-1 depicts the minimal time span allowable for a DPREP run. In a normal operation of DPREP, a twenty four hour time span would be prepared for. This would require additional 2 hour chunks and thus additional files of data would need to be registered. Before the registration process can take place a number of files will have to be updated to process a block of data for a particular time period. Therefore, DPREP input files will have to be identified and various templates for the SSI&T process will require annotation.

The sections that follow have been highlighted with notations as to what SSI&T process applies in the preparation of each template and the function required to register each section. With a particular function identified, other portions of this manual can be referred to for more detailed precedures to be used to carry out the full SSI&T process.

Whenever new input files are introduced or updated executables are re-introduced it is wise to set up the PGE to run from the Command Line. This will determine if what has been introduced will run error free. After a successful Command Line run it is advisable then to complete the SSI&T effort to run from the PDPS. Command Line Runs include the use the PCF to run from in the Science Data Server. PDPS runs include ESDT's and ODL files to generate internal PCF's.

DPREP File Processes

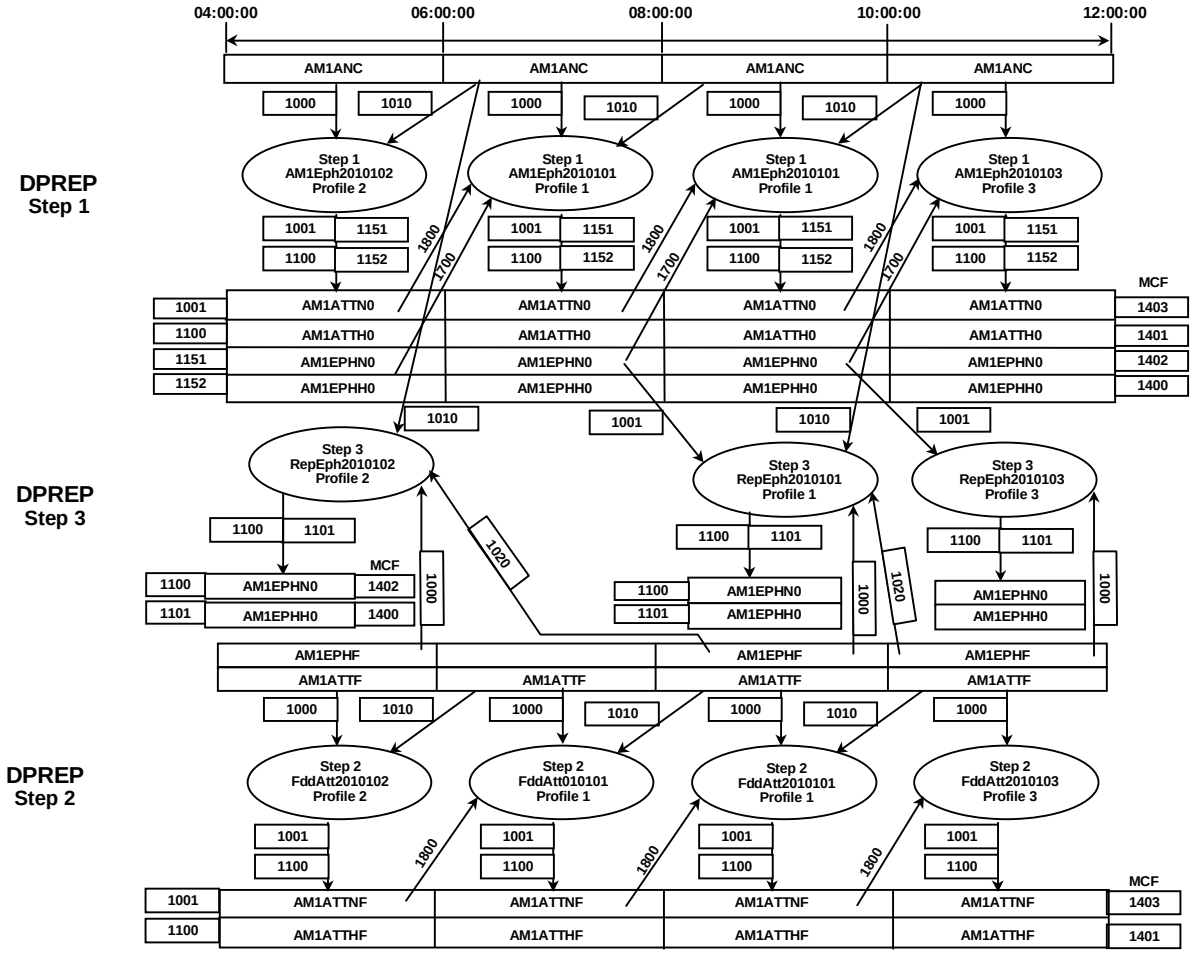


Figure 26.18.2-1. DPREP File Processes

26.18.3 DPREP Processes and Procedures

Processes and procedures are provided in the following files on an SGI machine used to support SSI&T:

DPREP README and HowToRunAM1DPREP files located at
:/usr/ecs/OPS/CUSTOM/data/DPS/

DPREP binary located: **:/usr/ecs/OPS/CUSTOM/bin/DPS/**

The following files are taken from an SSI&T SGI machine using the /data/DPS/ thread for referencing DPREP files in this section.

:

p0spg01:/usr/ecs/OPS/CUSTOM/data/DPS[45] > ls -lrt

```
total 10080
-rwxr-xr-x   1 cmops   cmops     15390 Dec 15 04:02 PGS_101
-rwxr-xr-x   1 cmops   cmops     16315 Dec 15 04:02 HowToRunDPREP
-rwxr-xr-x   1 cmops   cmops      3371 Dec 15 04:02 HowToCreateDprepTarFile
-rwxr-xr-x   1 cmops   cmops       384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_1.A
-rwxr-xr-x   1 cmops   cmops    450048 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_1.B
-rwxr-xr-x   1 cmops   cmops    449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_1.A
-rwxr-xr-x   1 cmops   cmops       384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_2
-rwxr-xr-x   1 cmops   cmops       384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_1.B
-rwxr-xr-x   1 cmops   cmops    449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_3
-rwxr-xr-x   1 cmops   cmops    449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_2
-rwxr-xr-x   1 cmops   cmops       384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_4
-rwxr-xr-x   1 cmops   cmops       384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_3
-rwxr-xr-x   1 cmops   cmops    449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_4
-rwxr-xr-x   1 cmops   cmops    225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_3
-rwxr-xr-x   1 cmops   cmops    225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_2
-rwxr-xr-x   1 cmops   cmops    225280 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_1.B
-rwxr-xp0spg01:/usr/ecs/OPS/CUSTOM/data/DPS[45] > ls -lrt
total 10080
-rwxr-xr-x   1 cmops   cmops     15390 Dec 15 04:02 PGS_101
-rwxr-xr-x   1 cmops   cmops     16315 Dec 15 04:02 HowToRunDPREP
-rwxr-xr-x   1 cmops   cmops      3371 Dec 15 04:02 HowToCreateDprepTarFile
```

```
-rwxr-xr-x  1 cmops  cmops     384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_1.A
-rwxr-xr-x  1 cmops  cmops  450048 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_1.B
-rwxr-xr-x  1 cmops  cmops  449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_1.A
-rwxr-xr-x  1 cmops  cmops     384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_2
-rwxr-xr-x  1 cmops  cmops     384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_1.B
-rwxr-xr-x  1 cmops  cmops  449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_3
-rwxr-xr-x  1 cmops  cmops  449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_2
-rwxr-xr-x  1 cmops  cmops     384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_4
-rwxr-xr-x  1 cmops  cmops     384 Dec 15 04:02
EDOS_LEVEL_ZERO_00.Profile_3
-rwxr-xr-x  1 cmops  cmops  449984 Dec 15 04:02
EDOS_LEVEL_ZERO_01.Profile_4
-rwxr-xr-x  1 cmops  cmops  225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_3
-rwxr-xr-x  1 cmops  cmops  225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_2
-rwxr-xr-x  1 cmops  cmops  225280 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_1.B
-rwxr-xr-x  1 cmops  cmops  225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_1.A
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_2
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_1
-rwxr-xr-x  1 cmops  cmops  225248 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_4
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_3
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_4.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_3.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_2.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_1.B.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_1.A.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_4.met
r-x  1 cmops  cmops  225248 Dec 15 04:02
AM1_DEFINITIVE_ATT.fdd.Profile_1.A
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_2
```

```
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_1
-rwxr-xr-x  1 cmops  cmops  225248 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_4
-rwxr-xr-x  1 cmops  cmops  411600 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_3
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_4.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_3.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_2.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_1.B.met
-rwxr-xr-x  1 cmops  cmops    1655 Dec 15 04:03 AM1ANC.Profile_1.A.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_4.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_3.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_2.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_1.B.met
-rwxr-xr-x  1 cmops  cmops    1679 Dec 15 04:03
AM1_DEFINITIVE_ATT.fdd.Profile_1.A.met
-rwxr-xr-x  1 cmops  cmops    5909 Dec 15 04:03 ReplaceEphemeris.tar.met
-rwxr-xr-x  1 cmops  cmops    5909 Dec 15 04:03 FDDAttitude.tar.met
-rwxr-xr-x  1 cmops  cmops   29206 Dec 15 04:03
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.Profile_3
-rwxr-xr-x  1 cmops  cmops   29299 Dec 15 04:03
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.Profile_2
-rwxr-xr-x  1 cmops  cmops   29818 Dec 15 04:03
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.Profile_1.B
-rwxr-xr-x  1 cmops  cmops   29818 Dec 15 04:03
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.Profile_1.A
-rwxr-xr-x  1 cmops  cmops    1663 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_3.met
-rwxr-xr-x  1 cmops  cmops    1663 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_2.met
-rwxr-xr-x  1 cmops  cmops    1663 Dec 15 04:03
AM1_REPAIR_EPH.fdd.Profile_1.met
-rwxr-xr-x  1 cmops  cmops    5909 Dec 15 04:03
AM1_Ancillary_DPREP.tar.met
-rwxr-xr-x  1 cmops  cmops   27664 Dec 15 04:03
EcDpPrAm1FddEphemerisDPREP_PGE.Step3.PCF.Profile_2
-rwxr-xr-x  1 cmops  cmops   27944 Dec 15 04:03
EcDpPrAm1FddEphemerisDPREP_PGE.Step3.PCF.Profile_1
-rwxr-xr-x  1 cmops  cmops   26902 Dec 15 04:03
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.Profile_4
-rwxr-xr-x  1 cmops  cmops   27257 Dec 15 04:03
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.Profile_3
```

```
-rwxr-xr-x    1 cmops    cmops      27184 Dec 15 04:03
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.Profile_2
-rwxr-xr-x    1 cmops    cmops      27543 Dec 15 04:03
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.Profile_1.B
-rwxr-xr-x    1 cmops    cmops      27543 Dec 15 04:03
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.Profile_1.A
-rwxr-xr-x    1 cmops    cmops      28695 Dec 15 04:03
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.Profile_4
-rwxr-xr-x    1 cmops    cmops      26584 Dec 15 04:03
EcDpPrAm1FddEphemerisDPREP_PGE.Step3.PCF.Profile_3
-rwxr-xr-x    1 cmops    cmops       5617 Dec 15 04:03 DPREP_README
-rw-rw-r--    1 cmshared cmshared     433 Dec 15 15:35 ODL
p0spg01:/usr/ecs/OPS/CUSTOM/data/DPS[46] >
data3/ecs/OPS/CUSTOM/data/DPS
```

26.18.3.1 DPREP consists of three PGE's each run separately.

1 The first step is a **ksh** script called **EcDpPrAm1EdosEphAttDPREP_PGE**, which serves as a driver for three executables:
   EcDpPrAm1EdosAncillary
   EcDpPrAm1EdosEphemerisRepair
   EcDpPrAm1ToolkitHdf
2 The second step is **EcDpPrAm1FddAttitudeDPREP_PGE.**
3 The third step is **EcDpPrAm1FddEphemerisDPREP_PGE**.

STEP ONE

EcDpPrAm1EdosAncillary reads in AM-1 L0 (EDOS) Ancillary Dataset (logical id 1000, ESDT AM1ANC). It also reads another set of AM-1 L0 (EDOS) Ancillary Dataset (logical id 1010, ESDT AM1ANC). The second set of L0 data is required to insure that incomplete orbits in the first data set get complete orbit metadata records. The only data that will be extracted from the second data set is the descending node time and longitude.

EcDpPrAm1EdosAncillary also reads in ephemeris and attitude data (toolkit native format) under logical ids 1700 (ESDT AM1EPHN0) and 1800 (ESDT AM1ATTN0). These would be the last ephemeris/attitude data sets generated from a previous run of this PGE.

EcDpPrAm1EdosEphemerisRepair

If EcDpPrAm1EdosAncillary signals a short gap was detected then EcDpPrAm1EdosEphemerisRepair reads the scratchfile created by EcDpPrAm1EdosAncillary and performs the gap fill and writes a gap filled native format ephemeris file. If no short gap is signaled then the scratch file is simply renamed to the native format ephemeris file.

EcDpPrAm1ToolkitHdf

This process takes the native format ephemeris file and produces a corresponding HDF file and a metadata file.

This PGE produces Toolkit and HDF format attitude and ephemeris data sets. The attitude data sets are produced using logical ids 1001 (ESDT AM1ATTN0) and 1100 (ESDT AM1ATTH0). The ephemeris data sets are produced using logical ids 1101 (ESDT AM1EPHN0) and 1102 (ESDT AM1EPHH0). The corresponding MCFs are accessed using logical ids 1400 (ESDT AM1EPHH0), 1401 (ESDT AM1ATTH0), 1402 (ESDT AM1EPHN0) and 1403 (ESDT AM1ATTN0).

The PGE produces an ASCII report file under logical id 2000
EcDpPrAm1EdosEphAttDPREP_PGE. Steps1ab.PCF.curr is a sample pcf used by this PGE. This pcf has directory locations and file names which may not be valid in all environments. Be careful before directly using this pcf. A sample set of input files for this PGE are
1000 EDOS_LEVEL_ZERO_00.curr
1000 EDOS_LEVEL_ZERO_01.curr
1010 EDOS_LEVEL_ZERO_00.next
1010 EDOS_LEVEL_ZERO_01.next
and a copy of each of these files is provided. The files in this group do contain small data gaps which will cause EcDpPrAm1EdosEphemerisRepair to be run.
The remaining pcfs for Step 1 DPREP:
EcDpPrAm1EdosEphAttDPREP_PGE.Steps1ab.PCF.next
EcDpPrAm1EdosEphAttDPREP.Steps1ab.PCF.next_1
will move through the sequence of the Level Zero files provided.
example; EDOS_LEVEL_ZERO_00.next is treated as current and
EDOS_LEVEL_ZERO_00.next_1 is treated as next etc...
These pcfs can be utilized at the testers discretion and are not necessary for the running of steps 2 and 3. The input files indicated in EcDpPrAm1EdosEphAttDPREP.Steps1ab.PCF.next_1 contain no gaps so testers shouldn't be concerned that EcDpPrAm1EdosEphemerisRepair is not invoked when using this pcf.
Note:
The Level Zero datasets are received in 2 hour chunks but processing can't be performed on the most recently available 2 hour chunk. Step 1 processing needs to look forward in the time stream in order to complete orbit metadata processing.

STEP TWO

EcDpPrAm1FddAttitudeDPREP_PGE reads in the current FDD Attitude Dataset under logical ID 1000 and the next FDD Attitude Dataset under logical ID 1010. It also reads in the attitude data set it produced with it's last run under logical ID 1502. The output of this process is a native format attitude file (logical ID 1001) and an HDF format attitude file (logical ID 1100). A .met file is also produced for each.
EcDpPrAm1FddAttitudeDPREP_PGE.Step2.PCF.curr is a sample pcf used by this PGE. This pcf has directory locations and file names which may not be valid in all environments. Be careful before directly using this pcf. A sample set of input files for this PGE are
1000 AM1_DEFINITIVE_ATT.fdd.curr
1010 AM1_DEFINITIVE_ATT.fdd.next
(IMPORTANT: These files contain incorrect data and were delivered only as a placeholder. EcDpPrAm1FddAttitudeDPREP_PGE will not run with these files as input. When valid FDD Definitive Attitude files become available they should be moved to these filenames and EcDpPrAm1FddAttitudeDPREP_PGE should run successfully.)
and a copy of each of these files is provided.

Note: Step 2 processing must follow 2 hours behind step 1 processing.

STEP THREE

If step1 finds too many missing data points in the ephemeris data it signals that a definitve ephemeris file is needed from FDD which EcDpPrAm1FddEphemerisDPREP_PGE will use to replace the ephemeris dataset that was generated from the Level Zero data.

EcDpPrAm1FddEphemerisDPREP_PGE reads in the definitive Ephemeris dataset received from FDD (logical ID 1000) and the EOS_AM1 Ephemeris data (logical id 1001) in toolkit native format.
It produces Replacement ephemeris datasets (logical id 1101, ESDT AM1EPHH0) and (logical id 1100, ESDT AM1EPHN0). The corresponding MCFs are accessed using logical ids 1400 (ESDT AM1EPHH0) & 1402 (ESDT AM1EPHN0)

EcDpPrAm1FddEphemerisDPREP_PGE.Step3.PCF.curr is a sample pcf used by this PGE. This pcf has directory locations and file names which may not be valid in all environments. Be careful before directly using this pcf. A sample FDD input file for this PGE is
1000 AM1_REPAIR_EPH.fdd
and a copy of this file is provided.
Toolkit Initialization Settings
SGI Application Binary Interface New 32
Disk Space Used for PGE Run 50.000 MB
Shared Memory ON
Use Test Files if SM Fails ON
Use Log Files ON
Continue if Logging Fails OFF

26.18.4 Creating DPREP Tar Files

If tar files are not available for registering the three AM-1 DPREP PGEs,

Follow the instructions found in the HowToCreateDprepTarFiles file to generate the

necessary tar files.

26.18.4.1 HowToCreateDprepTarFiles

p0spg01:/usr/ecs/OPS/CUSTOM/data/DPS[46] > more HowToCreateDprepTarFile
This file provides information to create the DPREP pge tar files

Objective : To create AM1_Ancillary_DPREP.tar & FddAttitude.tar
and using SSIT Manager archive the PGE tar files

Files needed :

- **PGS_101**
- **EcDpPrAm1EdosEphAttDPREP_PGE**
- **EcDpPrAm1EdosAncillary**
- **EcDpPrAm1EdosEphemerisRepair**
- **EcDpPrAm1ToolkitToHdf**
- **EcDpPrAm1FddAttitudeDPREP_PGE**
- **EcDpPrAm1FddEphemerisDPREP_PGE**

Steps to create AM1_Ancillary_DPREP.tar

1. Login to the science processing machine **(SGI)**

2. Check to make sure that the executables
EcDpPrAm1EdosEphAttDPREP_PGE
EcDpPrAm1EdosAncillary
EcDpPrAm1EdosEphemerisRepair
EcDpPrAm1ToolkitToHdf
are delivered and reside in **$ECS_HOME/<MODE>/CUSTOM/bin/DPS**
3. Check to make sure that the data file **PGS_101** is delivered and resides in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**

4. cd to $ECS_HOME/<MODE>/CUSTOM/data/DPS
5. /bin/tar cvf AM1_Ancillary_DPREP.tar PGS_101 -C $ECS_HOME/<MODE>/CUSTOM/bin/DPS EcDpPrAm1EdosAncillary EcDpPrAm1EdosEphAttDPREP_PGE EcDpPrAm1EdosEphemerisRepair EcDpPrAm1ToolkitToHdf
Step 5 will create the **AM1_Ancillary_DPREP.tar** file in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it will be archived
The corresponding met file for this tar file is **AM1_Ancillary_DPREP.tar.met** and is delivered to **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it (and the tar file itself) will be archived

Steps to create FddAttitude.tar

1. Login to the science processing machine **(SGI)**
2. Check to make sure that the executable **EcDpPrAm1FddAttitudeDPREP_PGE** is delivered and resides in **$ECS_HOME/<MODE>/CUSTOM/bin/DPS**
3. Check to make sure that the data file **PGS_101** is delivered and resides in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**

4. cd to $ECS_HOME/<MODE>/CUSTOM/data/DPS

5. /bin/tar cvf FDDAttitude.tar PGS_101 -C $ECS_HOME/<MODE>/CUSTOM/bin/DPS EcDpPrAm1FddAttitudeDPREP_PGE

Step 5 will create the **FDDAttitude.tar file** in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it will be archived
The corresponding met file for this tar file is **FDDAttitude.tar.met** and is delivered to **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it (and the corresponding tar file) will be archived**.**

Steps to create ReplaceEphemeris.tar

1. Login to the science processing machine (SGI)
2. Check to make sure that the executable **EcDpPrAm1FddEphemerisDPREP_PGE** is delivered and resides in **$ECS_HOME/<MODE>/CUSTOM/bin/DPS**
3. Check to make sure that the data file **PGS_101** is delivered and resides in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
4. cd to $ECS_HOME/<MODE>/CUSTOM/data/DPS
5. /bin/tar cvf ReplaceEphemeris.tar PGS_101 -C $ECS_HOME/<MODE>/CUSTOM/bin/DPS EcDpPrAm1FddEphemerisDPREP_PGE
Step 5 will create the ReplaceEphemeris.tar file in **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it will be archived
The corresponding met file for this tar file is ReplaceEphemeris.tar.met and is delivered to **$ECS_HOME/<MODE>/CUSTOM/data/DPS**
This file then needs to be ftped to the SSIT machine from where it (and the corresponding tar file) will be archived.

26.18.4.2 HowToRunDPREP

p0spg01:/usr/ecs/OPS/CUSTOM/data/DPS[48] > more HowToRunDPREP
This document contains information on how to schedule and run **AM-1 DPREP**
AM-1 DPREP is made up of **3 PGEs** that this document refers to as **Step 1 DPREP, Step 2 DPREP**, and **Step 3 DPREP:**

Step 1 DPREP is the **EDOS-**supplied ephemeris and attitude preprocessor**.**
Step 2 DPREP is the **FDD-**supplied attitude preprocessor**.**
Step 3 DPREP is the **FDD-**supplied ephemeris preprocessor**.**

Operationally, Steps 1 and 2 are scheduled daily and run independently of one another. Step 3 is scheduled and run on an as-needed basis.
In order to run, **DPREP** processing expects data to be available not only from the current segment, but also from the preceding and following segments as well. Data from the preceding and following segments are used to consistency check ephemeris and attitude data streams when the data streams bridge segment boundaries. The availability of data from the preceding and following segments is not guaranteed, however. Four data processing profiles have been developed to handle the various permutations of preceding and following data segments that could be available to **DPREP:**

Profile 1 expects data from the preceding, current, and following segments.
Profile 2 expects data from the current and following segments only.
Profile 3 expects data from the preceding and current segments only.
Profile 4 expects data from the current segment only.

These profiles are recognized by all **DPREP Steps.** The **ESDT** that is provided to **DPREP** from the preceding, current, and following data segments depends on the Step being run, however.

Profile 2 (no preceding data, but following data is available) initializes **DPREP's** processing of a given Step's ephemeris and/or attitude data stream. **Once Profile 2** has been run on a data segment, **Profile 1** (preceding and following data available) assumes processing responsibility on all data segments thereafter until data dropout or mission end is encountered. **Profile 3** (preceding data available, but no following data) then processes data segment that immediately precedes data dropout and, therefore, terminates processing on a given Step's ephemeris and/or attitude data stream**.**

In the big picture then, DPREP processing requires a single **Profile 2** to run on the first data segment, is followed by an indeterminant number of **Profile 1** processes, and is terminated by a single **Profile 3** process**.**

Profile 4 processes isolated data segments and is not likely to be scheduled operationally.

More information on the **AM-1 DPREP** can be found in document **"TERRA Spacecraft Ephemeris** and **Attitude Data Preprocessing"** (document number **184-TP-001-001).**

Note: Each PGE will use both the **Science Metadata update tool** and the **Operational Metadata Update tool,** which are located in the **SSIT Manager,** to incorporate the specific parameters listed in each of the direction sections below.

26.18.4.3 Registering DPREP PGEs

So that the following tests can be conducted, all four Profiles must be registered for all three DPREP Steps. Each DPREP PGE is registered once.

Step 1 DPREP Test Instructions

For Step 1, **current** and **following** granules are EDOS-supplied L0 Ancillary (APID 4, ShortName AM1ANC), the preceding granule is the output Toolkit format ephemeris and attitude products produced by the preceding run of Step 1 (ShortNames AM1EPHN0, AM1ATTN0). DPREP expects both the L0 Ancillary header granule and the data granule (hence, 2 granules) for the current and following L0 Ancillary. Step 1 requires 2 preceding data sets, one ephemeris product and one attitude product from the preceding Step 1 run (two granules, one for ephemeris and one for attitude).

Step 1 DPREP Specifications

Tar File Name : $ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_Ancillary_DPREP.tar

Contents : EcDpPrAm1EdosEphAttDPREP_PGE
EcDpPrAm1EdosAncillary
EcDpPrAm1EdosEphemerisRepair
EcDpPrAm1ToolkitToHdf
PGS_101
PGE Met File :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_Ancillary_DPREP.tar.met

Top Level Exe : EcDpPrAm1EdosEphAttDPREP_PGE

PGE Name : AM1Eph
Version Number : 20202

Profile 1 ODL : PGE_AM1Eph#2020101.odl
Profile 2 ODL : PGE_AM1Eph#2020102.odl
Profile 3 ODL : PGE_AM1Eph#2020103.odl
Profile 4 ODL : PGE_AM1Eph#2020104.odl

Test Scenario
The Step 1 DPREP test is divided into two parts. The first part exercises Profiles 1, 2, and 3 over four consecutive data segments. The first data segment is scheduled for Profile 2 processing, the next two segments are scheduled for Profile 1 processing, and the last scheduled for Profile 3 processing. The data segments to be processed follow.

Segment 1: 1997 July 31 04:00:00 to 1997 July 31 06:00:00 (Profile 2)
Segment 2: 1997 July 31 06:00:00 to 1997 July 31 08:00:00 (Profile 1)
Segment 3: 1997 July 31 08:00:00 to 1997 July 31 10:00:00 (Profile 1)
Segment 4: 1997 July 31 10:00:00 to 1997 July 31 12:00:00 (Profile 3)

The second Step 1 DPREP test exercises Profile 4 on an isolated data segment.

Segment 5: 1997 July 31 18:00:00 to 1997 July 31 20:00:00 (Profile 4)

There is one dynamic granule (2 files for each granule) for each of the data segments described above. Insert all five dynamic granules into the archive using the SSIT Manager's insert dynamic data tool.

Segment 1:

ESDT Short Name : AM1ANC
File Names : EDOS_LEVEL_ZERO_00.Profile_2
EDOS_LEVEL_ZERO_01.Profile_2
Met file : AM1ANC.Profile_2.met

Time Range : 1997 July 31 04:00:00 to 1997 July 31 06:00:00

Segment 2:

ESDT Short Name : AM1ANC
File Names : EDOS_LEVEL_ZERO_00.Profile_1.A
EDOS_LEVEL_ZERO_01.Profile_1.A
Met file : AM1ANC.Profile_1.A.met
Time Range : 1997 July 31 06:00:00 to 1997 July 31 08:00:00

Segment 3:

ESDT Short Name : AM1ANC
File Names : EDOS_LEVEL_ZERO_00.Profile_1.B
EDOS_LEVEL_ZERO_01.Profile_1.B
Met file : AM1ANC.Profile_1.B.met
Time Range : 1997 July 31 08:00:00 to 1997 July 31 10:00:00

Segment 4:

ESDT Short Name : AM1ANC
File Names : EDOS_LEVEL_ZERO_00.Profile_3
EDOS_LEVEL_ZERO_01.Profile_3
Met file : AM1ANC.Profile_3.met
Time Range : 1997 July 31 10:00:00 to 1997 July 31 12:00:00

Segment 5:

ESDT Short Name : AM1ANC
File Names : EDOS_LEVEL_ZERO_00.Profile_4
EDOS_LEVEL_ZERO_01.Profile_4
Met file : AM1ANC.Profile_4.met
Time Range : 1997 July 31 18:00:00 to 1997 July 31 20:00:00

Create three production requests for the following time ranges. The number of DPRs that should be created are given below.

Request	Timespan	Profile	DPRs
1	1997 July 31 04:00:00 to 06:00:00	2	1
2	1997 July 31 06:00:00 to 10:00:00	1	2
3	1997 July 31 10:00:00 to 12:00:00	3	1

Schedule all three production requests to run as a single production plan. This completes the first part of the Step 1 DPREP test.

Next, create one production request that for the time range that follow.

Request	Timespan	Profile	DPRs
1	1997 July 31 18:00:00 to 20:00:00	4	1

Schedule this a seperate production plan. This completes the second part of the Step 1 DPREP test.

Results
Find output granules for each of the timespans described above for ESDTs AM1EPHN0, AM1EPHH0, AM1ATTN0, and AM1ATTH0. ESDT AM1EPHN0 is the Toolkit format ephemeris granule, while ESDT AM1EPHH0 is the HDF format ephemeris granule. ESDT AM1ATTN0 is the Toolkit format attitude granule, while ESDT AM1ATTH0 is the HDF format attitude granule.

Step 2 DPREP Test Instructions

For Step 2, current and following granules are FDD-supplied attitude (ShortName AM1ATTF), the preceding granule is the output Toolkit format attitude product produced by the preceding run of Step 2 (ShortName AM1ATTNF).

Step 2 DPREP Specifications

Tar File Name : $ECS_HOME/<MODE>/CUSTOM/data/DPS/FDDAttitude.tar
Contents : EcDpPrAm1FddAttitudeDPREP_PGE
PGS_101
PGE Met File : $ECS_HOME/<MODE>/CUSTOM/data/DPS/FDDAttitude.tar.met

Top Level Exe : EcDpPrAm1FddAttitudeDPREP_PGE

PGE Name : FddAtt
Version Number : 20201

Profile 1 ODL : PGE_FddAtt#2020101.odl
Profile 2 ODL : PGE_FddAtt#2020102.odl
Profile 3 ODL : PGE_FddAtt#2020103.odl
Profile 4 ODL : PGE_FddAtt#2020104.odl

Test Scenario

The Step 2 DPREP test, like the Step 1 test scenario, is divided into two parts. The first part exercises Profiles 1, 2, and 3 over four consecutive data segments. The first data segment is scheduled for Profile 2 processing, the next two segments are scheduled for Profile 1 processing, and the last scheduled for Profile 3 processing. The data segments to be processed follow.

Segment 1: 1997 July 31 04:00:00 to 1997 July 31 06:00:00 (Profile 2)
Segment 2: 1997 July 31 06:00:00 to 1997 July 31 08:00:00 (Profile 1)

Segment 3: 1997 July 31 08:00:00 to 1997 July 31 10:00:00 (Profile 1)
Segment 4: 1997 July 31 10:00:00 to 1997 July 31 12:00:00 (Profile 3)

The second Step 2 DPREP test exercises Profile 4 on an isolated data segment.

Segment 5: 1997 July 31 18:00:00 to 1997 July 31 20:00:00 (Profile 4)

There is one granule for each of the data segments described above. Insert all five granules into the archive using the SSIT Manager's insert dynamic data tool.

Segment 1:

ESDT Short Name : AM1ATTF
File Names : AM1_DEFINITIVE_ATT.fdd.Profile_2
Met file : AM1_DEFINITIVE_ATT.fdd.Profile_2.met
Time Range : 1997 July 31 04:00:00 to 1997 July 31 06:00:00

Segment 2:

ESDT Short Name : AM1ATTF
File Names : AM1_DEFINITIVE_ATT.fdd.Profile_1.A
Met file : AM1_DEFINITIVE_ATT.fdd.Profile_1.A.met
Time Range : 1997 July 31 06:00:00 to 1997 July 31 08:00:00

Segment 3:

ESDT Short Name : AM1ATTF
File Names : AM1_DEFINITIVE_ATT.fdd.Profile_1.B
Met file : AM1_DEFINITIVE_ATT.fdd.Profile_1.B.met
Time Range : 1997 July 31 08:00:00 to 1997 July 31 10:00:00

Segment 4:
ESDT Short Name : AM1ATTF
File Names : AM1_DEFINITIVE_ATT.fdd.Profile_3
Met file : AM1_DEFINITIVE_ATT.fdd.Profile_3.met
Time Range : 1997 July 31 10:00:00 to 1997 July 31 12:00:00

Segment 5:
ESDT Short Name : AM1ATTF
File Names : AM1_DEFINITIVE_ATT.fdd.Profile_4
Met file : AM1_DEFINITIVE_ATT.fdd.Profile_4.met
Time Range : 1997 July 31 18:00:00 to 1997 July 31 20:00:00

Create three production requests for the following time ranges. The number of DPRs that should be created are given below.

Request	Timespan	Profile	DPRs
1	1997 July 31 04:00:00 to 06:00:00	2	1
2	1997 July 31 06:00:00 to 10:00:00	1	2
3	1997 July 31 10:00:00 to 12:00:00	3	1

Schedule all three production requests to run as a single production plan. This completes the first part of the Step 2 DPREP test.

Next, create one production request that for the time range that follow.

Request	Timespan	Profile	DPRs
1	1997 July 31 18:00:00 to 20:00:00	4	1

Schedule this a seperate production plan. This completes the second part of the Step 2 DPREP test.

These plans can be scheduled and run completely independently of the Step 1 DPREP plans described earlier.

Results
Find output granules for each of the timespans described above for ESDTs AM1ATTNF and AM1ATTHF. ESDT AM1ATTNF is the Toolkit format attitude granule, while ESDT AM1ATTHF is the HDF format attitude granule.

Step 3 DPREP Test Instructions

For Step 3, the current granule is FDD-supplied ephemeris (ShortName AM1EPHF), the preceding granule is the output Toolkit format ephemeris product produced by preceding run of Step 1, and the following granule is EDOS-supplied L0 Ancillary (ShortName AM1ANC).

Step 3 DPREP Specifications
Tar File Name : $ECS_HOME/<MODE>/CUSTOM/data/DPS/ReplaceEphemeris.tar
Contents : EcDpPrAm1FddAttitudeDPREP_PGE
PGS_101
PGE Met File : $ECS_HOME/<MODE>/CUSTOM/data/DPS/ReplaceEphemeris.tar.met

Top Level Exe : EcDpPrAm1FddEphemerisDPREP_PGE

PGE Name : RepEph
Version Number : 20201

Profile 1 ODL : PGE_RepEph#2020101.odl
Profile 2 ODL : PGE_RepEph#2020102.odl
Profile 3 ODL : PGE_RepEph#2020103.odl

Profile 4 ODL : PGE_RepEph#2020104.odl

Test Scenario

Operationally, Step 3 AM-1 DPREP will most likely be scheduled to run on single data segments. Hence, the Step 3 test scenario consists of scheduling and running four plans that exercise the four Profiles individually on a single data segments. The data segments to be processed follow.

Segment 1: 1997 July 31 04:00:00 to 1997 July 31 06:00:00 (Profile 2)
Segment 2: 1997 July 31 08:00:00 to 1997 July 31 10:00:00 (Profile 1)
Segment 3: 1997 July 31 10:00:00 to 1997 July 31 12:00:00 (Profile 3)
Segment 4: 1997 July 31 18:00:00 to 1997 July 31 20:00:00 (Profile 4)

There is one granule for each of the data segments described above. Insert all four granules into the archive using the SSIT Manager's insert dynamic data tool.

Segment 1:

ESDT Short Name : AM1EPHF
File Names : AM1_REPAIR_EPH.fdd.Profile_2
Met file : AM1_REPAIR_EPH.fdd.Profile_2.met
Time Range : 1997 July 31 04:00:00 to 1997 July 31 06:00:00

Segment 2:
ESDT Short Name : AM1EPHF
File Names : AM1_REPAIR_EPH.fdd.Profile_1
Met file : AM1_REPAIR_EPH.fdd.Profile_1.met
Time Range : 1997 July 31 08:00:00 to 1997 July 31 10:00:00

Segment 3:
ESDT Short Name : AM1EPHF
File Names : AM1_REPAIR_EPH.fdd.Profile_3
Met file : AM1_REPAIR_EPH.fdd.Profile_3.met
Time Range : 1997 July 31 10:00:00 to 1997 July 31 12:00:00

Segment 4:
ESDT Short Name : AM1EPHF
File Names : AM1_REPAIR_EPH.fdd.Profile_4
Met file : AM1_REPAIR_EPH.fdd.Profile_4.met
Time Range : 1997 July 31 18:00:00 to 1997 July 31 20:00:00

Create four individual production plans consisting of a single DPR for each of the time ranges that follow. Schedule and allow a given plan to complete before scheduling the next.

Request	Timespan	Profile	DPRs
1	1997 July 31 04:00:00 to 06:00:00	2	1
2	1997 July 31 06:00:00 to 10:00:00	1	1
3	1997 July 31 10:00:00 to 12:00:00	3	1
4	1997 July 31 18:00:00 to 20:00:00	4	1

These plans can be scheduled and run only after the Step 1 AM-1 DPREP test has been successfully completed. The Step 1 DPREP test scenario must be completed before CONTINUING with the Step 3 test scenario. Continuation of the Step 3 test scenario requires the archive environment that was produced from the successful completion of the Step 1 DPREP test scenario.

Results

Find output granules for each of the timespans described above for ESDTs AM1EPHN0 and AM1EPHH0. ESDT AM1EPHN0 is the Toolkit format ephemeris granule, while ESDT AM1EPHH0 is the HDF format ephemeris granule. Output from Step 3 processing is placed in the same ESDTs as the Step 1 products.

26.18.5 Results of DPREP processing from earlier work

26.18.5.1 Step1 DPREP directions taken from earlier work:

There are 2 profiles for DPREP. Profile 1 takes in previous DPREP output and is expected to be run most of the time at the DAACs. Profile 2 takes in only the AM1 Ancillary data and will be run only for the first run of DPREP (because there is no previous output). Both profiles should be registered and executed.

PGENAME : AM1Eph
PGEVERSION : 2.0
PROFILE : 1
TopLevelShellName : EcDpPrAm1EdosEphAttDPREP_PGE
PGENAME : AM1Eph
PGEVERSION : 2.0
PROFILE : 2
TopLevelShellName : EcDpPrAm1EdosEphAttDPREP_PGE (same executable, only insert it once)
PGE Tar file location :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_Ancillary_DPREP.tar
PGE Met file location :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_Ancillary_DPREP.tar.met
Science Software Release 4

The PGE tar file contains the executables: EcDpPrAm1EdosEphAttDPREP_PGE, EcDpPrAm1EdosAncillary, EcDpPrAm1EdosEphemerisRepair, EcDpPrAm1ToolkitHdf.and the toolkit message file PGS_101
This PGE has no static data. This means Insert Static from SSIT can be skipped for this PGE.

However, there are 4 dynamic granules (2 files each) that this PGE needs as input. These 4 granules can be inserted from SSIT **Insert Test Dynamic Tool**

Granule 1:
ESDT Short Name : AM1ANC
Version : 001
Multi File granule : Y
Directory : $ECS_HOME/<MODE>/CUSTOM/data/DPS
File Names : EDOS_LEVEL_ZERO_00.curr, EDOS_LEVEL_ZERO_01.curr
Met file : AM1ANC.curr.met

Granule 2:
ESDT Short Name : AM1ANC
Version : 001
Multi File granule : Y
Directory : $ECS_HOME/<MODE>/CUSTOM/data/DPS
File Names : EDOS_LEVEL_ZERO_00.next, EDOS_LEVEL_ZERO_01.next
Met file : AM1ANC.next.met

Granule 3:
ESDT Short Name : AM1ANC
Version : 001
Multi File granule : Y
Directory : $ECS_HOME/<MODE>/CUSTOM/data/DPS
File Names : EDOS_LEVEL_ZERO_00.next_1, EDOS_LEVEL_ZERO_01.next_1
Met file : AM1ANC.next_1.met

Granule 4:
ESDT Short Name : AM1ANC
Version : 001
Multi File granule : Y
Directory : $ECS_HOME/<MODE>/CUSTOM/data/DPS
File Names : EDOS_LEVEL_ZERO_00.next_2, EDOS_LEVEL_ZERO_01.next_2
Met file : AM1ANC.next_2.met
3 PRs need to be created, two for Profile 1 and one for Profile 2. Create the PR for Profile 2 first (it runs on earlier data) and then create the PR for Profile 1.
The time frame for Production request editor for Profile 2 is 1998 June 30 00:00:00 to 1998 June 30 02:00:00
The time frame for Production request editor for Profile 1 is 1998 June 30 02:00:00 to 1998 June 30 04:00:00
The time frame for Production request editor for the second Profile 1 is 1998 June 30 04:00:00 to 1998 June 30 06:00:00

26.18.5.2 Step2 DPREP directions:

****

Step 2 DPREP can run ONLY after all three PRs for Step 1 DPREP have completed

****

There are 2 profiles for Step 2 DPREP. Profile 1 takes in previous Step 2 DPREP output and is expected to be run most of the time at the DAACs. Profile 2 takes in only the Fdd Att data and will be run only for the first run of DPREP (because there is no previous output). Both profiles should be registered and executed.

PGENAME : FddAtt
PGEVERSION : 1.0
PROFILE : 1
TopLevelShellName : EcDpPrAm1FddAttitudeDPREP_PGE

PGENAME : FddAtt
PGEVERSION : 1.0
PROFILE : 2
TopLevelShellName : EcDpPrAm1FddAttitudeDPREP_PGE (same executable, only insert it once)

PGE Tar file location : $ECS_HOME/<MODE>/CUSTOM/data/DPS/FDDAttitude.tar
PGE Met file location : $ECS_HOME/<MODE>/CUSTOM/data/DPS/FDDAttitude.tar.met
Science Software Version: 1.0

The PGE tar file contains the executable EcDpPrAm1FddAttitudeDPREP_PGE and the toolkit message file PGS_101
This PGE has no static data. This means Insert Static from SSIT can be skipped for this PGE. However, there are 3 dynamic granules that this PGE needs as input. These 3 granules can be inserted from SSIT insert test dynamic tool

Granule 1:
ESDT Short Name : AM1ATTF
Version : 001
Multi File granule : N
File Name :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.curr
Met file :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.curr.met

Granule 2:
ESDT Short Name : AM1ATTF
Version : 001
Multi File granule : N
File Name :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.next

Met file :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.next.met

Granule 3:
ESDT Short Name : AM1ATTF
Version : 001
Multi File granule : N
File Name :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.next_1
Met file :
$ECS_HOME/<MODE>/CUSTOM/data/DPS/AM1_DEFINITIVE_ATT.fdd.next_1.met

2 PRs need to be created, one for Profile 1 and one for Profile 2. Create the PR for Profile 2 first (it runs on earlier data) and then create the PR for Profile 1.

The time frame for Production request editor for Profile 2 is 1998 June 30 02:00:00 to 1998 June 30 04:00:00

The time frame for Production request editor for Profile 1 is 1998 June 30 04:00:00 to 1998 June 30 06:00:00

Note: The Data files and tar files are delivered only to the Science Data Server and cannot be directly accessed by SSIT. In order to insert these files they must be moved to the SSIT server manually.

26.18.5.3 DPREP Step1 profile1 PCF

PRODUCT INPUT FILES
##
(**Initial Construction Record**:)
1000|P0420004AAAAAAAAAAAAAA98300080000000.PDS|/net/p0drg01/dss_amass/testdata/instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<A) insert UR here>||2
(**Current 2 hour Data File**)
1000|P0420004AAAAAAAAAAAAAA98300080000001.PDS|/net/p0drg01/dss_amass/testdata/instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<B) insert UR here>||1
:
(**Future 2 hour Construction Record**)

1010|P0420004AAAAAAAAAAAAAA98300100000000.PDS|/net/p0drg01/dss_amass/testdata/instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<C)
insert UR here>||2
(**Future 2 hour Data File**)
1010|P0420004AAAAAAAAAAAAAA98300100000001.PDS|/net/p0drg01/dss_amass/testdata/instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<D)
insert UR here>||1

The last ephemeris/attitude data sets generated. **(Found in PGE template)**

Ephemeris (Toolkit native format) **(Previous Time Range)**

1500|EOS_AM1_Ephemeris.21206.step1b.eph|/home/cmts1/DPREP/demooutput||<E) insert UR here>||1
Attitude (Toolkit native format)
1502|EOS_AM1_Attitude.21206.step1.att|/home/cmts1/DPREP/demooutput||<F) insert UR here>||1
:
? PRODUCT OUTPUT FILES
##
(Current Time Range)
--
Toolkit attitude datasets (output of EcDpPrAm1EdosAncillary).
--
1001|EOS_AM1_Attitude.21208.step1.att|/home/cmts1/DPREP/demooutput||||1
1100|EOS_AM1_Attitude.21208.step1.hdf|/home/cmts1/DPREP/demooutput||||1
--
Toolkit ephemeris datasets (output of EcDpPrAm1EdosAncillary).
--
1102|EOS_AM1_Ephemeris.21208.hdf|/home/cmts1/DPREP/demooutput||||1
--
Gap-filled HDF ephemeris dataset (output of EcDpPrAm1EphemerisGapFill).
--
1151|EOS_AM1_Ephemeris.21208.step1b.eph|/home/cmts1/DPREP/demooutput||||1
--
Gap-filled Toolkit ephemeris dataset (output of
EcDpPrAm1FormatNativeEphemeris).
--
1152|EOS_AM1_Ephemeris.21208.step1b.hdf|/home/cmts1/DPREP/demooutput||||1
:
SUPPORT OUTPUT FILES
##
10100|LogStatus|/home/cmts1/DPREP/logs||||1
10101|LogReport|/home/cmts1/DPREP/logs||||1
10102|LogUser|/home/cmts1/DPREP/logs||||1
10103|TmpStatus|||||1
10104|TmpReport|||||1
10105|TmpUser|||||1
10110|MailFile|||||1
:
? USER DEFINED RUNTIME PARAMETERS
##
999|No Previous Data Set; 1=true, 0=false.|0 **(Profile 1 set to 0, Profile 2 set to 1)**
--
Toolkit version for which this PCF was intended.
DO NOT REMOVE THIS VERSION ENTRY!
--

10220|Toolkit version string|DAAC B.0 TK5.2.4 **(Caution: May need to change to higher level Toolkit)**
END

26.18.5.4 DPREP Step1 profile2 PCF (Profile 2 is start up PCF)

```
PRODUCT INPUT FILES
##############################################################################
1000|P0420004AAAAAAAAAAAAAA98300060000000.PDS|/net/p0drg01/dss_amass/testdata/
instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<A) insert UR here>||2
1000|P0420004AAAAAAAAAAAAAA98300060000001.PDS|/net/p0drg01/dss_amass/testdata/
instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<B) insert UR here>||1
1010|P0420004AAAAAAAAAAAAAA98300080000000.PDS|/net/p0drg01/dss_amass/testdata/
instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<C) insert UR here>||2
1010|P0420004AAAAAAAAAAAAAA98300080000001.PDS|/net/p0drg01/dss_amass/testdata/
instrument_data/AM1/EDOS_HK_ANC/EDOSL0_ANC_PDS_0002||<D) insert UR here>||1
# ----------------------------------------------------------------------
# The last ephemeris/attitude data sets generated.
# ----------------------------------------------------------------------
# Ephemeris (Toolkit native format)
#1500|EOS_AM1_Ephemeris.prev.eph|.||<E) insert UR here>||1
# Attitude (Toolkit native format)
#1502|EOS_AM1_Attitude.21206.step1.att|/home/cmts1/DPREP/output||<F) insert UR here>||1
:
?  PRODUCT OUTPUT FILES
##############################################################################
# ----------------------------------------------------------------------
# Toolkit attitude datasets (output of EcDpPrAm1EdosAncillary).
# ----------------------------------------------------------------------
1001|EOS_AM1_Attitude.21206.step1.att|/home/cmts1/DPREP/demooutput||||1
1100|EOS_AM1_Attitude.21206.step1.hdf|/home/cmts1/DPREP/demooutput||||1
# ----------------------------------------------------------------------
# Toolkit ephemeris datasets (output of EcDpPrAm1EdosAncillary).
# ----------------------------------------------------------------------
1102|EOS_AM1_Ephemeris.21206.hdf|/home/cmts1/DPREP/demooutput||||1
# ----------------------------------------------------------------------
# Gap-filled HDF ephemeris dataset (output of EcDpPrAm1EphemerisGapFill).
# ----------------------------------------------------------------------
1151|EOS_AM1_Ephemeris.21206.step1b.eph|/home/cmts1/DPREP/demooutput||||1
# ----------------------------------------------------------------------
# Gap-filled Toolkit ephemeris dataset (output of EcDpPrAm1FormatNativeEphemeris).
# ----------------------------------------------------------------------
1152|EOS_AM1_Ephemeris.21206.step1b.hdf|/home/cmts1/DPREP/demooutput||||1
?  SUPPORT OUTPUT FILES
##############################################################################
10100|LogStatus|/home/cmts1/DPREP/logs||||1
10101|LogReport|/home/cmts1/DPREP/logs||||1
```

```
10102|LogUser|/home/cmts1/DPREP/logs||||1
10103|TmpStatus|||||1
10104|TmpReport|||||1
10105|TmpUser|||||1
10110|MailFile|||||1
:
?   USER DEFINED RUNTIME PARAMETERS
##############################################################################
999|No Previous Data Set; 1=true, 0=false.|1
# ------------------------------------------------------------------
# Toolkit version for which this PCF was intended.
# DO NOT REMOVE THIS VERSION ENTRY!
# ------------------------------------------------------------------
10220|Toolkit version string|DAAC B.0 TK5.2.4
?   END
```

26.18.5.5 DPREP Step2 profile1 PCF

```
PRODUCT INPUT FILES
##############################################################################
1000|AM1_DEFATT_080000_212_1997_01.FDD|/home/cmts1/DPREP/am1defatt||<A) insert
UR here>||1
:
1010|AM1_DEFATT_100000_212_1997_01.FDD|/home/cmts1/DPREP/am1defatt||<B) insert
UR here>||1
# ------------------------------------------------------------------
# The last ephemeris/attitude data sets generated.
# ------------------------------------------------------------------
# Attitude (Toolkit native format)
1502|EOS_AM1_Attitude.21206.step2.att|/home/cmts1/DPREP/demooutput||<C) insert UR
here>||1
:
#1400|AM1EPHMH.mcf|.||||1
1401|AM1ATTHF.MCF|/home/cmts1/DPREP/mcf||||1
#1402|AM1EPHMN.mcf|.||||1
1403|AM1ATTNF.MCF|/home/cmts1/DPREP/mcf||||1
10250|MCF|||||1
10252|GetAttr.temp|||||1
10254|MCFWrite.temp|||||1
:
10501|EOS_AM1_Ephemeris.21208.step1b.eph|/home/cmts1/DPREP/demooutput||||2
10501|EOS_AM1_Ephemeris.21210.step1b.eph|/home/cmts1/DPREP/demooutput||||1
10502|INSERT_ATTITUDE_FILES_HERE|||||1
:
?   PRODUCT OUTPUT FILES
##############################################################################
:1001|EOS_AM1_Attitude.21208.step2.att|/home/cmts1/DPREP/demooutput||||1
```

```
1100|EOS_AM1_Attitude.21208.step2.hdf|/home/cmts1/DPREP/demooutput||||1
:
?   USER DEFINED RUNTIME PARAMETERS
##############################################################################
:
999|First Mission Data Set; 1=true, 0=false.|0
:10220|Toolkit version string|DAAC B.0 TK5.2.4
:END
```

26.18.5.6 DREP Step2 profile2 PCF

```
PRODUCT INPUT FILES
##############################################################################
1000|AM1_DEFATT_060000_212_1997_01.FDD|/home/cmts1/DPREP/am1defatt||<A) insert
UR here>||1
:
1010|AM1_DEFATT_080000_212_1997_01.FDD|/home/cmts1/DPREP/am1defatt||<B) insert
UR here>||1
# ----------------------------------------------------------------
# The last ephemeris/attitude data sets generated.
# ----------------------------------------------------------------
# Attitude (Toolkit native format)
1502|EOS_AM1_Attitude.21206.step1.att|/home/cmts1/DPREP/demooutput||<C) insert UR
here>||1
:
1400|AM1EPHMH.mcf|.||||1
1401|AM1ATTHF.MCF|/home/cmts1/DPREP/mcf||||1
1402|AM1EPHMN.mcf|.||||1
1403|AM1ATTNF.MCF|/home/cmts1/DPREP/mcf||||1
10250|MCF|||||1
10252|GetAttr.temp|||||1
10254|MCFWrite.temp|||||1
:
10501|EOS_AM1_Ephemeris.21206.step1b.eph|/home/cmts1/DPREP/demooutput||||2
10501|EOS_AM1_Ephemeris.21208.step1b.eph|/home/cmts1/DPREP/demooutput||||1
10502|INSERT_ATTITUDE_FILES_HERE|||||1
:
?   PRODUCT OUTPUT FILES
##############################################################################
1001|EOS_AM1_Attitude.21206.step2.att|/home/cmts1/DPREP/demooutput||||1
1100|EOS_AM1_Attitude.21206.step2.hdf|/home/cmts1/DPREP/demooutput||||1
:
?   USER DEFINED RUNTIME PARAMETERS
##############################################################################
:999|First Mission Data Set; 1=true, 0=false.|1
:
10220|Toolkit version string|DAAC B.0 TK5.2.4
```

:
? END

26.18.5.7 Setups for DPREP

set path = (/usr/ecs/TS1/CUSTOM/TOOLKIT/toolkit/bin/sgi_daac_f90 $path)
setenv PGS_PC_INFO_FILE /home/cmts1/DPREP/pcf/Step1DP21206.PCF
setenv PGSMSG /home/cmts1/DPREP/msg
setenv PGSHOME /usr/ecs/TS1/CUSTOM/toolkit
setenv PGS_PC_INFO_FILE /home/cmts1/DPREP/pcf/Step1DP21208.PCF
setenv PGSMSG /home/cmts1/DPREP/msg
setenv PGSHOME /usr/ecs/TS1/CUSTOM/toolkit
setenv PGS_PC_INFO_FILE /home/cmts1/DPREP/pcf/Step1DP21210.PCF
setenv PGSMSG /home/cmts1/DPREP/msg
setenv PGSHOME /usr/ecs/TS1/CUSTOM/toolkit
setenv PGS_PC_INFO_FILE /home/cmts1/DPREP/pcf/Step2DP21206.PCF
setenv PGSMSG /home/cmts1/DPREP/msg
setenv PGSHOME /usr/ecs/TS1/CUSTOM/toolkit
setenv PGS_PC_INFO_FILE /home/cmts1/DPREP/pcf/Step2DP21208.PCF
* setenv PGSMSG /home/cmts1/DPREP/msg
* setenv PGSHOME /usr/ecs/TS1/CUSTOM/toolkit

Note: The **setenv** parameters are used for the **Command Line PGE** test. Only the **(*)** are used for **PDPS PGE** runs.

26.18.6 FDD Ephemeris Reprocessing Due To QA Failure

The following events occur when FDD-supplied ephemeris fails QA checking:

1. A subscription is placed on ESDT AM1EPHNF for replacement data (ESDT AM1EPHF) when QaPercentMissingData or QaOutOfBoundsData is greater than or equal to 1.
2. The AM1 DPREP FDD ephemeris processor ("RepEph" or "Step 3" DPREP PGE) detects a data gap or an out-of-bounds data point in the FDD-supplied ephemeris timeline (supplied through ESDT ShortName AM1EPHF). Assuming that no fatal errors are otherwise encountered, the PGE exits with status code 216, a success condition that indicates that the FDD-supplied ephemeris data set must be replaced by a re-delivery of the *same* data set. The *QaPercentMissingData* or QaOutOfBoundsData metadata has been set to a value greater than or equal to 1.
3. A subscription triggers when the DPREP output data set (i.e. AM1EPHNF) that is generated in step 2 is inserted into the archive. The subscription triggers because of *QaPercentMissingData* or QaOutOfBoundsData being greater than or equal to 1.
4. The GDAAC operator receives an E-mail notification indicating that the subscription has triggered. The universal reference (UR) of the AM1EPHNF granule that triggered the subscription is provided in the E-mail.
5. The GDAAC operator uses the database ID imbedded within the UR to identify the AM1EPHNF granule in the archive using *EcCoDbViewer*. Retrieve the start and end times of this granule.
6. The GDAAC operator telephones the FOT (telephone number 301-614-5431) and requests an FDD ephemeris data set (ESDT ShortName AM1EPHF) be sent for the time period

spanned by the start and end times determined in step 5. The operator explicitly states what the replacement granule's start and end time must be, to the second. The replacement ephemeris data set will be sent to both the GDAAC and the LDAAC.

7. The GDAAC operator notifies LDAAC operations (telephone number 757-864-9197) that a new AM1EPHF ephemeris data set is going to arrive for the time period determined in step 5.
8. The GDAAC and LDAAC operators schedule the AM-1 DPREP FDD ephemeris processor ("RepEph" or "Step 3" DPREP PGE) to process the time interval determined in step 5. If data replacement occurs on the first granule following a period of data dropout, Profile 2 processing must be scheduled. Otherwise, Profile 1 can be scheduled to run.

26.18.6.1 Boot-up of EDOS Ephemeris Processing

AM-1 DPREP EDOS ephemeris processor ("AM1Eph" or "Step 1" DPREP) Profile 2 requires a special procedure in order to achieve boot-up processing at the start of the mission and following periods of data dropout. The required steps follow:

1. Wait for the first EDOS-supplied AM1ANC data set that follows the interval of data dropout to be ingested.
2. Use the *EcCoDbViewer* archive browser to determine the start and end time of the granule.
3. Call the FOT (telephone number 301-614-5431) and ask to speak with the on-line engineer.
4. Ask the on-line engineer to provide the orbit number at the granule start time determined in step 2.
5. Telephone LDAAC operations (telephone number 757-864-9197). Pass-on the orbit number determined in step 4 to the operator so the LDAAC can proceed with steps 6 through 9. This avoids having both the LDAAC and the GDAAC perform steps 1 through 4.
6. In directory /usr/ecs/OPS/CUSTOM/data/DPS, locate ODL file PGE_AM1Eph*VVVVV*02.odl. *VVVVV* is the version number of the operational AM-1 DPREP.
7. Edit this file using vi. Locate logical ID 998 (*PGE_PARAMETER_NAME InitialOrbitNumber*) within the ODL file and insert the orbit number provided by the FOT on-line engineer into the line beginning with *PGE_PARAMETER_DEFAULT*. This step requires *allmode* privileges in order to edit the ODL file.
8. Register the "AM1Eph" AM-1 DPREP.
9. Schedule "AM1Eph" AM-1 DPREP, Profile 2 to process the interval given by the start and end times determined in step 2.

26.18.6.2 FDD Replacement Ephemeris Processing

The following events occur in FDD replacement ephemeris processing:

1. A subscription is placed on ESDT AM1EPHN0 for replacement data (ESDT AM1EPHF) when QaPercentMissingData is greater than or equal to 1.

2. The AM1 DPREP EDOS ephemeris processor ("AM1Eph" or "Step 1" DPREP PGE) detects a long data gap in the EDOS-supplied ephemeris timeline (supplied through ESDT ShortName AM1ANC). Assuming that no fatal errors are otherwise encountered, the PGE exits with status code 216, a success condition that indicates the replace ephemeris condition has been detected. The *QaPercentMissingData* metadata is set to a value greater than or equal to 1, depending on the size of the data gap that is detected.
3. A subscription triggers when the DPREP output data set (i.e. AM1EPHN0) that is generated in step 2 is inserted into the archive. The subscription triggers because of *QaPercentMissingData* being greater than or equal to 1.
4. The GDAAC operator receives an E-mail notification indicating that the FDD replacement ephemeris data set subscription has triggered. The universal reference (UR) of the AM1EPHN0 granule that triggered the subscription is provided in the E-mail.
5. The GDAAC operator uses the database ID imbedded within the UR to identify the AM1EPHN0 granule in the archive using *EcCoDbViewer*. Retrieve the start and end times of this granule.
6. Given the start and end granules times, the GDAAC operator derives the replacement time range. The procedure will be demonstrated by example. If the 2-hour AM1EPHN0 granule ideally spans 22h – 24h of day 2000-06-07, the replacement ephemeris granule time span is

   Start time = 2000-06-07 22:00:00.000
   End time = 2000-06-07 23:59:59.000

   Replacement data starts on the hour and ends one second prior to the start of the subsequent 2-hour granule.
7. The GDAAC operator telephones the FOT (telephone number 301-614-5431) and requests an FDD replacement data set (ESDT ShortName AM1EPHF) be sent for the time period spanned by the start and end times determined in step 6. The operator explicitly states what the replacement granule's start and end time must be, to the second. The replacement ephemeris data set will be sent to both the GDAAC and the LDAAC.
8. The GDAAC operator notifies LDAAC operations (telephone number 757-864-9197) that a replacement ephemeris data set (AM1EPHF) is going to arrive for the time period determined in step 6.
9. The GDAAC and LDAAC operators schedule the AM-1 DPREP replacement ephemeris processor ("RepEph" or "Step 3" DPREP PGE) to process the time interval determined in step 6. If data replacement occurs on the first granule following a period of data dropout, Profile 2 processing must be scheduled. Otherwise, Profile 1 can be scheduled to run.

26.18.6.3 FDD Attitude Reprocessing Due To QA Failure

The following events occur when FDD-supplied attitude fails QA checking:

a. A subscription is placed on ESDT AM1ATTNF for replacement data (ESDT AM1ATTF) when QaPercentMissingData or QaOutOfBoundsData is greater than or equal to 1.
b. The AM1 DPREP FDD attitude processor ("FddAtt" or "Step 2" DPREP PGE) detects a data gap or an out-of-bounds data point in the FDD-supplied attitude timeline (supplied through ESDT ShortName AM1ATTF). Assuming that no fatal errors are otherwise encountered, the PGE exits with status code 216, a success condition that indicates that the FDD-supplied attitude data set must be replaced by a re-delivery of the *same* data set.

The *QaPercentMissingData* or QaOutOfBoundsData metadata has been set to a value greater than or equal to 1.

c. A subscription triggers when the DPREP output data set (i.e. AM1ATTNF) that is generated in step 2 is inserted into the archive. The subscription triggers because of *QaPercentMissingData* or QaOutOfBoundsData being greater than or equal to 1.
d. The GDAAC operator receives an E-mail notification indicating that the subscription has triggered. The universal reference (UR) of the AM1ATTNF granule that triggered the subscription is provided in the E-mail.
e. The GDAAC operator uses the database ID imbedded within the UR to identify the AM1ATTNF granule in the archive using *EcCoDbViewer*. Retrieve the start and end times of this granule.
f. The GDAAC operator telephones the FOT (telephone number 301-614-5431) and requests an FDD attitude data set (ESDT ShortName AM1ATTF) be sent for the time period spanned by the start and end times determined in step 5. The operator explicitly states what the replacement granule's start and end time must be, to the *millisecond*. The replacement attitude data set will be sent to both the GDAAC and the LDAAC.
g. The GDAAC operator notifies LDAAC operations (telephone number 757-864-9197) that a new AM1ATTF attitude data set is going to arrive for the time period determined in step 5.
h. The GDAAC and LDAAC operators schedule the AM-1 DPREP FDD attitude processor ("FddAtt" or "Step 2" DPREP PGE) to process the time interval determined in step 5. If data replacement occurs on the first granule following a period of data dropout, Profile 2 processing must be scheduled. Otherwise, Profile 1 can be scheduled to run.

Technical Notes:

1. The DPREP for Aqua ephemeris will be run separately for predictive and definitive orbit data (e.g. at GSFC and/or Langley)

2. The predictive data output will be used by the Aqua Attitude PGE to produce attitude for AIRS. This is being done because predictive is available sooner for AIRS need for an early update for their processing requirements.

3. The attitude DPREP will be run later when definitive orbit data is available to produce definitive attitude for Aqua.

4. The latest Toolkit requires the tag: EOSPMGIIS to be in the ephemris and attitude files. This replaces EOSPM1.

26.19 PGE Chaining

26.19.1 Chaining PGE's

1. Create PRs (so that DPRs) for the PGEs to be chained.

   This can be done by using PR Editor. Follow the same procedure as creating independent PR.

A few points need to be noticed. Let's say among the chained PGEs, the output of PGE A will be the input of PGE B.1) In ESDT odl for this shared granule, "DYNAMIC_FLAG" has to be set to "I", i.e., dynamic internal. 2) First create PR for PGE A, then for PGE B. Otherwise PGE B PR may not be able to be generated.

2. Create the plan for a bunch of PRs which are chained.

   In Work Bench GUI, 1) pull down "file" menu and select "new" to create the new plan; 2) highlight all PRs that are chain by clicking on their names on "unscheduled" area of Production Request area; 3) click schedule button to schedule these PRs.

3. Activate the plan.

   In the Workbench GUI, click "activate" button, a GUI will pop up to ask for saving the plan. Answer "yes". Then another GUI will pop up to confirm whether to really activate the plan. Answer "yes" and the lowest level of DPR(s) in the chain will kick off.

   In the pdps database, the PlDataProcessingRequest table is where the PRs are successfully generated, the "completionState" for all DPRs in the chain are "NULL". When the plan is successfully activated, the "CompletionState" for lowest level of DPR(s) is changed from "NULL" to "STARTED". The high level of DPR(s) in the chain is changed from "NULL" to "CQ_HOLD". Eventually, the low level of DPR(s) finish so that the input for high level of DPR(s) become available, Then the high level DPR(s) kick off and the "CompletionState" then changes from "CQ_HOLD" to "STARTED".

26.20 Updating the Orbit Model

26.20.1 Introduction to Updating the Orbit Model

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

be necessary. The following procedures are provided to bring about an updated Orbit Model within ECS.

26.20.2 Procedures to Update the Orbit Model

Upon receipt of updated orbit parameters: ORBIT_ NUMBER,
ORBIT_PERIOD,
ORBIT Path Number and
ORBIT_START Time.

Proceed with the following steps.

---

1 Telnet or Rlogin to location (ais) system where ODL files are stored. ie; **"/usr/ecs/OPS/CUSTOM/data/DPS/ODL"**

2 Select the ORBIT.odl that is currently being used.

3 Using **vi, update the following files with the new parameter values received:**

- **ORBIT_AM1.odl** and/or **ORBIT_EOSAM1.odl** if they both are in use.

4 Have someone double check your entries for accuracy before preoceeding to the SSIT Manager for registering the new ODL file in the PDPS system.

5 For **Test Data** only; determine the Instrument PGE ODL that will be updated. MISR, MODIS etc.

- Using **vi, update the corresponding PATHMAP_Instrument_.odl** file with the new parameter values received.
- Ensure that the ABSOLUTE_PATH and MAPPED_PATH parameters agree with those in the new ORBIT_XXXX.odl.

6 SSI&T personnel will execute an Orbit Model Update by running a Dummy PGE established for this purpose at each of the DAAC's. Note: A dummy PGE is ran since a normal PGE cannot be re-registered if any DPRs exist in the system.

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

26.21 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. It is divided into subsections by category of failure. Some types of failures are common to many steps in the SSIT process, so their investigation follows a common path. Thus, the SSIT user should refer to the following sections on the basis of the underlying process (e.g., file insertion to the Science Data Server) rather than by the distinct stage in the SSIT procedure.

This section is divided into six main categories of failure, each section will discuss ESDT Installation failure

- File Insert Failure
- File Acquire Failure
- DPR Generation Failure
- Scheduling a DPR Failure

- PGE Execution Failure

26.21.1 Description

ESDT versioning has been implemented in this release, which means a DLL can be shared by more than one ESDT descriptor files.. In this release, the DLL file which will be used during installation is specified in the ESDT descriptor file. Therefore during ESDT installation, a user only need to provide ESDT descriptor file name instead of ESDT descriptor file and DLL file.

26.21.2 Handling an ESDT Installation Failure

During ESDT installation, there are two types of failure. One is general failure, which means no ESDT can be installed. This case is usually caused by improper operations, which means the Servers may not be brought up properly, or a login is expired etc. The other is particular ESDT installation failure, which means certain type of ESDT can not be installed, but other ESDTs may be able to installed. This case is relatively hard to for a user trace the problem. The problem is usually caused by particular attributes in the ESDT descriptor file, which means the ESDT descriptor file is incompatible with Science Data Server Database table definitions and also some attributes in the ESDT descriptor file may violate the validation rule defined in the Data Dictionary. Therefore as a general rule, a user needs to identify the failure type first. The following sections are general guidance for a user to handling an ESDT installation failure.

Check the status of the servers: When you install an ESDT, there are certain servers, which has to be brought up properly. If either or all the servers are down or hasn'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.

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 file and DLL. Those entries are DSSDESCINPUTDIR and DSSDLLDIR. In the ESDT descriptor file, there is an attribute name which 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.

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.

Check the Science Data Server ALOG File: During ESDT installation the Science Data Server writes status messages to two log files, EcDsScienceDataServer.ALOG and EcDsScienceDataServerDebug.log. 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.
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 which is 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 or has a server connectivity problem.

26.21.3 Insert File Failure

Description

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. Also, one phase (Destaging) of the execution of a PGE by the system involves the insertion of output data products to the Science Data Server.

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. Once a data type has been made known to the PDPS database via PGE registration, files of that type may be inserted to the Science Data Server regardless of the PGE for which they will serve as input or output. The PGEEXE.tar file can only successfully insert following registration of the particular PGE.

26.21.3.1 Handling an Insert Failure

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.
Check for ESDT in SDSRV Database: 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 data base using SQL commands.

Table 26.21.3.1-1. 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 list the steps required to view the SDSRV data base using a database viewer.

Table 26.21.3.1-2. 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 could succeed..

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.

Sample Listing of ESDT Entries in Advertising Database

IoAdAdvMaster
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 data base using SQL commands.

26.21.4 Viewing the Advertising Database Using SQL Commands

Quick-Step Procedures:

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

Table 26.21.4-1. 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> DBPASSWD: <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). 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 describes ESDT removal from Science Data Server, Advertising and Data Dictionary while reinstallation is covered in Adding ESDTs to the System.

Section 26.11 describes ESDT removal from Science Data Server, Advertising and Data Dictionary while reinstallation is also covered in Section 26.11.

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 26.11.5 for more information on how to install a volume group.

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.

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.

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.

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.

Verify that the Servers are Running: File insertion requires not only that SDSRV be running, but also the IOS 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.

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.

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	/usr/bin/ps -ef | grep *mode*	press Enter

26.21.5 Acquire Failure

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.

26.21.5.1 Handling an Acquire failure:

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

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
:DsSrWorkingCollectionDistributeO
neDistributFile
Msg: File 2 to be distributed: SCMOD03.0011369.met

then SDSRV has completed its role in the acquire. Therefore acquire script usually will indicate success.

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 Enter
3	vi EcDsScienceDataServer.ALOG	press Enter

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 DsDbCleanGranules.

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.

Check the Archive Server ALOG File: Acquire success from the Sciecne 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.

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 Enter
3	vi EcDsStArchiveServer.ALOG	press Enter

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.

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 Enter
3	ls -lrt	press Enter

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

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 Enter
3	vi EcDsStStagingDiskServer.ALOG or EcDsStStagingMonitorServer.ALOG	press Enter

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

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 Enter
3	df -k **.**	press Enter

26.21.6 Failure During DPR Generation

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.

26.21.6.1 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 EcPlPREditor.ALOG resided at /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 are kept in differenct tables. Inside the table PlDataTypeMaster, 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. On the other hand, inside the table PlDataTypeMaster, 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. If the PGE contains static files, the UR for the static files have to been included in PlDataGranuleShort table before a DPR can be successfully generated. For dynamic granules, the UR values will become available in the PlDataGranuleShort 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.

26.21.7 Failure Scheduling a DPR

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.

26.21.7.1 Handling a DPR Scheduling failure

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

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

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

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	/home/opscm/dbr/dbrowser-syb	press Return
4	enter into GUI login*: PDPSservername, PDPSusername, PDPSpassword*	
5	select pdps_*mode*	
6	select "sample data" under "view" menu	
7	select PlDataProcessingRequest	

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

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 -Updpsusername -Ppassword -Spdpsserver	press Return
3	use pdps_*mode*	press Return
4	go	press Return
5	select * from PlDataProcessingRequest where dprId = "*dprID*"	press Return
6	go	press Return

Check the PDPS Database: PlDataGranuleShort: 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 PlDataGranuleShort, under universalReference. Standalone PGE, those not running as part of a PGE chain, need 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" instead of "internal", 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 JobScape within Autosys will change, as soon as the input granules are available.

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

26.21.8 Failures During Execution

Description

PGEs scheduled for execution in the system follow seven 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.

26.21.8.1 Handling a Failures During Execution

Resource Allocation: The first stage of PGE processing in Autosys is Resource Allocation. 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 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.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

3 At the UNIX prompt, type **vi *DPR#. ALOG or DPR#A.ALOG***, then press **Enter.**

Staging: The Staging step in processing involves acquiring files from SDSRV. Thus, it should be handled as in Acquire Failure, above.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

At the UNIX prompt, type **vi *DPR#. ALOG or DPR#S.ALOG***, then press **Enter.**

Preprocessing: Preprocessing rarely completely fails. It may not generate the system PCF file correctly. Placing a "hold" on the PGE execution stage (through Autosys, in the Job Scape GUI, the Job Console button brings up the Job Console - PGE processing stages can be put on hold using the "On Hold" button.). While execution is on hold, the system PCF can be inspected to see whether it matches expectations.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

At the UNIX prompt, type **vi *DPR#. ALOG or DPR#P.ALOG***, then press **Enter.**

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***, then press **Enter**.

3 At the UNIX prompt, type **vi pgeId.Pcf**, then press **Enter.**

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 these files, 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***, then press **Enter**.

3 At the UNIX prompt, type **vi pgeId.Pcf** or **pgeId.TkStatus**, then press **Enter.**

Postprocessing: Postprocessing does not often fail, but it may show as a failure in Autosys if the Execution stage has failed.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

3 At the UNIX prompt, type **vi *DPR#. ALOG or DPR#p.ALOG***, then press **Enter.**

Destaging: Destaging involves the insertion of output data granules into SDSRV.

Thus the section on 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.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

At the UNIX prompt, type **vi *DPR#. ALOG or DPR#I.ALOG***, then press **Enter.**

Deallocation: Rarely are there cases of failure in Deallocation.

To inspect the Data Processing ALOG file, 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/logs**, then press **Enter**.

At the UNIX prompt, type **vi *DPR#. ALOG or DPR#D.ALOG***, then press **Enter.**

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

26.22 Using IQ Software to Create Reports

26.22.1 Creating Reports Using IQ Software

ECS no longer plans to offer a Report Generator GUI. Consequently, DAAC operations personnel must use other means to generate various types of reports.
IQ (Intelligent Query) software is a set of commercial off-the-shelf (COTS) products that provides flexible access to the PDPS database from which data for reports can be retrieved. The cost of that flexibility is a somewhat complicated process for initially setting up reports. However, once a particular type of report has been set up, reports can be generated fairly quickly.
The procedure for creating reports using IQ software starts with the assumption that the Production Planner has logged in to the system.

Creating Reports Using IQ Software

NOTE: If using an X-Terminal, it may be necessary to add the following line to the **.Xdefaults** file in the home directory before performing the task for the first time:

```
iqx*background:                grey
```

NOTE: Commands in Steps 1 through 8 are typed at a UNIX system prompt.

1 Type **rlogin** ***hostname*** refers to the host (e.g., **e0mss21**, **g0mss21**, **l0mss21**, or **n0mss21**) on which GUIs are to be launched during the current operating session. Multiple hostnames can be specified on the same line.

2 Type **setenv DISPLAY** ***clientname*****:0.0** then press the **Return/Enter** key.

- Use either the X terminal/workstation IP address or the machine-name for the ***clientname***.
- When using secure shell, the DISPLAY variable is set just once, before logging in to remote hosts. If it were to be reset after logging in to a remote host, the security features would be compromised.

3 Open another UNIX (terminal) window.

4 Start the log-in to the Applications Server host by typing **/tools/bin/ssh** ***hostname*** (e.g., **e0mss21**, **g0mss21**, **l0mss21**, or **n0mss21**) in the new window then press the **Return/Enter** key.

- If you receive the message, **Host key not found from the list of known hosts. Are you sure you want to continue connecting (yes/no)?** type **yes** ("**y**" alone will not work).
- If you have previously set up a secure shell passphrase and executed **sshremote**, a prompt to **Enter passphrase for RSA key '<*user@localhost*>'** appears; continue with Step 5.
- If you have not previously set up a secure shell passphrase; go to Step 6.

5 If a prompt to **Enter passphrase for RSA key '<*user@localhost*>'** appears, type your ***Passphrase*** then press the **Return/Enter** key.

- Go to Step 7.

6 At the **<*user@remotehost*>'s password:** prompt type your ***Password*** then press the **Return/Enter** key.

7 Type **setenv ECS_HOME /usr/ecs/** then press the **Return/Enter** key.

- When logging in as a system user (e.g., cmshared), the ECS_HOME variable may be set automatically so it may not be necessary to perform this step.

8 Type **cd /usr/ecs/*MODE*/COTS/iqx5** then press **Return/Enter**.

- Change directory to the directory containing the IQ software (directory path may vary from site to site).
- The ***MODE*** will most likely be one of the following operating modes:
- OPS (for normal operation).
- TS1 (for Science Software Integration and Test (SSI&T)).
- TS2 (new version checkout).
- Note that the separate subdirectories under /usr/ecs apply to (describe) different operating modes.

9 Type **iqx &** then press **Return/Enter**.

- If the GUIs are not displayed when the command **iqx** is given, try using **./iqx** instead.
- The **iqx IQView** GUI (Figure 26.19.6-1) and either the **iqx Sybase Database Logon** GUI (Figure 26.19.6-2) or **iqx Open IQView** GUI (Figure 26.19.6- 3) are displayed.

— If IQViews have been defined previously, they are listed on the **iqx Open IQView** GUI (Figure 26.19.6-3); otherwise, a list of database tables is displayed.

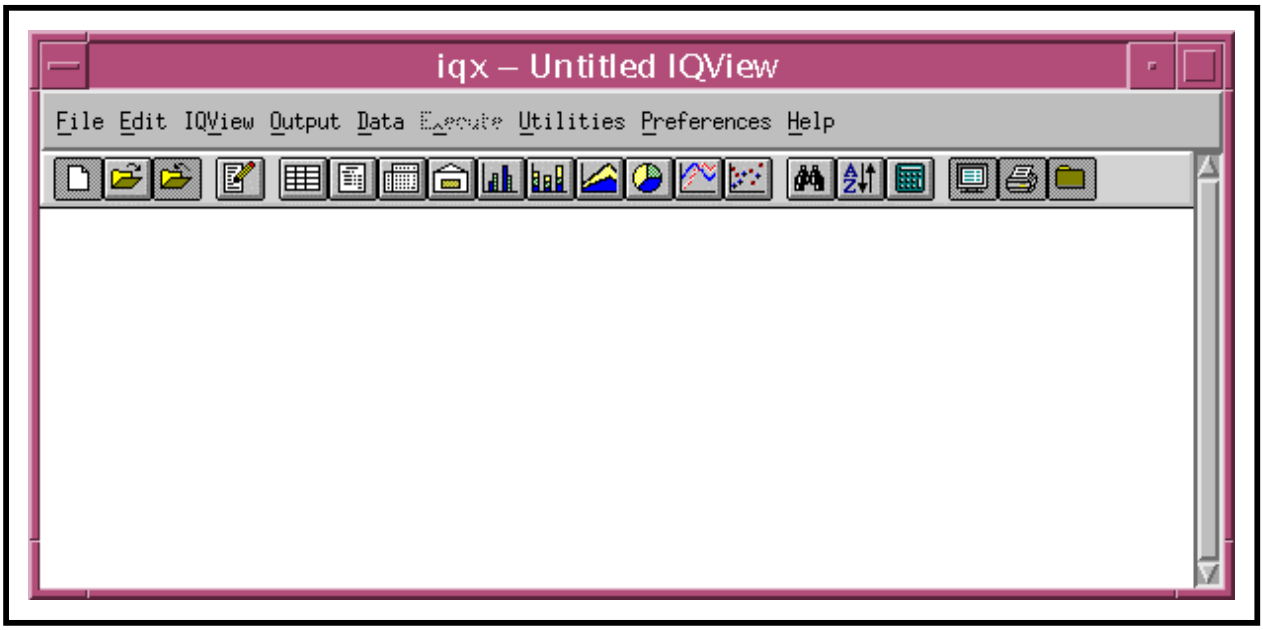


Figure 26.22.1-1. iqx IQView GUI

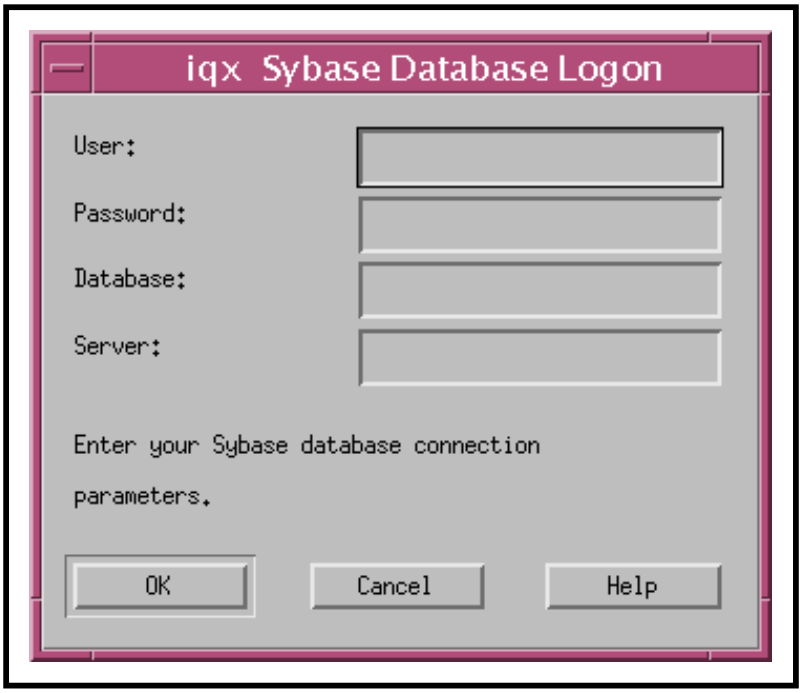


Figure 26.22.1-2. iqx Sybase Database Logon GUI

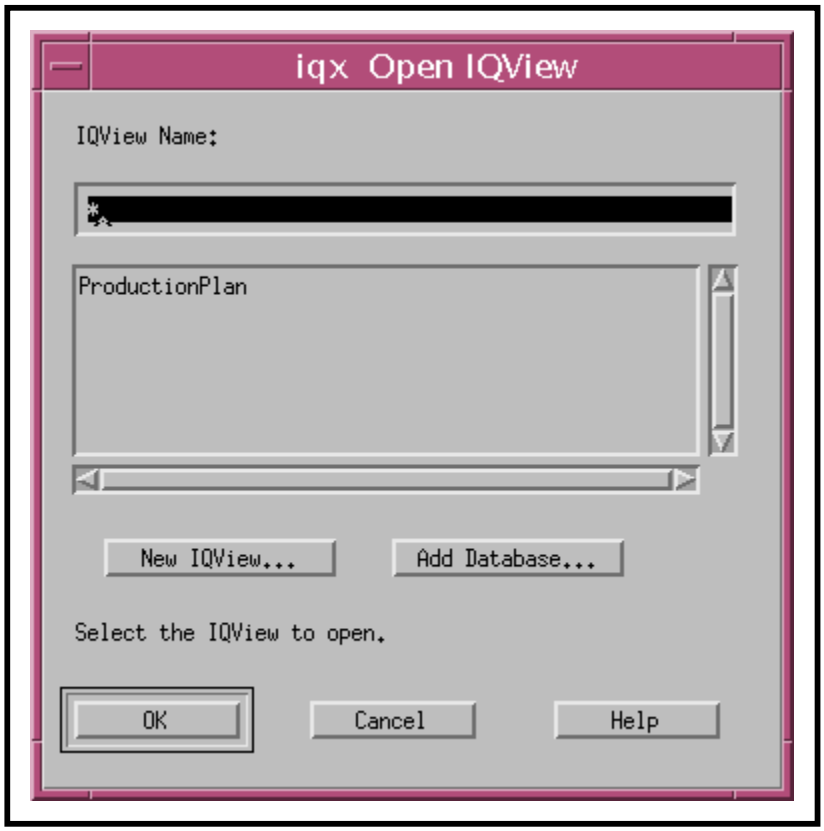


Figure 26.22.1-3. iqx Open IQView GUI

- If the **iqx License** dialogue box (Figure 26.22.1-4) is displayed, click on the **Cancel** button.

— The **iqx IQView** GUI (Figure 26.22.1-1) and either the **iqx Sybase Database Logon** GUI (Figure 26.22.1-2) or **iqx Open IQView** GUI (Figure 26.22.1-3) are displayed.

10 If the **iqx Sybase Database Logon** GUI (Figure 26.22.1-2) is displayed, go to Step 16.

11 If the **iqx Open IQView** GUI (Figure 26.22.1-3) is displayed and the desired IQView has been defined previously, perform Steps 12 through 14; otherwise, go to Step 15.

- If IQViews have been defined previously, they are listed on the **iqx Open IQView** GUI (Figure 26.22.1-3); otherwise, a list of database tables is displayed.

12 If the desired IQView has been defined previously, highlight the IQView to be opened by clicking on its entry in the list of IQViews.

13 Click on the **OK** button.

— The **iqx Sybase Database Logon** GUI (Figure 26.22.1-2) is displayed.

14 Go to Step 16.

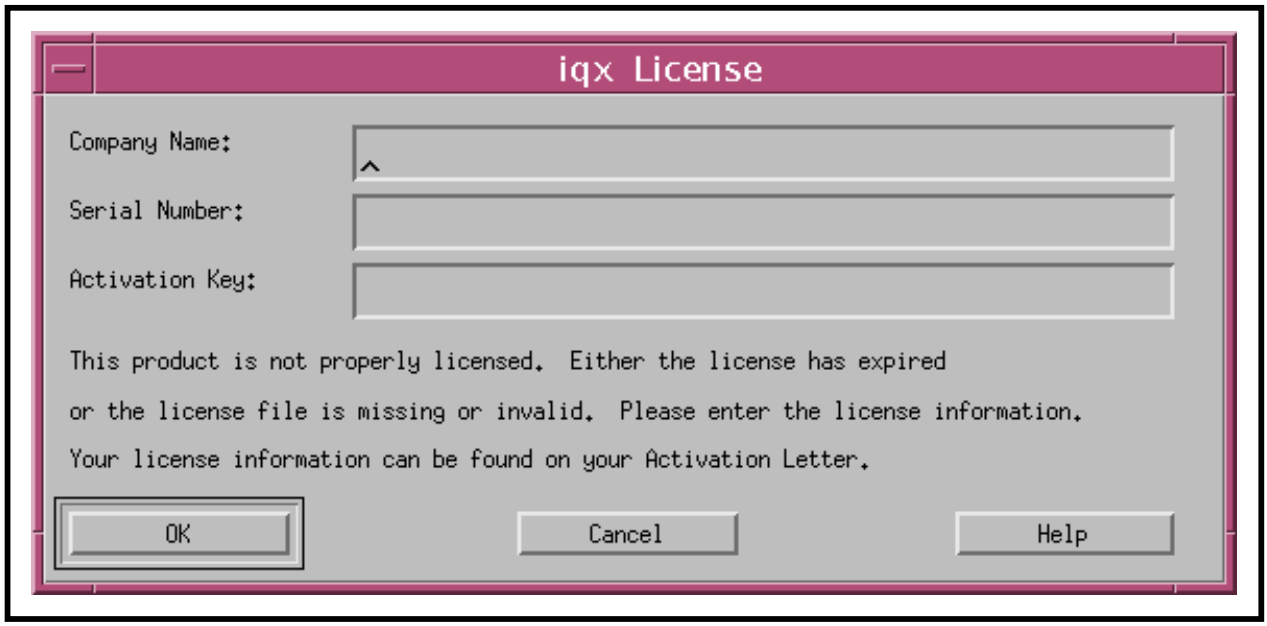


Figure 26.22.1-4. iqx License Dialogue Box

15 If the **iqx Open IQView** GUI (Figure 26.22.1-3) is displayed and the desired IQView has not been defined previously, click on the **Add Database…** button.

— The **iqx Sybase Database Logon** GUI (Figure 26.22.1-2) is displayed.

16 When the **iqx Sybase Database Logon** GUI (Figure 26.22.1-2) is displayed, type the appropriate entries in the following fields:

- **User:**

— For example: **pdpsUsr**

— The DAAC Database Administrator can provide the actual values to be entered.

— **Password:**

— For example: **dbpa$$wd**

— **Database:**

— For example: **pdps_TS1**

— **Server:**

— For example: **x0pls02_srvr**

— Click on the **OK** button.

— Either the **iqx Open IQView** GUI (Figure 26.22.1-3) or the **iqx IQView** GUI (Figure 26.22.1-1) is displayed.

— If the **iqx Open IQView** GUI (Figure 26.22.1-3) is displayed, continue with Step 19; if the **iqx IQView** GUI (Figure 26.22.1-1) is displayed, go to Step 22.

— Click on the **New IQView…** button.

— The **iqx Table Selection** GUI (Figure 26.22.1-5) is displayed.

— Move database table names between the **Available Tables:** and **Selected Tables:** lists as necessary by selecting (highlighting) the name of the table to be moved, then clicking on either the **Select** or **Remove** button (as applicable) to move the table name to the other list.

— Database tables and the columns within each table are described in the 311-series documents (e.g., 311-CD-520-001, Release 5B Planning and Data Processing Subsystem Database Design and Schema Specifications for the ECS Project). The documents are available on the ECS Data Handling System (i.e., at http://edhs1.gsfc.nasa.gov).

21 When the desired table(s) has/have been moved to the **Selected Tables:** list, click on the **OK** button.

- The **iqx IQView** GUI (Figure 26.22.1-1) is displayed.

22 Select **Output → Columnar** from the pull-down menu.

- The **iqx Columnar Output** GUI (Figure 26.22.1-6) is displayed.

23 Move database table column names between the **Available columns:** and **Selected columns:** lists as necessary by selecting (highlighting) the column to be moved, then clicking on either the **Select** or **Remove** button (as applicable) to move the column name to the other list.

- The order in which columns are listed in the **Selected columns:** list is the order in which the columns will be listed in the eventual report.
- Database tables and the columns within each table are described in the 311-series documents (e.g., 311-CD-520-001, Release 5B Planning and Data Processing Subsystem Database Design and Schema Specifications for the ECS Project). The documents are available on the ECS Data Handling System (i.e., at http://edhs1.gsfc.nasa.gov).

24 If changing the order in which columns are listed in the **Selected columns:** list, select (highlight) the column to be moved, then click on the **Up** or **Down** button as necessary to reposition the selected column.

4. Highlighted column changes position in the **Selected columns:** list.

25 When the desired columns have been moved to the **Selected columns:** list, click on the **OK** button.

- The **iqx IQView** GUI (Figure 26.22.1- 7) is displayed.

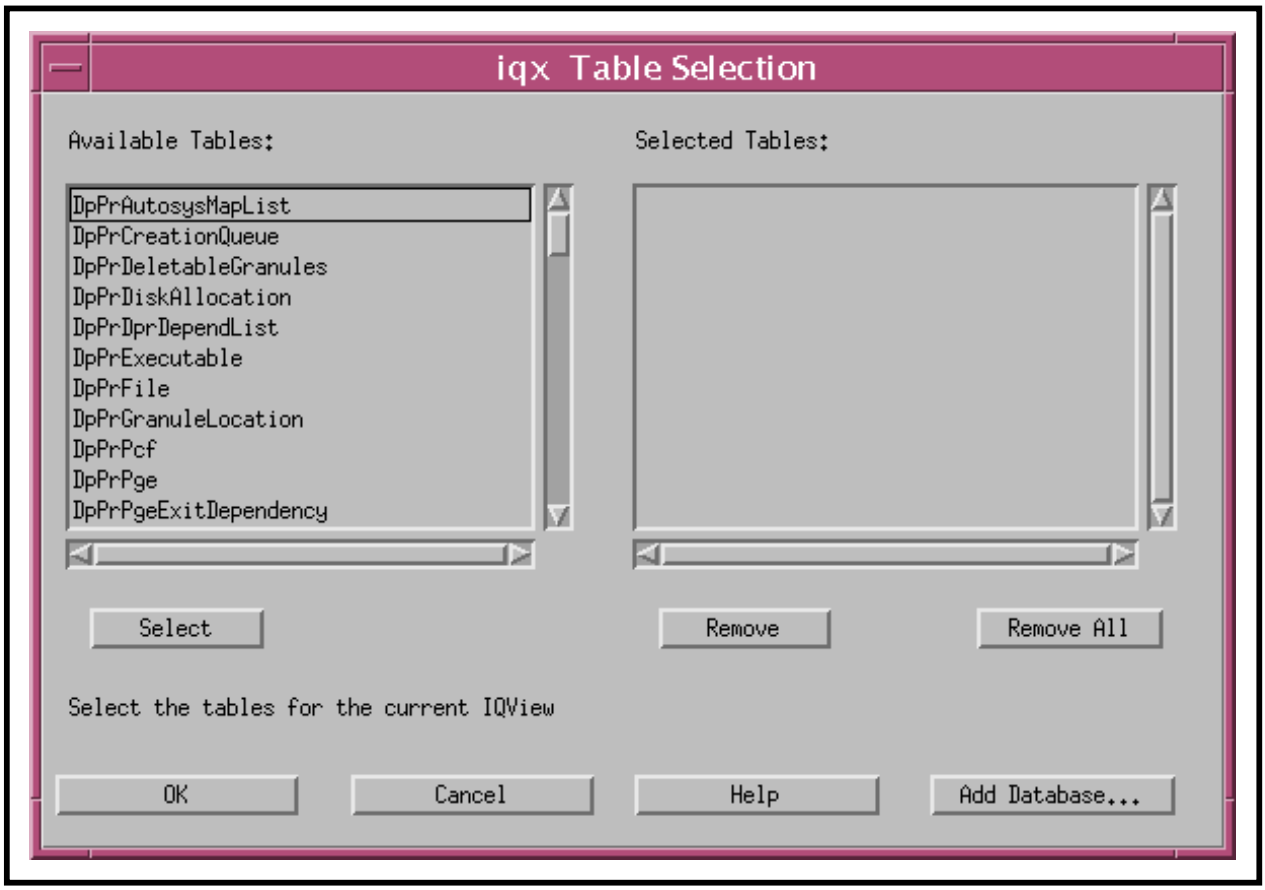


Figure 26.22.1-5. iqx Table Selection GUI

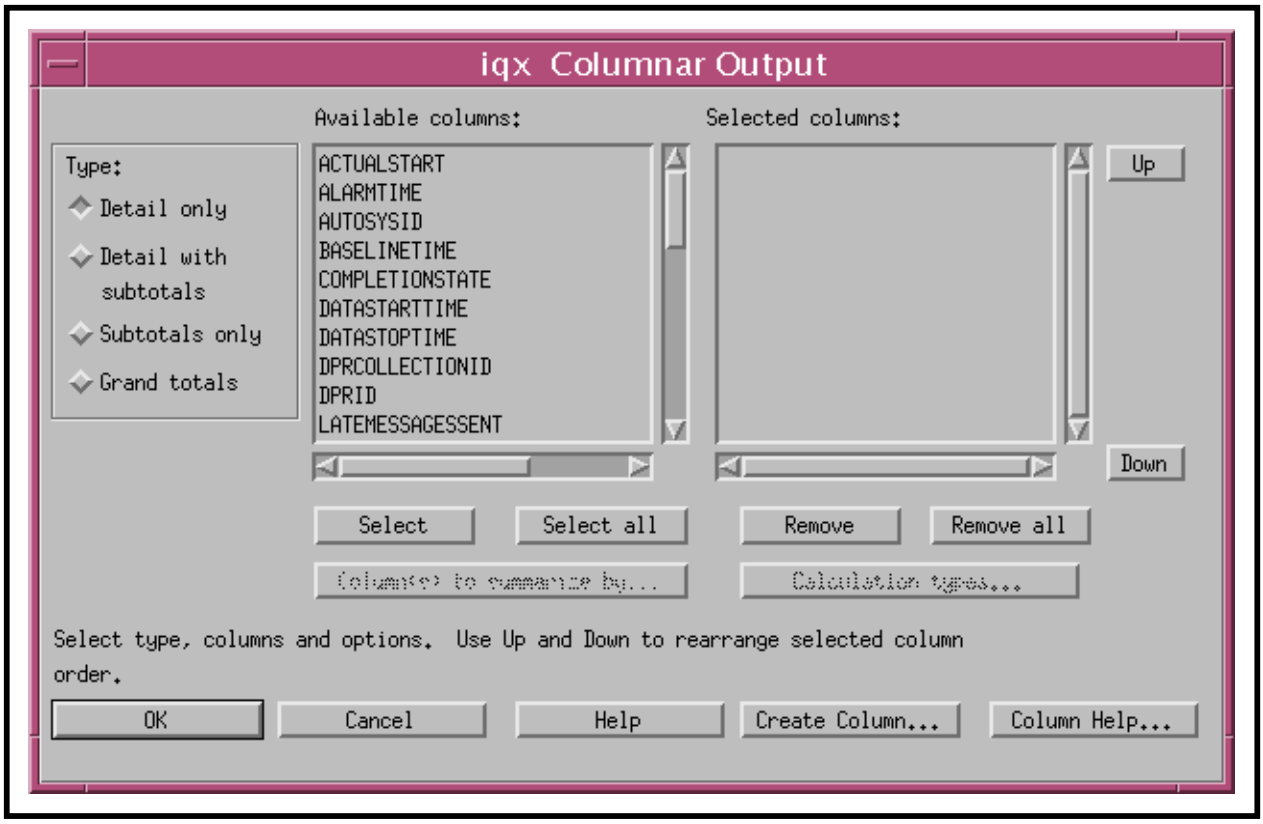


Figure 26.22.1-6. iqx Columnar Output GUI

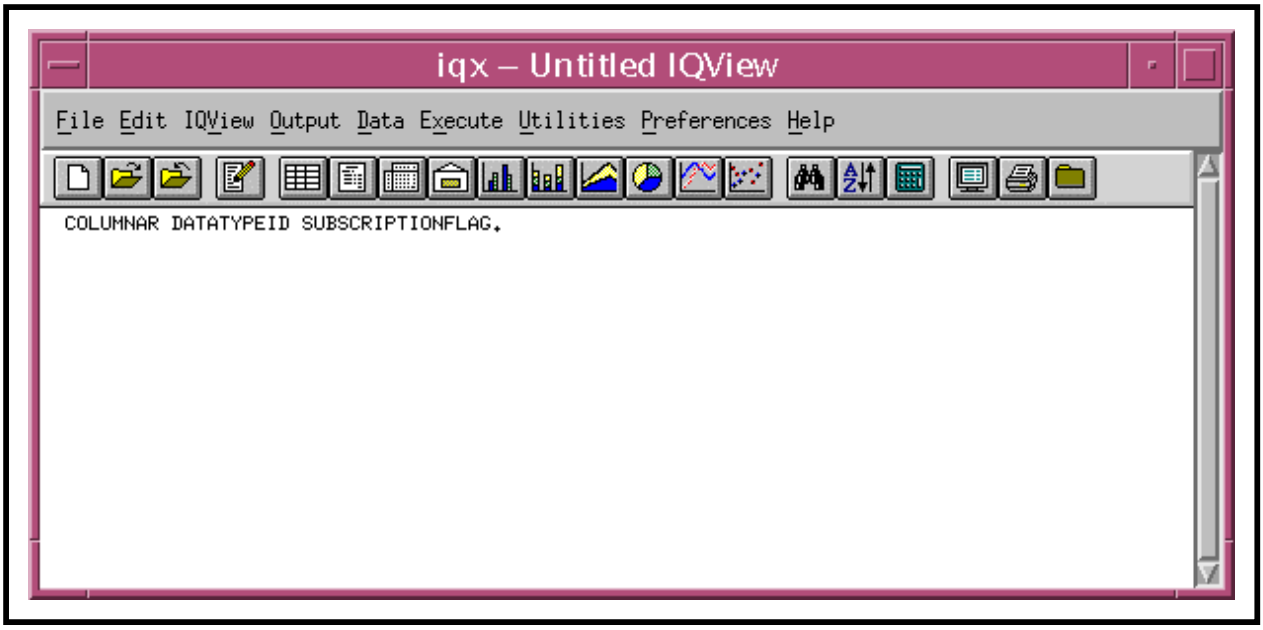


Figure 26.22.1-7. iqx IQView GUI

- The columnar selections are listed on the **iqx IQView** GUI as shown in Figure 26.22.1-7.

26 To generate a report make one of the following selections from the pull-down menu:

- **Execute → to Display** – to display the report on the terminal screen.
  - — The **iqx IQ Output** GUI (Figure 26.22.1-8) is displayed.
  - — Go to Step 35 after viewing the report.
- **Execute → to Printer** – to print the report.
  - — The **iqx Execute to Printer** GUI (Figure 26.22.1-9) is displayed.
  - — Go to Step 29.
- **Execute → to File** – to save the report in a file.
  - — The **iqx Execute to File** GUI (Figure 26.22.1-10) is displayed.
  - — Continue with Step 27.

27 Type a valid ***path/filename*** in the **Name:** field of the **iqx Execute to File** GUI (Figure 26.22.1-10).

- For example: **/home/cmshared/reportfile**
  - — Where **/home/cmshared/** represents the path and **reportfile** is the file name.

28 Click on the **OK** button.

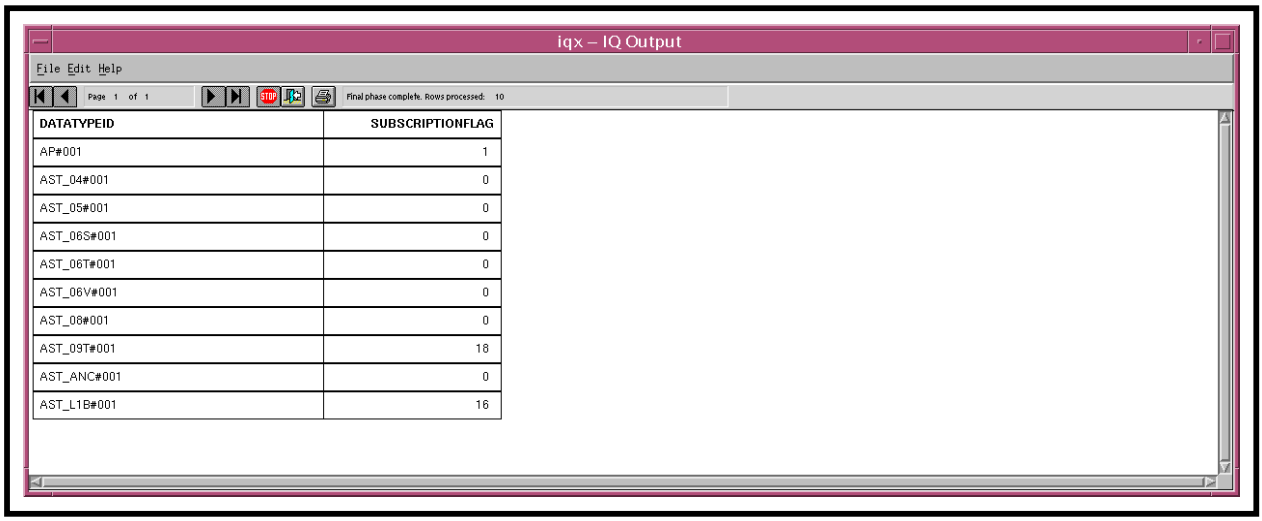


Figure 26.22.1-8. iqx IQ Output GUI

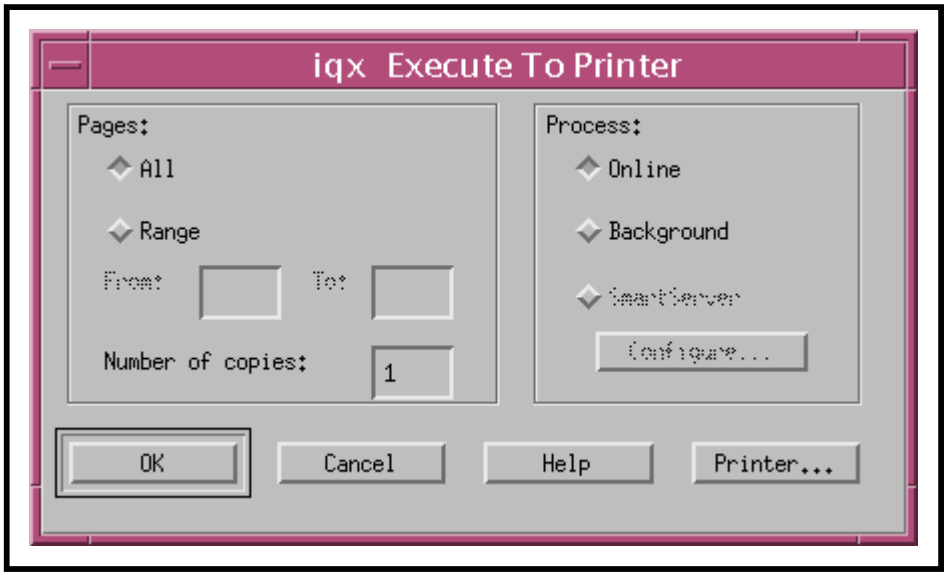


Figure 26.22.1-9. iqx Execute to Printer GUI

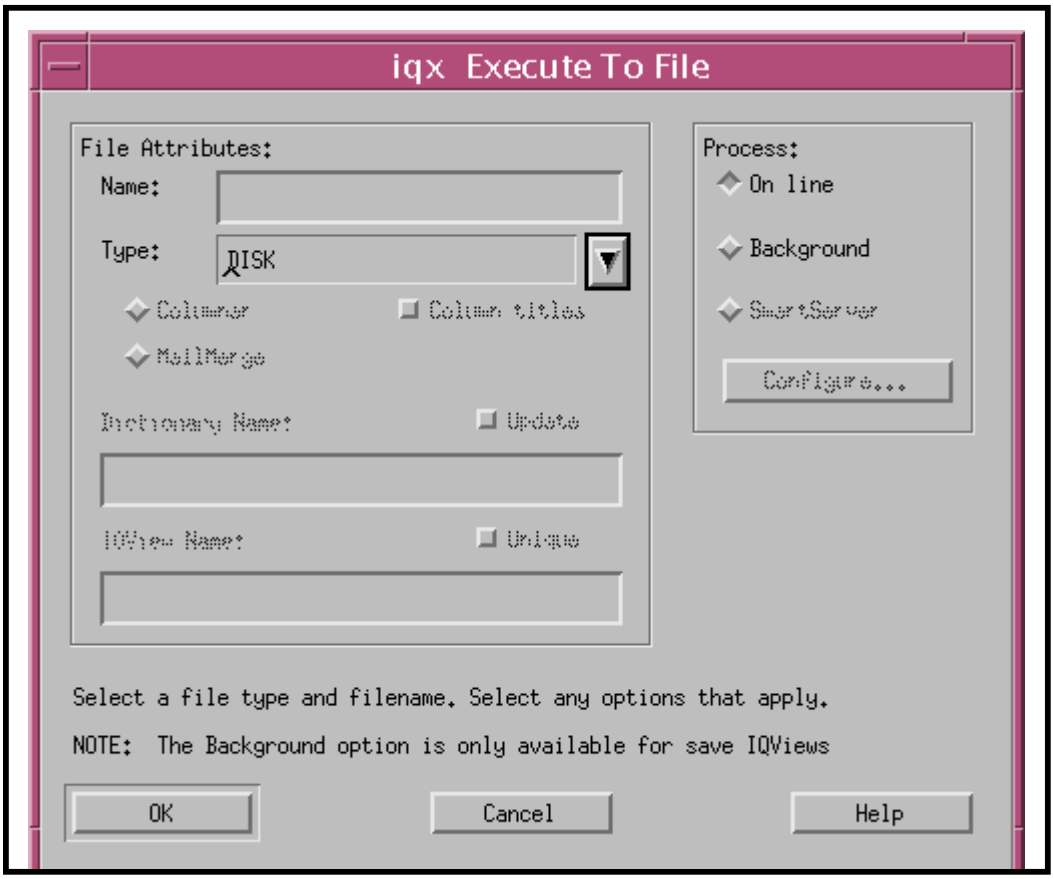


Figure 26.22.1-10. iqx Execute to File GUI

- Go to Step 35.

29 Click on the **Printer…** button on the **iqx Execute to Printer** GUI (Figure 26.22.1- 9).

- The **iqx Print Setup** GUI (Figure 26.22.1- 11) is displayed.

30 To list the available printers, first click on the option button associated with the **Printer** field.

- An option menu of printers is displayed.

31 Highlight the desired printer in the option menu.

- The desired printer is shown in the **Printer** field.
- For example: **Postscript printer one**.

32 If a report in landscape format is desired, click on the **Landscape** button.

33 Click on the **OK** button.

- The **iqx Print Setup** GUI (Figure 26.22.1- 11) is dismissed.

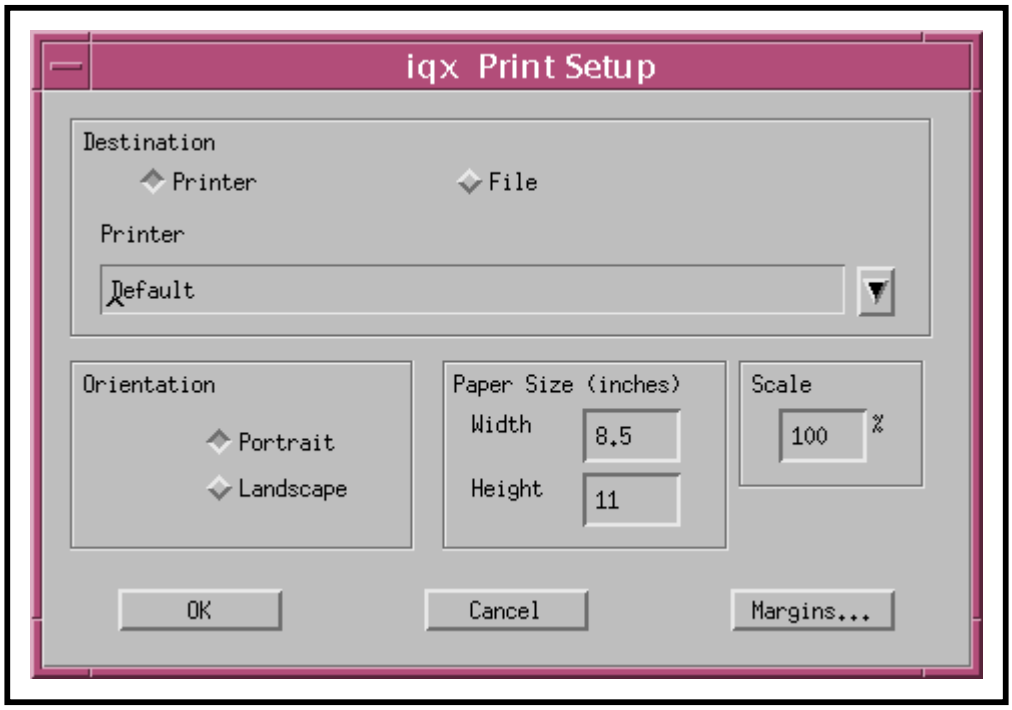


Figure 26.22.1-11. iqx Print Setup GUI

- The **iqx Execute to Printer** GUI (Figure 26.22.1-9) is displayed.

34 Click on the **OK** button.

35 To save the procedure/IQView, continue with Step 36; otherwise go to Step 43.

36 Select **File → Save Procedure As…** from the pull-down menu.

- The **iqx Save IQView** GUI (Figure 26.22.1- 12) is displayed.

37 Type a file name for the IQView in the name field.

38 Click on one of the following buttons if applicable:

- **Public**.
- **Public read only**.
- **Private**.

39 Click on the **OK** button.

- The **iqx Save Procedure** GUI (Figure 26.22.1-13) is displayed.

40 Type a file name for the procedure in the name field.

41 Click on one of the following buttons if applicable:

- **Public**.

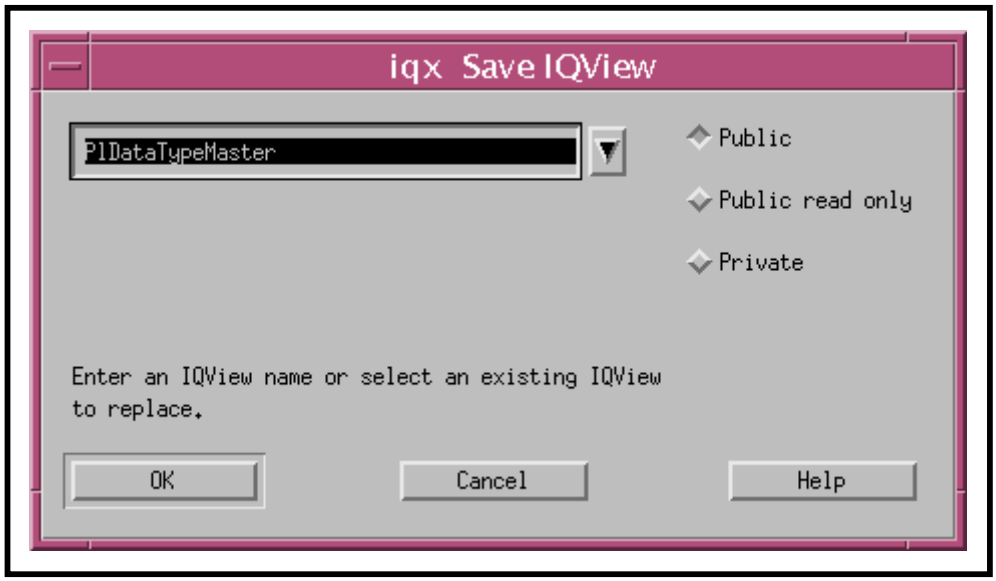


Figure 26.22.1-12. iqx Save IQView GUI

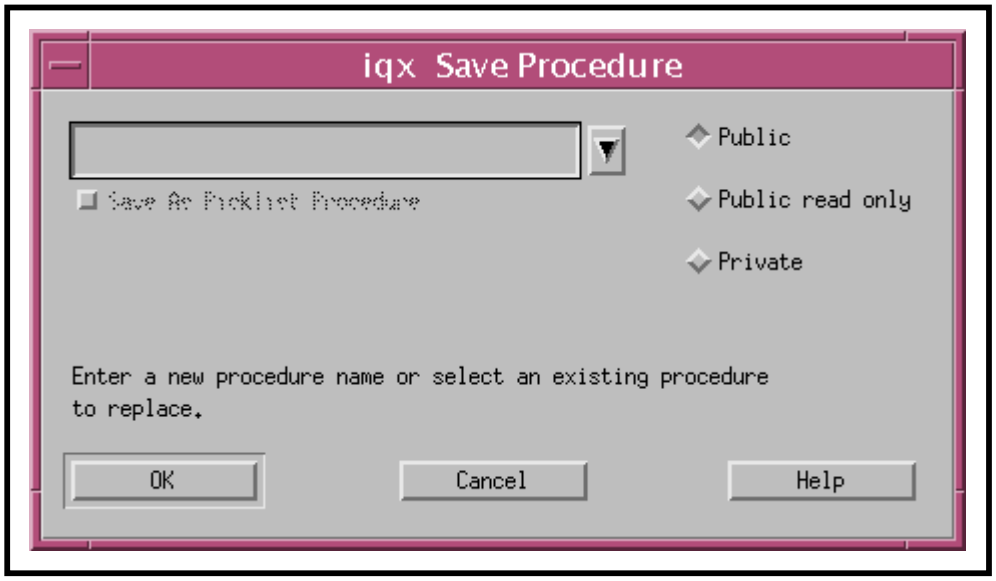


Figure 26.22.1-13. iqx Save Procedure GUI

- **Public read only**.
- **Private**.

42 Click on the **OK** button.

43 Select **File → Exit** from the pull-down menu to exit from the **iqx IQView** GUI (Figure 26.22.1- 1).

26.22.2 Formatting IQ Software Reports

[TBS]

26.23. Learn more about SSI & T

26.23.1 References:

PDPS Home Page: http://dmserver.gsfc.nasa.gov/ecsdev/relb/pdps/index.html

` URL for ECS Project Training Material Volume 16: SSI&T December 1997:

http://m0mss01.ecs.nasa.gov/smc/dc master.html

MISR Science Data Processing Software Test Plan, Volume 2, Detailed Procedures and Facilities Version 2.0, Part 1 (PGE 1) June 1998.

http://dmserver.gsfc.nasa.gov/relbit/relbit.html

SV DOC

REPOSITORY

Home

Drop Build Plans

Acceptance Test Plan

System Verifcation Test Plan

Access Database

Release B Testdata

Site Install and Checkout Test

End To End Test Procedures

Goddard DAAC M&O Status

VATC Status Page

ECS TEST

PAGES

Advertising Service (VATC)
User Registration Tool
(VATC)
V0 Web Client (VATC TS2)
V0 Web Client (DAAC
MODE TS2)
V0 Web Client (DAAC
MODE TS1)
V0 Web Client (DAAC
MODE OPS)

ECS TOOLS

EP7
RTM
CCR
EDHS
DDTS
Network Status Page
ECS Newsroom Server
Configuration Management
Release B Integration
ECS Telephone & Email Dir

ISO 9001

Business Manual Tab
Job Description Tab
Organization Charts Tab
Process Directives Tab
Training Tab

Miscellaneous Tab

Frequently Asked questions

EDHS

Raytheon Company

---

1. **ESDT Basics** **http://dmserver.gsfc.nasa.gov/esdt/EsdtSection1/index.html**
2. GDAAC Directory for SSI&T:**http://gsfcsrvr8.gsfcmo.ecs.nasa.gov/SSIT/**
3. The ESDT Process (updated for drop 4) by Karl W. Cox, 22 December 1997
4. MODIS - Science Data Processing Software Release 4 System Description
5. SDST-104, May 19, 1998
6. ECSINFO: **http://ecsinfo.hitc.com/iteams/Science/science.html**
7. PDPS howto are located on the EDF machines at: /home/PDPS/docs/
8. PDPS Web Page: **http//dmserver.gsfc,nasa.gov/ecsdev/relb/pdps/index.html**
9. For Troubleshooting or use the following EDF machines:
10. PDPS Troubleshooting Techniques are located on the EDF machines at : /home/PDPS/troubleshooting/
11. DPREP README files located at **:/usr/ecs/TS1/CUSTOM/data/DPS/**
12. DPREP binary located: **:/usr/ecs/TS1/CUSTOM/bin/DPS/**

26.23.2 Server Node Names Convention:

The naming convention is as follows:

Machine names are defined to be equivalent to the network hostname of the

machine. Network hostnames are limited to eight characters. On ECS we are now formatting these hostnames as svcimnni

Wheres : Site

- g – GSFC
- e – EDC
- l – LaRC
- n – NSIDC
- a – ASF
- j – JPL
- p- PVC
- t – VATC

v : Version

- 0 -- Release 4 At-Launch COTS design
- 1 -- Stood for B.1 COTS design; OBE, but still used in VATC
- s -- Used for special SSI&T machines set up at GSFC and EDC

ci : Hardware Configuration Item

- sp -- Science Processing (SPRHW)
- ai -- Algorithm Integration and Test (AITHW)
- aq -- Algorithm Quality Assurance (AQAHW)
- pl -- Planning (PLNHW)
- ms -- Management Subsystem (MSSHW)
- cs -- Communications Subsystem (CSSHW)
- in -- Interface (INTHW)
- dm -- Data Management (DMGHW)
- dr -- Data Repository (DRPHW)
- ac -- Access Control Management (ACMHW)
- ic -- Ingest Client (ICLHW)
- wk -- Working Storage (WKSHW)
- di -- Distribution (DIPHW)
- as -- ASTER (ASTHW) [Occurs only at EDC]
- te -- Test Equipment

m : Manufacturer

- s -- Sun
- g -- SGI
- h -- HP
- x -- X Terminal
- p -- PC

nn : One-up number (01, 02, et cetera -- should be unique for the CI)

i : Interface type

- <null> -- Production network
- u -- User network
- h -- HiPPI

Note that the machine name leaves off the last letter (the interface); hence, we generally refer to machines as "g0spg01", vice "g0spg01h". A machine may have multiple interfaces -- production, user, and HiPPI. So a single machine may show up in network documentation multiple times (g0spg01, g0spg01h, g0spg01u).

26.23.3 A Handy Alias file to use while conducting SSI&T:

p0spg01{emcleod}51: **alias**
+ pushd
- popd
More more !* |grep -v "Msg: Caught dce error: No more bindings (dce / rpc)" |grep -v "MsgLink :0 meaningfulname :EcAgManager::Recovery" |grep -v "MsgLink :0 meaningfulname :DsShSRequestRealSetStateSettingState" |grep -v "Command 1/1 execution complete"
cdstagebin cd /ecs/formal/STAGE/DSS/bin/sun5.5
dbg debugger -bg NavajoWhite -fn 12x24 !* &
dbs /home/jzhuang/bin/dbrowser-syb &
disp setenv DISPLAY !*
mgr DpAtMgr ConfigFile /usr/ecs/TS1/CUSTOM/cfg/DpAtMG.CFG ecs_mode TS1&
ops cd /usr/ecs/OPS/CUSTOM
ts1 cd /usr/ecs/TS1/CUSTOM
xslq_autosys isql -Uautosys -Pautosys -Sp0sps06_srvr
xsql_css isql -Ucss_role -Pwelcome -Sp0ins01_srvr
xsql_dss isql -UsdsrvApp -Pwelcome -Sp0acg01_sqs222_srvr
xsql_ios isql -Uios_role -Pwelcome -Sp0ins02_srvr
xsql_pdps isql -UpdpsUsers -Pwelcome -Sodyssey_srvr

 # alias for browser
Note: On Performance Verification Center (PVC), dbbrowser has to originate from workstation ODYSSEY to execute alias db_pdps to reach PDPS DB on p0pls02.
alias db_dss '/home/opscm/dbr/dbrowser-syb -UsdsrvApp -Pwelcome -Sp0acg01_sqs222_srvr &'
alias db_ios '/home/opscm/dbr/dbrowser-syb -Uios_role -Pwelcome -Sp0ins02_srvr &'
alias db_css '/home/opscm/dbr/dbrowser-syb -Ucss_role -Pwelcome -Sp0ins01_srvr &'
alias db_autosys '/home/opscm/dbr/dbrowser-syb -Uautosys -Pautosys -Sp0sps06_srvr &'
alias db_pdps '/home/opscm/dbr/dbrowser-syb -UpdpsUsers -Pwelcome -Sp0pls02_srvr &'
alias db_ing '/home/opscm/dbr/dbrowser-syb -UEcInPolling -P3nWK0fG1 -Sp0icg01_srvr &'
alias db_stmgt '/home/opscm/dbr/dbrowser-syb -UEcDsStFtpDisServer -PS71Oq4y3 -Sp0icg01_srvr &'

ls -laF look at root and .cshrc, .alias

26.23. 4 HOWTO_SSIT HELPFUL NOTES

Xterm format to bring up xterm windows for servers all at once

This list of xterm identifiers should be assigned a file name and placed into your home directory. Then invoke the file name when you want to create the entire list of xterms.

Example of filename: - xterm_pls

xterm -sb -sl 10000 -fg green -bg black -name "Resource Editor" &

```
xterm -sb -sl 10000 -fg green -bg black -name "Resource Model" &
xterm -sb -sl 10000 -fg green -bg black -name "Production Request Editor" &
xterm -sb -sl 10000 -fg green -bg black -name "Planning Workbench" &
xterm -sb -sl 10000 -fg green -bg black -name "Database" &
xterm -sb -sl 10000 -fg green -bg black -name "Planning Timeline" &
xterm -sb -sl 10000 -fg green -bg black -name "Logs" &
xterm -sb -sl 10000 -fg green -bg black -name "ECS Assist" &
xterm -sb -sl 10000 -fg green -bg black -name "g0sps06" &
xterm -sb -sl 10000 -fg green -bg black -name "g0ais01" &
xterm -sb -sl 10000 -fg green -bg black -name "g0spg01" &
xterm -sb -sl 10000 -fg green -bg black -name "g0drg01" &
xterm -sb -sl 10000 -fg green -bg black -name "g0pls02" &
# setenv ECS_HOME /usr/ecs
# setenv MODE TS2
```

26.23.5 howto_setup_orbits_and_pathmaps

ORBITS & PATHMAPS
The Path is a an orbit swath, defined for Landsat-7 ("WRS"),
which MISR uses in its processing.
Because the earth rotates under it, the path the satellite
traverses for its next orbit is not path 2, but path 17.
This mapping of orbit number to path number is found in the
PATHMAP ODL file.
In that file ABSOLUTE_PATH is what we call orbit number here,
and MAPPED_PATH is the path number.
Now, this mapping is fixed, and never changes.
What does change is the time each orbit starts.
This is because the orbit may drift and be subject to
maneuvers.
Periodically (say, every 2 weeks), the Flight Dynamics
Facility (FDF) at GSFC issues a new Orbit Start Time,
with corresponding Orbit Number.
When this happens, the PDPS ORBIT ODL must be updated,
with a new ORBIT_MODEL object, containing the new
ORBIT_START and corresponding ORBIT_NUMBER.
The new ORBIT_PATH_NUMBER is determined manually by
the operator, using the lookup table in the PATHMAP ODL file.
1. Example (MISR PGE7 test data)
FDF issues a bulletin stating that imaginary platform MPGE7
orbit number 27 starts at 14:37:39Z 07-Jan-96.
SSIT operator receives the bulletin, looks up in
the corresponding PATHMAP_WRS7.odl file to find that
ABSOLUTE_PATH XX corresponds to MAPPED_PATH XX.
NEED TO FIX THIS
The SSIT operator then creates a new ORBIT_MODEL object in the
ORBIT ODL file as follows:

```
OBJECT = ORBIT_MODEL
    CLASS = 2
    ORBIT_NUMBER = 27
    ORBIT_PERIOD = "SECS=5932"
    ORBIT_START = "01/07/1996 14:37:39Z"
    ORBIT_PATH_NUMBER = 90
END_OBJECT = ORBIT_MODEL
*******************************************************************************
```

26.23.6 Technical Notice concerning Leap Second/DPREP

Notice of Change to Toolkit Ephemeris and
Attitude Interpolation

March 15, 1999

Effective with a late patch to Drop 4PY, the time interval over which the Toolkit will interpolate spacecraft ephemeris or attitude is reduced from 121 seconds to 60 seconds. (The normal interval between packets for AM1 is 1.024 seconds, except for FDD replacement ephemeris, which will use 1 second time intervals.)

Impact of this Change

For reasons explained below, the only impact on real data processing would be to avoid the possibility of a less-than-ideal interpolation.

The impact on I&T would be that test data should always be generated or obtained with packet interval 60 seconds or less. There are a few inputs for the Toolkit test drivers, which are not a deliverable, but are occasionally supplied to Toolkit users, which (to save file space) are set up to work with 120 second packets. These lie in the file "orbsim.in" (which is, as explained, a PDPS add-on which is unsupported software when it is provided to users). To conduct tests without new and annoying failures, if you use this input file for "orbsim", you should edit it to replace "120" by "60" or less, globally. Results will not change significantly from the expected test results.

Rationale for this Change

The SDP Toolkit uses an analytic, cubic polynomial method to interpolate ephemeris that is extremely fast, but which could generate undesirable and spurious variations if

applied across too large a time interval, or to data that are not smooth.

DPREP, which will process all incoming ephemeris data, has a robust method for patching gaps up to 60 seconds, based on a least squares quartic fit, component by component, to position and velocity. Although this method is undesirably slow for repeated use within the Toolkit, it is ideal for DPREP, which runs only once on a data set. The existence of any gaps not repairable by DPREP (i.e. > 60 seconds) will trigger procedures to obtain replacement data files from FDD.

It is therefore unwise to allow the Toolkit to interpolate gaps which DPREP cannot fill, so it is being changed. The same limit is set for attitude. We expect no gaps whatever in FDD attitude, and the attitude is not of much use without the ephemeris, so this change was made consistently.

26.24 Landsat 7 Error Handling Tool

The Landsat 7 Error Handling Tool provides the ECS Operations Staff with the ability to Merge/Demerge/Promote/Delete Landsat 7 granules using a command line interface. This tool works exclusively in the Science Data Server (SDSRV) database. The tool only modifies tables in the SDSRV database.

The **Delete** command gives the user options to modify the DeleteFromArchive flag in the DsMdGranules table only or physically delete the granules from the archive and the inventory.

The **Merge** command mimics the process done by the SDSRV and Landsat 7 Dynamic Link Library (DLL) during an ingest of Landsat 7 data.

The **Demerge** operation allows the operations staff to separate incomplete combined granules and thus allows the recombination, Merging, of complete data sets.

The **Promote** tool is used to associate granules with only a single format of data, bands 1-6 format 1/bands 6-8 format 2, with the appropriately combined subinterval and thus making the granule available for ordering. Before this tool, the data was unavailable.

26.24.1 Quick Start Using the Landsat 7 Error Handling Tool

Entering the following command starts the Landsat 7 Error Handling Tool:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

There are three command line parameters and they are used in combination with each other. Table 26.24.1-1 describes these parameters.

Table 26.24.1-1. Command Line Parameters of the Landsat 7 Error Handling Tool

Parameter Name	Description
mode	Mode corresponding with the database to be modified
Start Time	Start time, insertTime, of the temporal range of granules to search. This will be used to search for unmerged subintervals/scenes when the Landsat 7 Error Handling Tool is started.
Stop Time	Stop time, insertTime, of the temporal range of granules to search. This will be used to search for unmerged subintervals/scenes when the Landsat 7 Error Handling Tool is started.

All of the parameters are mandatory. Before starting the tool the file "EcDsSrDbL7ErrorHandlingRC" should be modified to reflect the user's local environment.

This file is used by EcDsSrDbL7ErrorHandling to setup environment # variables, which makes EcDsSrDbL7ErrorHandling more tunable. # MUST be updated to customize at sites.#

```
export SYBASE=/tools/sybOCv11.1.1   # Directory where Sybase stuff resides
export SERVER=x0acgxx_srvr                # SQL server where ECS Science Data
Server
                                          # Database can be accessed
```

export ECS_HOME=/usr/ecs/

export SQSSERVER=x0acgxx_sqs322_srvr # SQS server

export DBUSERNAME=EcDsScienceDataServer # Valid user name to login into

database

```
export DBPASSWD=xxxxxxx                   # Password to access database

export DBNAME=EcDsScienceDataServerX      # Name of database

export WORKDIR=/usr/ecs/${MODE}/CUSTOM/data/DSS   # Directory where the
                                                  # script resides
```

export reportdir=/usr/ecs/${MODE}/CUSTOM/data/DSS # Directory for report files

```
export tempdir=/usr/ecs/${MODE}/CUSTOM/data/DSS   # Directory for
                                                  # temporary files
```

export errorfile=/usr/ecs/${MODE}/CUSTOM/data/DSS/EcDsSrDbL7ErrorHandling.errlog

File to use for holding any possible error messages

26.24.2 Landsat 7 Error Handling Tool Commands

The Landsat 7 Error Handling Tool provides the following granule modification options:

1. **Initiate Merge of Landsat 7 Subintervals/Scenes from the SDSRV database**. The selected Subintervals/Scenes must be passed to the tool via a file, which gets created

during the start of the Landsat 7 Error Handling tool. The format of the input file is very specific, it is listed below.

Subinterval sample file input:

```
dbID    ShortName Insert Time         Path    Starting Row Ending Row       Assoc. File Name
10248   L70RF1    May 27 1998 9:26AM 172     44           50              SC:L70RF1.001:10248
10247   L70RF2    May 27 1998 9:26AM 172     44           50              SC:L70RF2.001:10247
10669   L70RF1    May 27 1999 9:25AM 172     44           50              SC:L70RF1.002:10669
10661   L70RF2    May 27 1999 9:25AM 172     44           50              SC:L70RF2.002:10661
```

Scene sample file input:

```
dbID    ShortName      Insert Time           Path    Row     Associated File Name
13530   L70RWRS1       May 27 1998 9:25AM    172     44      SC:L70RWRS1.001:13530
13531   L70RWRS2       May 27 1998 9:25AM    172     44      SC:L70RWRS2.001:13531
13532   L70RWRS1       May 27 1998 9:26AM    172     45      SC:L70RWRS1.001:13532
13533   L70RWRS2       May 27 1998 9:26AM    172     45      SC:L70RWRS2.001:13533
```

2. **Promote Landsat 7 Subinterval/Scene from the SDSRV database.** The selected Subintervals/Scenes must be passed to the tool via a file, which gets created when the operators starts the Landsat 7 Error Handling tool.
3. **Demerge Landsat 7 Subinterval/Scene from the SDSRV database.** The selected Subinterval/Scene must be passed to the tool by entering the geoid of the granule when prompted.
4. **Delete Landsat 7 Subinterval/Scene from the SDSRV database**. The selected Subintervals/Scenes must be passed to the tool by entering the geoid of the granule when prompted.
5. **Generate list of Orphaned Landsat 7 Subintervals/Scenes in the SDSRV Database.** The list of orphaned Subintervals/Scenes will be generated based on the start time and stop time parameters passed in as parameter #2 and #3 at invocation of the tool.

26.24.2.1 Initiate Merge of Landsat 7 Subintervals/Scenes from the SDSRV Database

This command has the form:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

This command invokes the tool. The script will display a menu of commands. To initiate a merge, the user should select option 3, “Merge Subintervals/Scenes”. The user will be prompted for a filename. The user should enter the name of the file that contains the format as described in section 26.24.2. The script will then return a failed or successful status and then return to the menu.

26.24.2.2 Promote Landsat 7 Subinterval/Scene from the SDSRV database

This command has the form:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

This command allows the user to make data available that cannot be used. This command is used to make data with only one format appears as though both formats existed. The user enters the command and the script will display a list of options. The user should select option 5,"Promote Orphaned granules". The user will be prompted for a filename that contains the format as described in section 26.24.2. After entering the filename, the script will return a failed or successful status and then return to the menu.

26.24.2.3 Demerge Landsat 7 Subinterval/Scene from the SDSRV database

This command has the form:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

This command is used for incorrectly combined data sets. The user should select option 4, "Demerge L70RF1/F2 granules", from the menu. The script will prompt the user for the geoid of the granule to demerge. An example of a geoid is SC:L70R.001:12345. The first part is the type of the granule. SC represents science granules. The second part is the subtype and version of the granule. The last part is the dbId of the granule. This uniquely identifies the granule in the Science Data Server's database.

26.24.2.4 Delete Landsat 7 Subinterval/Scene from the SDSRV database

This command has the form:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

After entering the above command, the user will see a list of options. The user will should select option 6,"Delete Unmerged granules". The script prompts the user for the geoid of the granule to delete. The user will be prompted for confirmation of the deletion. The user also will be prompted to determine if the granule should be deleted from the archive and the inventory.

26.24.2.5 Generate list of Orphaned Landsat 7 Subintervals/Scenes in the SDSRV database

This command has the form:

EcDsSrDbL7ErrorHandling <mode> <Start Time> <Stop Time>

The first time the script is invoked it always searches the SDSRV database for orphaned granules within the time range given by the start time and stop time input parameters. After that, the user can generate the list by selecting command #2, "Update files on /tmp directory". This will create two files in the /usr/ecs/<MODE>/CUSTOM/data/DSS directory. One of the files contains all the unmerged subintervals, the file name is unmergedsubintervals. The other file contains all the unmerged scenes, the file name is unmergedscenes. The user may use these files for the merged Landsat 7 granules and promoted Landsat 7 granule options.

26.24.3 Required Operating Environment

For information on the operating environment, tunable parameters, and environment variables refer to the 920-TDx-013 "Custom Code Configuration Parameters" documentation series. The "x" refers to the installed location, e.g. 920-TDG-013 is for GSFC DAAC.

26.24.3.1 Interfaces and Data Types

Table 26.24-2 lists the supporting products that this tool depends upon in order to function properly.

Table 26.24-2. Interface Protocols

Product Dependency	Protocols Used	Comments
SDSRV Database	SQL	via SQL server machine

26.24.4 Databases

The Landsat 7 Error Handling tool does not include the direct managing of any database. It has an interface with the Science Data Server Database: however this interface is based on a simple parameter passing function. For further information of the Science Data Server Database refer to 311-CD-107-005, *Science Data Server Database Design and Schema Specifications for the ECS Project*.

26.24.4.1 Special Constraints

The Landsat 7 Error Handling Tool doesn't require any servers to be running. It is strongly recommended that as little as possible should be going on in the SDSRV database while the tool is being used.

26.24.4.2 Outputs

None.

26.24.4.3 Event and Error Messages

None

26.24.4.4 Reports

None

26.25 Deleting Granules

As of Release 6A, the system provides a **Granule Deletion** tool, complementing the automatic, scheduled deletion capability that permits operators to delete products produced and archived by

the Planning and Data Processing subsystems on a scheduled basis (e.g., deletion at a certain time (configurable by the operator) after product creation.

The **Granule Deletion** tool allows operators to delete products on demand. There are a variety of circumstances that may require deletion on demand, such as:

New PGE versions have been created and are used to reprocess large amounts of past data, creating new ESDT versions. As reprocessing progresses, operations deletes the granules for the old ESDT versions from the archive and inventory.

It is determined that certain lower-level (e.g., Level 2) products are of little or no interest to the science or public user community. In concert with the science teams, DAAC operations personnel decide to remove these products from the inventory. Since the products are still referenced by higher-level products as inputs, the DAAC decides to keep the inventory records for production history purposes.

One or more granules were found defective and were reprocessed on an individual basis. When the reprocessing is complete, the operator wishes to delete the old, defective granule(s) from the inventory.

A DAAC has extended ECS with subsetting services. The subsetted products are produced outside ECS, but are then inserted into the ECS archive to take advantage of the ECS distribution capability. The DAAC writes a script to delete the subsetted products on a regular basis.

26.25.1 Deletion Capability and Features

The Science Data Server has provided an application programming interface (API) for deleting granules from the archive, or from both the archive and inventory since earlier releases, but the Granule Deletion tool adds a front-end command-line utility that provides several ways for selecting granules for deletion. Confirmation is generally required so that granules are not inadvertently deleted. However, the confirmation may be suppressed so that operators can run regularly scheduled deletion scripts using background execution. This suppression possibility presents an opportunity for inadvertent loss of data and so must be used with care and only after thorough testing of any deletion script.

The Science Data Server captures deletions and related errors in the application log. Operators may also specify a separate and independent delete log for immediate analysis of the success or failure of a delete operation.

26.25.1.1 Deletion Sequence

The deletion of granules from the archive involves three elements, and therefore actually occurs in stages. Two of the elements are scripts that address the Science Data Server (SDSRV), and the third is a part of the Storage Management (STMGT) Graphical User Interface (GUI).

For the first stage, a delete script applies deletion checks to the selected granules, "logically" deleting from the inventory those granules that satisfy the checks. These granules are flagged as 'deleted' and can no longer be accessed, but their inventory entries are not yet removed. The deletion flag consists of a time stamp recording the logical deletion time.

The second stage is actual deletion from the inventory, which occurs when the operations staff runs the physical deletion script. The script removes all inventory rows for granules that were flagged as 'deleted,' and produces the list of the granule files that are now eligible for deletion from the archive. That list is transferred to the STMGT

database. The operations staff controls the lag time between logical deletion and physical deletion. That lag time is entered into the physical deletion script, which deletes only inventory entries for granules that have been logically deleted prior to that time period.

STMGT provides a GUI screen, as illustrated in Figure 26.25.1, that allows the operator to initiate the removal from the archive of the files listed its deletion table (populated by SDSRV). STMGT creates requests to the archive servers to delete files. The STMGT GUI can be used to look at the state of the deletion requests. Files that are successfully deleted have their associated rows removed from the STMGT database table.

Periodically, as sufficient data removal from the archive makes it appropriate, operations may elect to reclaim the tape space and recycle archive tapes. The AMASS software commands (*volcomp*, *volclean*, *volformat*, *volstat*) are used for that purpose.

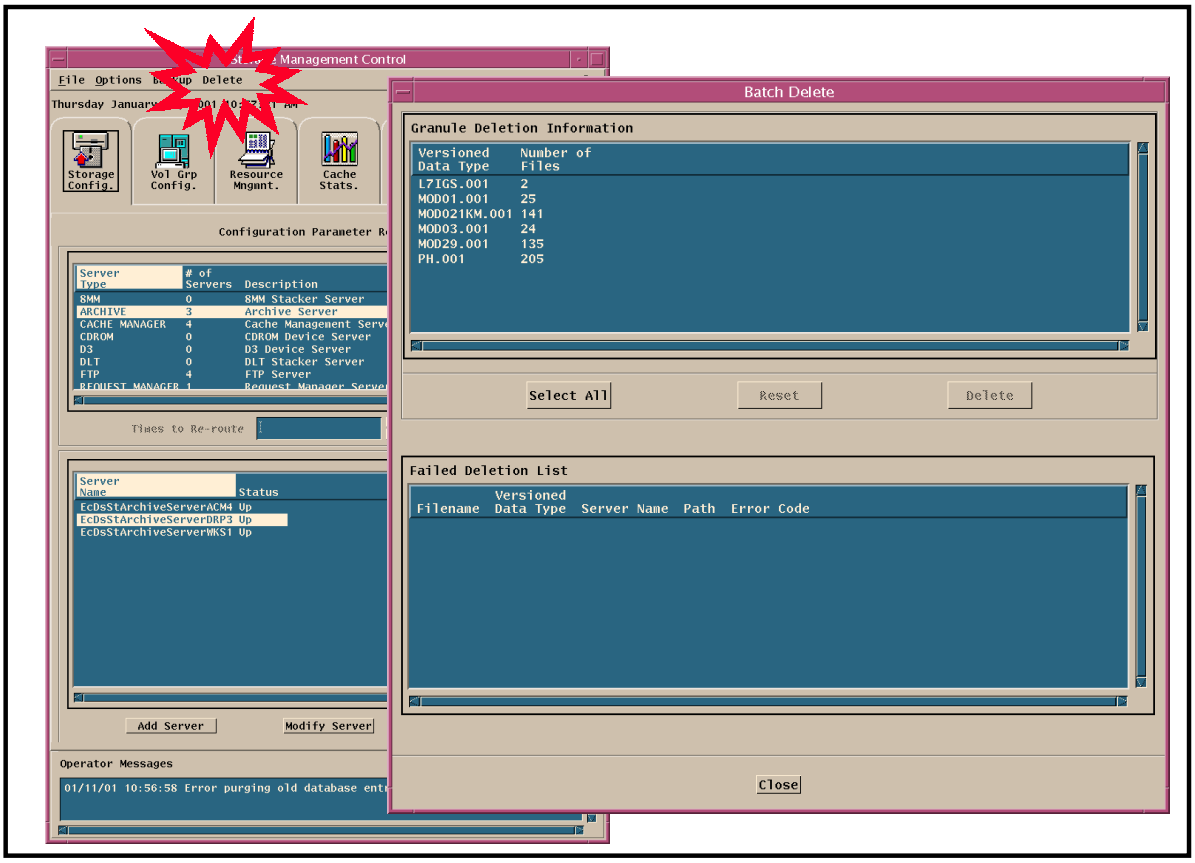


Figure 26.25.1. STMGT GUI screen for granule deletion from archive

26.25.2 Granule Deletion

This process of Granuale Deletion is for operators to delete granules from the inventory/archive on demand. Deletion of Granules is a three-part process.

- **Granule Deletion Client**
- **Deletion Cleanup Utility**
- **STMGT GUI**

1. First, a command line SDSRV utility, called the **Granule Deletion Client**, is available for selecting granules to be marked for deletion from the archive (DFA), with an option to also mark for delete from the inventory (PHYSICAL DELETE). A confirmation is generally available, but can be suppressed to allow for background execution. The Science Data Server logs will be inspected for messages related to deletions, and a separate log is available for immediate inspection of only deletion messages.
2. The second, a SDSRV script, called the **Deletion Cleanup Utility**, is run to send those granules that have been marked for deletion to the STMGT database so they can be physically deleted from the archive. In addition, if granules were marked for Physical Delete in SDSRV, they will also be deleted from the SDSRV database. A lag time is used so the operator can request that not all the granules marked for deletion are immediately deleted. Granules will be deleted when the Deletion Cleanup Utility is executed when they have been marked for deletion before <today - lagtime>. This lag time can be overridden by entering a lag time of 0, in which case all granules marked for deletion will be immediately deleted.
3. The third and last part of the process is the **STMGT GUI** can then be used to complete deletion of data granules from the archive (AMASS). Granules are selected for deletion from the STMGT GUI via the datatype name (ESDT ShortName) and Version ID.
4. Using the Granule Deletion Client, granules can be selected for deletion several different ways:
   - ESDT ShortName, Version and granule time coverage
   - ESDT ShortName, Version and granule insert time range
   - Separate Input file containing SDSRV Granule IDs
   - Separate Input file containing ShortName, Version, and Local Granule ID (Logical Granule ID noted in the ticket is referred to in this test as Local Granule ID, which is the name of the parameter as it resides in the database)

5. The number of granules returned can be verified by making the same query in the database using SQL commands. Granules can be deleted from both the inventory and the archive, from the archive only, in the foreground, and via background (cron) jobs. It can be verified that all occurrences of a file in the archive are deleted when requested, such as from a primary and a backup archive location. It can be verified that Browse, QA and PH granules associated with physically deleted granules are also deleted if not referenced by any other granules.

Error conditions include:
- Attempts to delete granules that are still being referenced by other granules
- Unauthorized users are indeed halted from deleting data
- Recovery from Archive Server, Science Data Server and Science Data Server DBMS server faults occurring during deletion requests

6. Using a user ID authorized for granule deletions, the operator has a number of interfaces to consider for the deletion of granules, including the mechanisms for selecting the granules to be deleted and the confirmation of the deletion.
   a. Granules can be selected by ESDT short name, ESDT version, and granule time coverage
   b. Granules can be selected by ESDT short name, ESDT version, and granule insert time range
   c. Granules can be specified in a separate input file containing either SDRV Granule Ids or Logical Granule Ids

   d. The input file can list granules belonging to collections belonging to different logical volume groups.
   e. The operator can optionally list the geoID and logical granule ID of each of the granules selected for deletion.
   f. The number of granules selected for deletion is displayed to the operator and the operator is asked to confirm the deletion
   g.The operator can suppress the confirmation prompt via a command line argument
   h.Granules are tagged for deletion from inventory and archive or from archive only, depending on operator choice
   i.By default, BROWSE, QA, and PH granules associated with physically deleted granules are deleted if no longer referenced otherwise
   j.The operator can suppress deletion of BROWSE, QA and PH granules
   k.The files associated with the deleted granule are deleted from all archive locations (including back-up locations)
   l.The deletions are logged as required to the SDSRV application log file and the operator specified granule deletion log file
   m.Granule deletions can be performed during normal SDSRV processing, i.e., while other requests such as insert, acquire, and searching are in progress

26.25.2.1 Granule Deletion Client Steps

Workspaces are created and each process is carried out for Granule Deletion

1. Create the following workspaces on the CDE window manager (xterms created below):

- SDSRV 5 xterms
- STMGT 7 xterms
- GUIs 4 xterms
- Database 2 xterms

2. In each workspace, open up the number of xterm windows as is indicated for each workspace above:

- Invoke a terminal window if necessary
- Log on to the appropriate machine for each workspace and create xterms for each workspace (see following commands as an example)
- **setenv DISPLAY <terminalId>:0.0**
- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- Invoke an xterm: **xterm -sb -sl 5000 –fg white –bg blue -n <workspace.1> &**

- **Invoke another xterm: xterm -sb -sl 5000 –fg white –bg blue -n <workspace.2>&**
- **Invoke database xterms as follows: xterm –sb –sl 5000 –fg black –bg green –n nnn_db&**

3.Verify that all Servers are up.

4. The user is logged into the SDSRV and STMGT databases

On the **Database** workspace, log into the SDSRV and STMGT databases from the two xterms and from the appropriate hosts:

- **rlogin t1acs03**
- **isql –U sdsrv_role –St1acg04_srvr**
- **<password>**
- **use EcDsScienceDataServer1_<MODE>**
- **go**
- **rlogin t1dps01**
- **isql –U stmgt_role –St1acg04_srvr**
- **<password>**
- **use stmgtdb1_<MODE>**
- **go**

5. The Distribution and Storage Management GUIs are displayed on the screen

On the **GUIs** workspace, invoke the **DDIST and STMGT GUIs**:

- **rlogin t1dps01**
- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **setenv DISPLAY <TerminalId:0.0>**
- **ls *Gui***
- **EcDsDdistGuiStart <MODE>**
- **EcDsStStmgtGuiStart <MODE>**

6. The following step may not be necessary. Check with your DB Manager.

Show Detailed Steps to Reset the Lock on the DsMdDeletedGranulesTable, when necessary**.**

In the **SDSRV** workspace,

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **ll *Lock***
- **EcDsResetLock.pl**
- Enter Mode of Operation : **<MODE>**
- Enter Log File name : **ResetLock.log**
- Enter Sybase User Name : **sdsrv_role**
- Enter Sybase Password : **<password>**
- Enter Sybase SQL Server Name : **t1acg04_srvr**

Enter SDSRVs database name : **EcDsScienceDataServer1_<MODE>**

The above steps will be performed over and over again, as directed in subsequent steps of this procedure

7. Perform a query in the **Database** workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and insertTime > "<insertbegdate/time>" and insertTime < <insenddate/time>"**
- **go**

A list of dbIDs is returned. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes using the same search criteria.

8. On the **SDSRV** workspace, execute the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule insert time coverage:

EcDsGranuleDelete –name <ShortName> -version <version no.> -insertbegin <insbegdate/time> –insertend <insenddate/time> –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel3.log -display ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

The script runs to completion. A list of geoIDs and local granule IDs is returned. Check to see if this list compares favorably with the isql query previously performed

9. View the Deletion log file and the tail of the SDSRV ALOG file:

- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule insert time of the request. These messages also demonstrate that the display option was referenced so the granules to be deleted are listed (geoidID and local granule ID) but not deleted.

A delete with ESDT Name/Version and by Granule Insert Time and specify Data From Archive (DFA), Granules are tagged for deletion

10. On the **SDSRV** workspace, run the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule insert time coverage:

EcDsGranuleDelete –name <ShortName> -version <version no.> -insertbegin <insbegdate/time> –insertend <insenddate/time> –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel7.log –DFA ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

The script is executed, and displays the number of granules for deletion, and prompts the user as to whether to continue.

a. When prompted, type: **y,** The script continues to completion.

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule insert time of the request, and a successful completion of the deletion.

11. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel7.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

12. On the **Database** workspace, view the SDSRV database to verify the granules were tagged for deletion:

- **Select * from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and insertTime < "<insenddatetime>" and insertTime > "<insbegdate/time>"**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 1.

26.25.2.2 Delete with ESDT Name/Version and by Granule Insert Time & specify physical delete

Granules are tagged for deletion.

1. On the **SDSRV** workspace, run the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule insert time coverage:

- **EcDsGranuleDelete –name <ShortName> -version <version no.> -insertbegin <insbegdate/time> –insertend <insenddate/time> –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel8.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

  a. y (when prompted)

The script displays the number of granules for deletion, and prompts the user as to whether to continue. The script completes successfully.

2. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel8.log**

- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule insert time of the request, and that the deletion completed successfully.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **Select * from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and insertTime < "<insenddate/time>" and insertTime > "<insbegdate/time>"**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 1.

26.25.2.3 Delete with ESDT Name/Version and by Granule Time and specify a physical delete

Granules are tagged for deletion

1. Perform a query in the **Database** workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and BeginningDateTime > "<granbegdate/time>" and EndingDateTime < "<granenddate/time>"**
- **go**

A list of dbIDs is returned. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes using the same search criteria.

This script runs to completion and the user is prompted with the number of granules to be deleted before confirming the delete.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule time coverage:

- **EcDsGranuleDelete –name <ShortName> -version <version no.> -BeginDate <granbegdate/time> –EndDate <granenddate/time> –log**

/usr/ecs/<MODE>/CUSTOM/logs/GranDel9.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

a. y (when prompted)

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel9.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and BeginningDateTime > "<granbegdate/time>" and EndingDateTime < "<granenddate/time>"**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and a DFA flag of 0.

26.25.2.4 Delete with ESDT Name/Version and by Granule Time & specify a DFA

Granules are taged for deletion

1. Perform a query in the **Database** workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and BeginningDateTime > "<granbegdate/time>" and EndingDateTime < "<granenddate/time>"**
- **go**

A list of dbIDs is returned. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes using the same search criteria.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule time coverage:

- **EcDsGranuleDelete –name <ShortName> -version <version no.> -BeginDate <granbegdate/time> –EndDate <granenddate/time> –log**

/usr/ecs/<MODE>/CUSTOM/logs/GranDel10.log –DFA ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

a. y (when prompted)

The script runs to completion and the user is prompted with the number of granules to be deleted before confirming the delete.

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel10.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <VersionID> and BeginningDateTime > "<granbegdate/time>" and EndingDateTime < "<granenddate/time>"**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and a DFA flag of 1.

26.25 3 Run SDSRV Deletion Cleanup Utility

Run SDSRV Deletion Cleanup Utility to notify STMGT of desired deletions and to physically remove granules as requested.

1. In the SDSRV database window on the **Database** workspace, view the granule IDs that will be physically deleted:

- **select dbID, ShortName, VersionID, deleteEffectiveDate from DsMdGranules where deleteEffectiveDate < <Today-lagtime>**
- **go**
- **select * from DsMdDeletedGranules where transactionTime < <Today-lagtime>**

go

A list of dbIDs and ESDTs is returned. These are the granules that will be physically deleted when the Deletion Cleanup Utility is executed in the next few steps.

2. On the **SDSRV** host and in the SDSRV window, execute the Deletion Cleanup Utility, first not entering the lag time, and then changing the lag time entered:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **ll**
- **EcDsDeletionCleanup.pl**
- Enter lag time in days: **<enter>**
- Enter lag time in days: **5**
- Confirm lag time: **N**
- Re-enter lag time: **2**
- Confirm lag time: **y**
- Enter mode of operation: **<MODE>**
- Enter log file name: **DelCleanup1.log**
- Enter Sybase User: **EcDsScienceDataServer**
- Enter Sybase User Password: **<password>**
- Enter sql server: **t1acg04_srvr**
- Enter DBName: **EcDsScienceDataServer1_<MODE>**
- Enter STMGT DBName: **stmtdb1_<MODE>**
- Enter Batch size: **3**

Do you wish to continue deleting these granules?: **y**

Execution of the Deletion Cleanup Utility begins and continues to prompt the user for the lag time until it is entered, and then executes unsuccessfully because the dbo userid was not specified.

a. View the Deletion Cleanup log file:

- **cd /usr/ecs/<MODE>/CUSTOM/logs**

vi DelCleanup1.log

There are a number of options to run concerning Lag Time such as:

- **Execute Deletion Cleanup Utility with Lag Time of a # of Days and deny the confirmation.**
- **Execute Deletion Cleanup Utility with Lag Time of a # of Days.**

b. Example: On the SDSRV host and in the SDSRV window, execute the Deletion Cleanup Utility, and confirm the lag time of two days:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **EcDsDeletionCleanup.pl**
- Enter lag time in days: **2**
- Confirm lag time: **y**
- Enter mode of operation: **<MODE>**
- Enter log file name: **DelCleanup1.log**
- Enter Sybase User: **sdsrv_role**
- Enter Sybase User Password: **<password>**

- Enter sql server: **t1acg04_srvr**
- Enter DBName: **EcDsScienceDataServer1_<MODE>**
- Enter STMGT DBName: **stmgtdb1_<MODE>**

Enter Batch size: **3**

c. Execution of the Deletion Cleanup Utility begins and the number of DFA and physically deleted granules is returned with a confirmation prompt.

Confirm the deletion when prompted:

Do you wish to continue deleting these granules?: **y** Execution of the Deletion Cleanup Utility continues to completion.

3. In the SDSRV database window on the **Database** workspace, view the DsMdStagingTable:

- **select * from DsMdStagingTable**
- **go**
- **select * from DsMdStagingTable**
- **go**
- **select * from DsMdStagingTable**

go

The SDSRV Staging table has data in it that is being transferred to the STMGT database in increments of 3 (the specified batch size). When all data has been transferred to STMGT, the table is empty. All granules from the DeletedGranules table that were placed there before the lag time, are transferred, whether they are DFA or physical deletes.

4. In the STMGT database window of the **Database** workspace, view the DsStPendingDelete table:

- **select * from DsStPendingDelete**
- **go**
- **select * from DsStPendingDelete**
- **go**
- **select * from DsStPendingDelete**

go

The STMGT Pending Delete table receives data from the SDSRV in increments of 3 (the specified batch size). All the granules expected to be transferred to STMGT (all those specified in the pretest delete requests) have been transferred.

5. View the Deletion Cleanup log. Messages in the log file include the information about the above request, and they indicate that information has been placed in the STMGT database in groups of 3 (requested batch size).

6. Execute the Deletion Cleanup Utility with a Lag Time of 0 is another option with a Batch size of 1.

26.25.3.1 Do a physical delete with a file containing a list of granule Ids and requesting no confirmation prompt. The granules are tagged for deletion

1. Perform a query in the **Database** workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select * from DsMdGranules where dbID in (xxx, yyy)**
- **go**

A list of granules is returned based on the supplied dbIDs. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes based on the input file.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

EcDsGranuleDelete –geoidfile dbids3.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel17.log –physical –noprompt ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

The script runs to completion and the operator is not prompted to confirm the delete before it occurs.

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel17.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were tagged for deletion.

4. On the **Database** workspace, view the SDSRV database to verify the granules were tagged for deletion:

- **Select * from DsMdGranules where dbID in (xxx, yyy)**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId in (xxx, yyy)**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion and a DFA flag of 0.

26.25.3.2 Do a deletion with a file containing a list of granule IDs & specify DFA

The Granules are tagged for deletion

1. Perform a query in the Database workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select * from DsMdGranules where dbID in (xxx, yyy)**
- **go**

A list of granules is returned based on the supplied dbIDs. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes based on the input file.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

- **EcDsGranuleDelete –geoidfile dbids4.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel18.log –DFA ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion and the operator is again prompted to confirm the delete before it occurs

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel18.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were tagged for deletion.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **Select * from DsMdGranules where dbID in (xxx, yyy)**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId in (xxx, yyy)**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and a DFA flag of 1.

26.25.3.3 Do a deletion with a file containing a list of local granule IDs specifying a DFA. Data is tagged for deletion.

1. Perform a query in the Database workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select dbID, LocalGranuleID from DsMdGranules where LocalGranuleID in ("aaa", "bbb", "ccc")**
- **go**

A list of granules is returned based on the supplied LocalGranuleIDs. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes based on the input file.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing Local Granule Ids:

- **EcDsGranuleDelete –localgranulefile locgrn1.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel19.log –DFA ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

  a. y (when prompted)

The script runs to completion and the operator is again prompted to confirm the delete before it occurs.

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel19.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were tagged for deletion.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select dbID, LocalGranuleID, deleteEffectiveDate, DeleteFromArchive from DsMdGranules where LocalGranuleID in ("aaa", "bbb", "ccc")**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and a DFA flag of 1.

26.25.3.4 Do a deletion with a file containing a list of local granule IDs specifying a physical delete

1. Perform a query in the Database workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select dbID, LocalGranuleID from DsMdGranules where LocalGranuleID in ("aaa", "bbb", "ccc")**
- **go**

A list of granules is returned based on the supplied LocalGranuleIds. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes based on the input file.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

- **EcDsGranuleDelete –localgranulefile locgrn4.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel22.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion and the operator is again prompted to confirm the delete before it occurs.

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel22.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and time of the request. These messages also demonstrate that the requested granules were deleted.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select dbID, LocalGranuleID, deleteEffectiveDate, DeleteFromArchive from DsMdGranules where LocalGranuleID in ("aaa" "bbb", "ccc")**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId in (dbid1, dbid2, dbid3)**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for the DeleteEffectiveDate column and a value of "N" for the DeleteFromArchive column. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 0.

26.25.4. REPEAT AS BEFORE: Run SDSRV Deletion Cleanup Utility

to notify STMGT of desired deletions and to physically remove granules as requested. STMGT is notified of granules to delete and only the physically deleted granules are removed from the SDSRV DATABASE.

---

1. In the SDSRV database window on the **Database** workspace, view the granule IDs that will be physically deleted:

- **select dbID, ShortName, VersionID, deleteEffectiveDate from DsMdGranules where deleteEffectiveDate != NULL**

go

A list of dbIDs and ESDTs is returned. These are the granules that will be physically deleted when the Deletion Cleanup Utility is executed in the next few steps.

2. On the **SDSRV** workspace execute the Deletion Cleanup Utility, and confirm the lag time:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **ll *Del***
- **EcDsDeletionCleanup.pl**
- Enter lag time in days: **0**
- Confirm lag time: **y**
- Enter mode of operation: **<MODE>**
- Enter log file name: **DelCleanup2.log**
- Enter Sybase User: **sdsrv_role**
- Enter Sybase User Password: **<password>**
- Enter sql server: **t1acg04_srvr**
- Enter DBName: **EcDsScienceDataServer1_<MODE>**
- Enter STMGT DBName: **stmgtdb1_<MODE>**
- Enter Batch size: **10**

Do you wish to continue deleting these granules?: **y**

The Deletion Cleanup Utility executes successfully, prompting the user with the number of granules to be DFA or physically deleted and whether or not to continue.

2. In the SDSRV database window on the **Database** workspace, view the DsMdStagingTable:

- **select * from DsMdStagingTable**
- **go**
- **select * from DsMdStagingTable**
- **go**
- **select * from DsMdStagingTable**

go

The SDSRV Staging table has data in it that is being transferred to the STMGT database in increments of 10 (the specified batch size). When all data has been transferred to STMGT, the table is empty. All granules from the DeletedGranules table are transferred (lag time is 'ignored' and all granules are deleted), whether they are DFA or physical deletes.

3. In the STMGT database window of the **Database** workspace, view the DsStPendingDelete table:

- **select * from DsStPendingDelete**
- **order by CreationTime**
- **go**
- **select * from DsStPendingDelete**
- **order by CreationTime**
- **go**
- **select * from DsStPendingDelete**
- **order by CreationTime**

go

The STMGT Pending Delete table receives data from the SDSRV in increments of 10 (the specified batch size). All the granules expected to be transferred to STMGT (all those specified in the above delete requests), have been transferred.

4. View the Deletion Cleanup log. Messages in the log file include the information about the above request, and they indicate that information has been placed in the STMGT database at the requested batch size.

a. In the **GUIs** workspace, view the STMGT GUI for the deletion requests just transferred to the STMGT DATABASE:

b. Select the **Delete** tab from the menu bar. The Delete screen displays number of files to be deleted for a given ESDT/Version ID pair (as noted in the above deletion requests).

4. In the SDSRV database window on the **Database** workspace, view the DsMdDeletedGranules table:

- **select * from DsMdDeletedGranules**

go

The SDSRV Deleted Granules table is again empty after all the granules have been deleted.

5. In the SDSRV database window on the **Database** workspace, view the DsMdGranules table and some additional tables to verify the correct granules where deleted:

- **select * from DsMdGranules where dbID = sss**
- **go**
- **select * from DsMdMeasuredParameter where granuleId = nnn (if applicable)**
- **go**
- **select * from DsMdFileStorage where granuleId = xxx**

go

The specified granules are not found in the various tables.

Note: where sss, nnn and xxx are dbIDs were just requested for a physical delete.

6. In the SDSRV database window on the **Database** workspace, view the DsMdGranules table and some additional tables to verify the correct granules where deleted:

- **select * from DsMdGranules where dbID = lll**
- **go**
- **select * from DsMdMeasuredParameter where granuleId = mmm (if applicable)**
- **go**
- **select * from DsMdFileStorage where granuleId = ooo**

go

The specified granules are found in the various tables.

Note: where lll, mmm and ooo are dbIDs just requested for DFA.

26.25.5 Do a deletion using GranuleID filelist representing different ESDTs

whose data resides in different archives and includes multi-file granules. Data is tagged for deletion.

---

1. In the **GUIs** workspace on the **STMGT GUI**, determine an ESDT that has data stored in multiple archives (an ESDT where the archive location has been changed, so it has a current path and previous path, and one where its data resides in a primary and a backup archive and where a granule has multiple files):

- select **VolGrpConfig tab**
- Click the **ESDT/Version** pair
- Click the **Display History** button

Record the **archive server(s)** and **paths**

The ESDTs archive servers and archive paths are found and recorded for use in the next few steps.

2. Perform a query in the **Database** workspace against the SDSRV database to determine which granules will be deleted when the Granule Deletion Client is run next:

- **select * from DsMdGranules where dbID in (aaa, bbb, ccc, ddd)**
- **go**

A list of granules is returned based on the supplied granule IDs. These will be referenced in the next few steps to compare with what the Granule Deletion Client deletes based on the input file.

3. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

- **EcDsGranuleDelete –geoidfile dbids5.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel25.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion and the operator is again prompted with the number of granules for delete, and to confirm the delete before it occurs.

4. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel25.log**
- **<esc>**
- **:q**

- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

5. On the **Database** workspace, view the SDSRV database to verify the granules were tagged for deletion:

- **select * from DsMdGranules where LocalGranuleId = aaa or LocalGranuleId = bbb or LocalGranuleId = ccc**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for the DeleteEffectiveDate column and a value of "N" for the DeleteFromArchive column. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 0.

26.25.6 Do a DFA deletion of granules with Browse, QA and PH

1. Perform a query in the **Database** workspace against the SDSRV database to determine which Browse, QA and PH granules are associated with science granules:

- **select * from DsMdGranules a, DsMdBrowseGranuleXref b where a.dbID = b.granuleId and a.ShortName like "%<ESDTName>&"**
- **go**
- **select * from DsMdGranules a, DsMdQaGranuleXref b where a.dbID = b.granuleId and a.ShortName like "%<ESDTName>%"**
- **go**
- **select ShortName, dbID, processingHistoryId from DsMdGranules where processingHistoryId > 1 and ShortName like "%<ESDTName>%"**

go

A list of granules is returned for each query request. These granules will be referenced in the next few steps for use with deleting associated granules.

a. Browse/PH/QA granules not referenced DFA delete. Granules are not tagged for deletion.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying an input file containing SDSRV Granule Ids (these Granule Ids are granules that have at least one Browse, QA and PH granule associated with them and they're not referenced by any other granule):

- **EcDsGranuleDelete –geoidfile dbids6.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel26.log –DFA ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion and the operator is again prompted to confirm the delete before it occurs.

3. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel26.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were not tagged for deletion.

4. On the **Database** workspace, view the SDSRV database to verify the granules were tagged for deletion:

- **select * from DsMdGranules where dbID = nnn or dbID = ooo or dbID = ppp**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**
- **go**
- **select * from DsMdGranules where dbID = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion. The Browse, QA and PH granules associated with the deleted science granules are still found in their respective tables with no deletion tags associated with them.

a. Browse/PH/QA granules not referenced physical delete. Granules are tagged for deletion.

5. On the **SDSRV** workspace, run the Granule Deletion Client specifying an input file containing SDSRV Granule Ids (these Granule Ids are granules that have at least one Browse, QA and PH granule associated with them and they're not referenced by any other granule):

- **EcDsGranuleDelete –geoidfile dbids7.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel27.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion and the operator is again prompted with the number of granules for deletion, and to confirm the delete before it occurs.

6. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel27.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

7. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select * from DsMdGranules where dbID = nnn**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion (Science, Browse, QA and Processing History granules), including a current time for TransactionTime and the granule's original insertTime (from the associated Granules, Browse, QA or PH table) for the insertTime, and a DFA flag of 0. The Browse, QA and PH granules associated with the deleted science granules are not found in their respective tables.

Note: where ooo, ppp, qqq, rrr, sss are granules just requested for deletion.

26.25.7 Browse/PH/QA granules referenced DFA.

26.25.7.1 Science Granules are tagged for deletion, but associated granules are not

1. On the **SDSRV** workspace, run the Granule Deletion Client specifying an input file containing SDSRV Granule Ids (these Granule Ids are granules that have at least one Browse, QA and PH granule associated with them and they are referenced by other granules):

- **EcDsGranuleDelete ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE> –geoidfile dbids8.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel28.log –DFA y (when prompted)**

The script runs to completion and the operator is again prompted with the number of granules for deletion and to confirm the delete before it occurs.

2. View the Deletion log file and the tail of the SDSRV ALOG file:
- **vi GranDel28.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were not tagged for deletion.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:
- **select * from DsMdGranules where dbID = nnn**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**
- **go**
- **select * from DsMdGranules where dbID = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of NULL for DeleteEffectiveDate and a value of "Y" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 1. The Browse, QA and PH granules associated with the deleted science granules are still found in their respective tables and no deletion tags are associated with them.

26.25.7.2 Browse/PH/QA granules referenced physical

Granules are not tagged for deletion.
(Deletes Science Granules and not Associated Granules)

1. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select * from DsMdGranules where dbID = nnn or dbID = ooo or dbID = ppp**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion, including a current time for TransactionTime and the granule's original insertTime (from the DsMdGranules table) for the insertTime, and a DFA flag of 0. The Browse, QA and PH granules associated with the deleted science granules are still found in their respective tables because they are referenced by other granules.

Note: where ooo, ppp, qqq, rrr, sss are granules just requested for deletion.

26.25.7.3 Do a deletion for granules and suppress the deletion of the associated Browse, QA and PH.

Granules are tagged for deletion but associated Browse, QA and PH granules are not. Browse/PH/QA referenced physical delete -noassoc

1. On the **SDSRV** workspace, run the Granule Deletion Client specifying an input file containing SDSRV Granule Ids (these Granule Ids are granules that have at least one Browse, QA and PH granule associated with them and they are referenced by other granules):

- **EcDsGranuleDelete ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE> –geoidfile dbids10.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel30.log –physical -noassoc**
  - **a. y (when prompted)**

The script runs to completion and the operator is again prompted with the number of granules for deletion and to confirm the delete before it occurs.

2. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel30.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the

ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select * from DsMdGranules where dbID = nnn or dbID = ooo or dbID = ppp**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**
- **go**
- **select * from DsMdGranules where dbID = qqq**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for the deleteEffectiveDate column and a value of "N" for the DeleteFromArchive column The DsMdDeletedGranules table contains the granules just requested for deletion. The Browse, QA and PH granules associated with the deleted science granules are still found in their respective tables and have no deletion times associated with them.

Note: where ooo, ppp, qqq, rrr, sss are granules just requested for deletion.

26.25.7.4 Browse/PH/QA not referenced physical delete -noassoc

1. On the **SDSRV** workspace, run the Granule Deletion Client specifying an input file containing SDSRV Granule Ids (these Granule Ids are granules that have at least one Browse, QA and PH granule associated with them and they are not referenced by other granules):

- **EcDsGranuleDelete ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE> –geoidfile dbids11.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel31.log –physical -noassoc**
  - **a. y (when prompted)**

The script runs to completion and the operator is again prompted to confirm the delete before it occurs.

2. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel31.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that the requested granules were deleted.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select * from DsMdGranules where dbID = nnn**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdBrowse where dbID = qqq**
- **go**
- **select * from DsMdQaGranule where dbID = rrr**
- **go**
- **select * from DsMdProcessingHistory where dbID = sss**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**
- **go**
- **select * from DsMdGranules where dbID = qqq**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for DeleteEffectiveDate and a value of "N" for DeleteFromArchive. The DsMdDeletedGranules table contains the granules just requested for deletion. The Browse, QA and PH granules associated with the deleted science granules are still found in their respective tables and have no deletion times associated with them.

26.25.8 Determine granules that are being referenced as input by other granules

Do a a physical deletion of granules still being referenced as inputs by other granules. Granules are not tagged for deletion but are listed.

---

1. In the **SDSRV** database window on the **Database** workspace, view the DsMdInputGranule table to determine which granules have pointers to other granules:

- **select granuleId, inputGranule, InputPointer from DsMdInputGranule where InputPointer like "%L7IGS%"**

go

Several granules that are inputs to other granules in the database are noted. Some of these granules are linked via UR and others are linked via LGID as noted in the query results.

2. On the **SDSRV** workspace, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

- **EcDsGranuleDelete –geoidfile dbids40.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel40.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)
The script runs to completion after prompting the user with the number of granules to be deleted.

3. View the Deletion log file and the tail of the SDSRV ALOG file:
- **vi GranDel40.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request, and the nature of the error encountered. These messages also demonstrate that the requested granules were not tagged for deletion because they have associated input granules.

4. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:
- **Select * from DsMdGranules where dbID = www dbID = xxx**
- **go**
- **select * from DsMdDeletedGranules**

go

The DsMdGranules table shows the granules are still in the database and they are not tagged for deletion. The DsMdDeletedGranules table does not contain the granules just requested for deletion.

26.25.9 Do a DFA delete of granules still being referenced as inputs by other granules

Granules are not tagged for deletion.

---

1. On the **SDSRV** workspace, using granules from the above database query, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:

EcDsGranuleDelete –geoidfile dbids41.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel41.log –DFA –noprompt ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>

The script runs to completion after displaying to the user the number of granules to be deleted

2. View the Deletion log file and the tail of the SDSRV ALOG file:
- **vi GranDel41.log**
- **<esc>**

- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request, and the nature of the error received. These messages also demonstrate that the requested granules were tagged for deletion.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:
- **Select * from DsMdGranules where dbID = www or dbID = xxx**
- **go**
- **select * from DsMdDeletedGranules**

go

The DsMdGranules table shows the granules are still in the database and they are tagged for deletion (contain a "Y" in the DeleteFromArchive field). The DsMdDeletedGranules table contains the granule just requested for deletion.

Where www and xxx are granule IDs just requested for deletion.

26.25.10 Do a physical delete of granules still being referenced as inputs by other granules and override the check

Granules are tagged for deletion.

---

1. On the **SDSRV** workspace, using granules from the above database query, run the Granule Deletion Client specifying the input file containing SDSRV Granule Ids:
- **EcDsGranuleDelete –geoidfile dbids43.in –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel43.log –physical –delref ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

y (when prompted)

The script runs to completion after prompting the user with the number of granules to be deleted.

2. View the Deletion log file and the tail of the SDSRV ALOG file:
- **vi GranDel43.log**
- **<esc>**
- **:q**
- **View the ScienceDataServer log tails**

View the EcDsGranuleDelete ALOG

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed. Included in these messages are the user id of the requester, the ShortName, VersionID and granule time of the request. These messages also demonstrate that

the requested granules were tagged for deletion, even though they're referenced by other granules.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **select * from DsMdGranules where dbID = yyy dbID = zzz**
- **go**
- **select * from DsMdDeletedGranules**
- **go**
- **select * from DsMdFileStorage where granuleId = nnn**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for the DeleteEffectiveDate and a value of "Y" for the DeleteFromArchive flag. The DsMdDeletedGranules table contains the granules just requested for deletion.

26.25.11 Delete files from the archive using the STMGT GUI

26.25.11.1 Single ESDT/single path deletion

1. In the **GUIs** workspace, view the ESDTs with granules targeted for deletion via the STMGT GUI:

- select **Delete** from the menu bar

All the granules tagged for deletion should be listed as ready for deletion.

2. From the list of ESDTs displayed on the delete screen, determine where a given ESDTs data is stored in the archive:

- select **Close** button
- select **VolGrpConfig tab**
- Click the **ESDT/Version pair**

Record the **archive server** and **path**

3. The ESDTs archive server and archive path are found and recorded for use in the next few steps.

ESDT Archive/path:

_____

_____

4. Select data for deletion:

- select **Delete** from the menu bar
- Click an **ESDT/Version pair**
  - **a.** Click the **Reset** button

The ESDT/Version pair is selected for deletion, and then the screen is reset, and no ESDTs are selected for deletion.

5. Select data for deletion:

- Click an **ESDT/Version pair**

a. Click the **Delete** button

The ESDT/Version pair is selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

b. Confirm the delete when prompted:

Click on **OK** button when prompted

The delete request continues to completion.

6. In the **STMGT** workspace, view the archive server log tails. The logs contain messages pertaining to the data deletion.

7. When the Delete Request is complete, view the archive path to verify the data was deleted in the **STMGT** workspace:

- **cd /dss_stk3/<MODE>/<dir>**

ll *<dbID>*

All requested data for the given ESDT has been deleted.

8. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

a. Select **Delete** from the menu bar

The ESDT/Version pair is not listed on the Delete screen as one of those ready to be deleted.

9. In the **Database** workspace, view the Pending Delete table to verify it no longer displays the data type just deleted:

- **select * from DsStPendingDelete**
- **go**

The database table listing does not include the ESDT/Version just deleted.

26.25.11.2 Single ESDT/Multiple archive and path deletion

1. In the **GUIs** workspace, view the ESDTs with granules targeted for deletion via the STMGT GUI:

a. select **Delete** from the menu bar

Granules tagged for deletion are listed as ready for deletion.

From the list of ESDTs displayed on the delete screen, determine an ESDT that has data stored in multiple archives and was used in verification of criterion 1706.11 above (an ESDT where the archive location has been changed, so it has a current path and a previous path, and one who's data resides in a primary and a backup archive and has multiple files):

- Click **Ok** button
- select **VolGrpConfig tab (if necessary)**
- Click the **ESDT/Version pair**

Record the **archive server(s)** and **paths**

The ESDTs archive servers and archive paths are found and recorded for use in the next few steps.

ESDT/Version:

_____

_____

Archives/paths:

_____

_____

2. In the **STMGT** workspace, view the paths identified above to verify data for the ESDT is archived there:

- **cd /dss_stk3/<MODE>/<dir>**
- **ls -altr *<dbID>***

on other archive window:

- **cd /dss_stk3/<MODE>/<dir>**

ls -altr *<dbID>*

Data for the given data type resides in both archives.

3. In the **GUIs** workspace on the STMGT GUI, select the ESDT with data stored in different archives for deletion:

- Select **Delete** from the Menu Bar
- Click the **ESDT/Version pair**
- Click the **Delete** button

Click the **OK** button when prompted

The ESDT/Version pair is selected for deletion, the delete is initiated, and the operator is prompted to confirm the delete.

4. In the **STMGT** workspace, view the archive server log tails for both archive servers. The logs contain messages pertaining to the data deletion.

5. When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd /dss_stk3/<MODE>/<dir1>**
- **ll** *<dbID>*
- **cd /dss_stk3/<MODE>/<dir2>**

ll *<dbID>*

The requested data for the given ESDT has been deleted.

6. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

Select **Delete** from the menu bar

The ESDT/Version pair is not listed on the Delete screen as one of those ready to be deleted.

7. In the **Database** workspace, view the Pending Delete table to verify it no longer displays the data type just deleted:

- **select distinct VersionedDataType from DsStPendingDelete**
- **go**

The database table listing does not include the ESDT/Version just deleted.

- On the **GUIs** workspace in the STMGT GUI, select the ESDTs with primary/backup data for deletion:Click the **ESDT/Version pair**
- Click the **Delete** button

  Click the **OK** button when prompted

The ESDT/Version pairs are selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

8. In the **STMGT** workspace, view the archive server log tails for both archive servers.

The logs contain messages pertaining to the data deletion.

9. When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd /dss_stk3/<MODE>/<dir1>**
- **ll** *<ESDT.Version>*
- **cd /dss_stk3/<MODE>/<dir2>**

ll *<ESDT.Version>*

All requested data for the given ESDT has been deleted, but the backup data still resides in the archive.

10. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

Select **Delete** from the menu bar.

The ESDT/Version pair is not listed on the Delete screen as one of those ready to be deleted.

11. In the **Database** workspace, view the Pending Delete table to verify it no longer displays the data type just deleted:

- **select * from DsStPendingDelete**
- **go**

The database table listing does not include the ESDT/Version just deleted.

26.25.11.3 Single ESDT/Multiple files

1. On the **GUIs** workspace in the STMGT GUI, select an ESDT with multiple files for deletion:

- Click the **ESDT/Version pair**
- Click the **Delete** button

Click the **OK** button when prompted

The ESDT/Version pair is selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

2. In the **STMGT** workspace, view the archive server log tail for the archive server.

The logs contain messages pertaining to the data deletion.

When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd /dss_stk3/<MODE>/<dir1>**

ll *<dbID>*

Check that all data files for the requested ESDT have been deleted.

3. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

Select **Delete** from the menu bar

The ESDT/Version pair is not listed on the Delete screen as one of those ready to be deleted.

4. In the **Database** workspace, view the Pending Delete table to verify it no longer displays the data type just deleted:

- **select * from DsStPendingDelete**
- **go**

Check that the database table listing does not include the ESDT/Version just deleted.

26.25.11.4 Multiple ESDT/Multiple archive and path deletion

1. In the **GUIs** workspace, view the ESDTs with granules targeted for deletion via the STMGT GUI:

a. select **Delete** from the menu bar

Granules tagged for deletion are listed as ready for deletion.

2. From the list of ESDTs displayed on the delete screen, determine two ESDTs that have data stored in multiple archives:

- select **VolGrpConfig tab**
- Click the first **ESDT/Version pair**
- Record the **ESDT, archive server(s)** and **paths**
- Click the second **ESDT/Version pair**

a. Record the **ESDT, archive server(s)** and **paths**

The ESDTs archive servers and archive paths are found and recorded for use in the next few steps.

ESDT/Version:

_____

_____

Archives/paths:

_____

_____

3. From the list of ESDTs displayed on the delete screen, determine two ESDTs that have data stored in multiple archives:

- select **VolGrpConfig tab**
- Click the first **ESDT/Version pair**
- Record the **ESDT, archive server(s)** and **paths**
- Click the second **ESDT/Version pair**

a. Record the **ESDT, archive server(s)** and **paths**

The ESDTs archive servers and archive paths are found and recorded for use in the next few steps.

4. In the **STMGT** workspace, view the paths identified above to verify data for the ESDT is archived there:

- **cd /dss_stk3/<MODE>/<dir>**
- **ls -altr *<dbID>***

on other archive window:

- **cd /dss_stk3/<MODE>/<dir>**

ls -altr *<dbID>*

Data for the given data type resides in both archives.

5. In the **GUIs** workspace on the STMGT GUI, select the ESDTs just identified for deletion:

- Click the first **ESDT/Version pair**
- Hold down the **ctrl key** and click the second **ESDT/Version pair**
- Click the **Delete** button

a. Click the **OK** button when prompted

The two ESDT/Version pairs are selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

6. In the **STMGT** workspace, view the archive server log tails for both archive servers.

The logs contain messages pertaining to the data deletion.

7. When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd <path/dir>**
- **ll** *<dbID>*
- **cd <path/dir>**

ll *<dbID>*

Requested data for the given ESDT has been deleted.

8. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

a. Select **Delete** from the menu bar.

The ESDT/Version pairs are not listed on the Delete screen as several of those ready to be deleted.

9. From the list of ESDTs displayed on the delete screen, determine an ESDT that has data stored in primary and backup archives:

- select **VolGrpConfig tab**
- Click the first **ESDT/Version pair**
- Record the **ESDT, archive server(s)** and **paths**
- Click the second **ESDT/Version pair**
- Record the **ESDT, archive server(s)** and **paths**

a. The ESDTs archive servers and archive paths are found and recorded for use in the next few steps.

ESDT/Version:

_____

_____

Archives/paths:

_____

_____

10. In the **STMGT** workspace, view the paths identified above to verify data for the ESDT is archived there:

- **cd <path/dir>**
- **ls -altr *<dbID>***

on other archive window:

- **cd <path/dir>**

ls -altr *<dbID>*

Data for the given data type resides in both archives.

11. In the **GUIs** workspace on the STMGT GUI, select the ESDTs just identified for deletion:

- Click the first **ESDT/Version pair**
- Hold down the **ctrl key** and click the second **ESDT/Version pair**
- Click the **Delete** button

a. Click the **OK** button when prompted.

The two ESDT/Version pairs are selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

Note: the ESDT/Version pair will be the same, but the backup will be suffixed with a 'B'

12. In the **STMGT** workspace, view the archive server log tails for the appropriate archive servers.

The logs contain messages pertaining to the data deletion.

13. When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd path1/dir1>**
- **ll *<dbID>***
- **cd <path2/dir2>**

ll *<dbID>*

Requested primary and backup data for the given ESDT has been deleted.

14. In the **GUIs** workspace, verify the ESDT/Versions just deleted are no longer displayed on the delete page:

- Select **Delete** from the menu bar

The ESDT/Version pairs are not listed on the Delete screen as one of those ready to be deleted.

15. In the **Database** workspace, view the Pending Delete table to verify it no longer displays the data type just deleted:

- **select * from DsStPendingDelete**
- **go**
- **select FileName from DsStFile where FileName like "%dbID%"**
- **go**

The database table listing does not include the ESDT/Versions just deleted.

26.25.12 Physically delete all files on an archive tape and recycle the tape

(MASS ADMIN needed for this scenario)

1. On the **SDSRV** workspace, run the Granule Deletion Client specifying ESDT ShortName, ESDT version and granule insert time coverage:

- **EcDsGranuleDelete –name <ShortName> -version <version no.> -insertbegin <insbegdate/time> –insertend <insenddate/time> –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel47.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

  a. y (when prompted)

The script is executed and the user is prompted with the number of granules for deletion.
Times Used:

_____

2. View the Deletion log file and the tail of the SDSRV ALOG file:

- **vi GranDel47.log**
- **<esc>**
- **:q**

  View the tail of the ScienceDataServer ALOG file

The Deletion log and the Science Data Server ALOG file display messages related to the deletion just performed.

3. On the **Database** workspace, view the SDSRV database to verify the granules were deleted:

- **Select deleteEffectiveDate, DeleteFromArchive, dbID from DsMdGranules where ShortName = "<ShortName>" and VersionID = <versionid>**
- **go**
- **select * from DsMdDeletedGranules**

go

The DsMdGranules table shows the granules are still in the database and they have a value of current time for the DeleteEffectiveDate column and a value of "N" for the DeleteFromArchive column. The DsMdDeletedGranules table contains the granules just requested for deletion.

4. On the **SDSRV** workspace, execute the Deletion Cleanup Utility, and confirm the lag time:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **ll *Del***
- **EcDsDeletionCleanup.pl**
- Enter lag time in days: **0**
- Confirm lag time: **y**
- Enter mode of operation: **<MODE>**
- Enter log file name: **<ent>**
- Enter Sybase User: **sdsrv_role**
- Enter Sybase User Password: **<password>**
- Enter sql server: **t1acg04_srvr**
- Enter DBName: **EcDsScienceDataServer1_<MODE>**
- Enter STMGT DBName: **stmgtdb1_<MODE>**
- Enter Batch size: **3**

Do you wish to continue deleting these granules?: **y**

The Deletion Cleanup Utility executes and the number of DFA and physically deleted granules is returned with a confirmation prompt. After confirming the deletion, the script completes. The number of granules corresponds to the number of granules that are on a given tape in the archive as noted during pretest activities.

a. View the Deletion Cleanup log. Messages in the log file include the information about the above request, and they indicate that information has been placed in the STMGT database.

b. In the **GUIs** workspace, view the STMGT GUI for the deletion requests just transferred to the STMGT DATABASE:

c. Select the **Delete** tab from the menu bar.

The Delete screen displays number of files to be deleted for a given ESDT/Version ID pair (as noted in the above deletion requests), and the number of files compares favorably with the number deleted (as noted in the above deletion requests).

On the **GUIs** workspace in the STMGT GUI, select for deletion the ESDT just deleted in SDSRV:

- Click the **ESDT/Version pair**
- Click the **Delete** button

Click the **OK** button when prompted.

The ESDT/Version pair is selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

5. In the **STMGT** workspace, view the archive server log tail for the appropriate archive server.

The logs contain messages pertaining to the data deletion.

a. When the Delete Request is complete, view the archive paths to verify the data was deleted in the **STMGT** workspace:

- **cd <path/dir1>**
- **ll *<ESDT.Version>***

All requested data for the given ESDT has been deleted.

b. In the **GUIs** workspace, verify the ESDT/Version just deleted is no longer displayed on the delete page:

c. Select **Delete** from the menu bar.

The ESDT/Version pair is not listed on the Delete screen as one of those ready to be deleted.

Note: Have the AMASS Administrator login to amass and determine if all the files have been removed from the amass archive tape identified earlier.

No files are found on the tape. **AMASS Administrator needed to perform this step.**

d. Have the AMASS Administrator perform any steps needed to recycle the tape and put the tape back in circulation for the same ESDT. The tape is recycled and ready for use.

26.25.13 Run Deletion Cleanup Utility on L7 Granule Deletion Output

Specifying both DFA and physical deletes.

1. On the SDSRV host and in the SDSRV window, execute the Deletion Cleanup Utility, and confirm the lag time:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **ll**
- **EcDsDeletionCleanup.pl**
- Enter lag time in days: **0**
- Confirm lag time: **y**
- Enter mode of operation: **<MODE>**
- Enter log file name: **DelCleanup75.log**
- Enter Sybase User: **sdsrv_role**
- Enter Sybase User Password: **<password>**
- Enter sql server: **t1acg04_srvr**
- Enter DBName: **EcDsScienceDataServer1_<MODE>**
- Enter STMGT DBName: **stmgtdb1_<MODE>**
- Enter Batch size: **3**

Do you wish to continue deleting these granules?: **y**

a. Execution of the Deletion Cleanup Utility begins and the number of DFA and physically deleted granules is returned with a confirmation prompt. The script completes after confirming the delete.

b. View the Deletion Cleanup log tail. Execution of the Deletion Cleanup Utility begins and the number of DFA and physically deleted granules is returned with a confirmation prompt. The script completes after confirming the delete.

c. In the **GUIs** workspace, view the ESDTs with granules targeted for deletion via the STMGT GUI:

- select **Delete** from the menu bar.

Check to see if all the granules tagged for deletion are listed as ready for deletion.

d. From the list of ESDTs displayed on the delete screen, determine where a given ESDTs data is stored in the archive:

- select **VolGrpConfig tab**
- Click the **ESDT/Version pair**

e. Record the **archive server** and **path**

2. The ESDTs archive server and archive path are found and recorded for use in the next few steps.

a. Select data for deletion:

- Click an **ESDT/Version pair**
- Click the **Delete** button

b. Click the **OK** button when prompted

The ESDT/Version pair is selected for deletion, the delete is initiated and the operator is prompted to confirm the delete.

3. In the **STMGT** workspace, view the archive server log tails.

The logs contain messages pertaining to the data deletion, which is in progress.

4. When the Delete Request is complete, view the archive path to verify the data was deleted in the **STMGT** workspace:

- **cd /dss_stk3/<MODE>/<dir>**

ll *<dbID>*

26.25.14 Setup a Cron Job to run the Granule Deletion Client

(Operational execution verification - not in criteria)

1. On the **SDSRV** workspace, create a cron job for the Granule Deletion Client:

- **cd /usr/ecs/<MODE>/CUSTOM/utilities**
- **vi delfiles**
- **0,5,10,15,20,25,30,35,40,45,50,55 * * * * /usr/ecs/<MODE>/CUSTOM/bin/DSS/ EcDsGranuleDelete –name TEST01 -version 001 -insertbegin <insbegdate/time> – insertend 12/31/2001 –log /usr/ecs/<MODE>/CUSTOM/logs/GranDel44.log –physical ConfigFile /usr/ecs/<MODE>/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode <MODE>**

-
- **esc**
- **wq!**
- **crontab delfiles**

crontab -l

2. Cron job is set up and is executed when specified in the files.

Note: A script may need to be written that will source the needed environment before the deletion script can run as a cron job. This Granule Deletion Client runs every 5 minutes starting on the hour.

3. Tail the deletion logs in the **SDSRV** workspace, so the logs can be viewed as the deletion scripts execute (via cron):

- **tail -f GranDel44.log**

tail -f DelCleanup.log

The log tails show the deletion scripts are running at the intervals in the cron file.

4. On the **Database** workspace, view the granules in the SDSRV database to verify they are being tagged for deletion and deleted according to the cron file:

- **Select dbID, ShortName, VersionID, deleteEffectiveDate, DeleteFromArchive from DsMdGranules where deleteEffectiveDate != NULL**
- **go**
- **select * from DsMdDeletedGranules**

go

The granules are either marked for deletion, or deleted from the database, depending on which cron jobs have completed.

a. On the **GUIs** workspace, view the STMGT GUI:

- select **Delete** from the menu bar

Note: Perioductly, repeat this process until all the cron jobs have executed.

26.25.15 Post Granule Deletion Actions

1. On the **GUIs** workspace, terminate each of the GUIs (DDIST, STMGT, SDSRV, MSS):

- Select **File/Close**

Each of the GUIs are terminated.

On the appropriate workspaces, stop the log tails:

ctrl c

2. Delete Granule Deletion log files after any necessary analysis has completed:

- **cd /usr/ecs/<MODE>/CUSTOM/logs**
- **ll *GranDel***

- **rm *GranDel***
- **ll *DelClea***
- **rm *DelClea***

3. Exit each of the xterms brought up in each of the workspaces.

26.25.16 Granule Deletion Summary

The purpose of the Granule Deletion capability is to allow an Operator to delete granules from the inventory/archive on demand. Deleting granules is a three part process.

The first, the Granule Deletion Client is run via the command line, based on operator inputs, to mark granules for deletion in the inventory. Granules are inaccessible (tagged for deletion) at this point. Tagged for deletion means that either the DFA flag = "Y" or the deleteEffectiveDate has a non-NULL value. This is the only part of the 3-part process where the operator has some freedom. Granules can be selected by data type and data time; data type and insert time; an input file list of geoids (type:subtype:dbID, such as SC:AST_ANC.001:14754); an input file list of LocalGranuleIDs. This is also where the operator decides whether to delete from the archive and the inventory db (physical delete) or to delete from the archive only (DFA). The operator can also use the client to simply determine which granules would be deleted if a particular set of criteria was requested.

The second, the Deletion Cleanup Utility, a perl script, is used to delete data granules from the inventory (when specified previously), and to place information in the STMGT database for subsequent deletion from the archive. Deletions are performed based on a 'lag time'. This is calculated by taking today minus the lag time (in days) and delete only granules that were tagged for deletion on or before that day (such as, lag time = 30 days; today is March 31; only delete granules that were marked for deletion on March 1 or before). All granules that meet the lag time will be deleted.

The third, which is executed via the STMGT GUI, allows the operator to select granules based on ESDT Name/Version, and actually deletes the data from the archive. Again, all granules for the requested ESDT Name/Version pair will be deleted when requested via the GUI. Data for multiple ESDTs can be deleted with one execution of the GUIs BatchDelete command.

Examples

Granule insert time

EcDsGranuleDelete -name TEST03 -version 001 -insertbegin 01/15/2001 12:00:00 -insertend 01/22/2001 13:45:00 -log /usr/ecs/TS3/CUSTOM/logs/GranDelTest.log -physicalConfigFile /usr/ecs/TS3/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode TS3

Granule data time

EcDsGranuleDelete -name TEST03 -version 001 -begindate 12/15/1995 07:41:32 -enddate 12/15/1995 13:45:00 -log /usr/ecs/TS3/CUSTOM/logs/GranDelTest.log -physical ConfigFile /usr/ecs/TS3/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode TS3

Input Geoid file

EcDsGranuleDelete -geoidfile testtest -log /usr/ecs/TS2/CUSTOM/logs/GranDel53.log -physical ConfigFile /usr/ecs/TS2/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode TS2

Input LocalGranuleID file

/usr/ecs/TS2/CUSTOM/bin/DSS/EcDsGranuleDelete -localgranulefile locgran.in -log /usr/ecs/TS2/CUSTOM/logs/GranDel19.log -DFA -noassoc ConfigFile /usr/ecs/TS2/CUSTOM/cfg/EcDsGranuleDelete.CFG ecs_mode TS2

Geoid file for input:
SC:TEST01.001:16249
SC:TEST02.001:16259
SC:TEST02.001:16260
SC:MOD29.001:15907

LocalGranuleID file for input:
TEST01 001 case7.dat
TEST01 001 case5.dat
L7CPF 002 L7CPF19980101_19981231.01

-noassoc – don't delete the associated granules
-delref – delete even if it's an input granule to another granule
-noprompt – no confirmation prompt for deletion
-display – display the geoid and LocalGranuleID for each science granule matching the input criteria, with no deletion

Deletion Cleanup Script

```
EcDsDeletionCleanup.pl
Enter Mode of Operation : TS2
Setting up environment and run the script in subshell

Enter Lag Time in Days: 0
A Lag Time of '0' Days has been entered.
Is this correct? [y/n]
y

Enter Log File name:
Using default Log File name 'DeletionCleanup.LOG'
Enter Sybase User Name : sdsrv_role
Enter Sybase password :
```

```
Enter Sybase SQL Server Name : t1acg04_srvr
Enter SDSRV's database name : EcDsScienceDataServer1_TS2
Enter STMGT's database name : stmgtdb1_TS2
Enter Batch Size(10000) : 3
Ready to get into sdsrv database... EcDsScienceDataServer1_TS2.
The number of Inventory Granules to be deleted = 15
The number of DFA Granules to be deleted = 0
Do you wish to continue deleting these granules? [y/n] :
n
Terminating granule deletion process.
```

Granule Insert From the DSS Client Driver

TBS

26.26 Machine-to-Machine Gateway

Operations Concept:

The purpose of this capability is to support reordering for reprocessing and in case of errors by a Science Investigator-led Processing System (SIPS). Though this capability can be used to resolve other issues, this is not its stated goal (i.e., extensions to support other objectives need to go through a change process).

Currently, SIPS receive data they require as input into their processing via standing orders (i.e., subscriptions). If reprocessing starts with products produced and archived by ECS, this mechanism could continue to feed higher level reprocessing at a SIPS. For example, if MODIS reprocessing were to include Level 1A and Level 1B, then ECS can push via subscriptions granules for the new Level 1B ESDT to MODAPS as they are being inserted into the ECS archive.

This paradigm does not address situations such as these:

1. The SIPS may need ancillary inputs for reprocessing that are not being regenerated or reinserted (and thus could not be delivered via subscriptions)

2. The SIPS may decide to start reprocessing from some higher processing level (e.g., level 2). In that case, the data needed for reprocessing need to be requested by the SIPS from ECS explicitly. The SIPS would submit orders on a regular basis that matches the SIPS production plan.

3. Software or hardware failure can cause the occasional loss of granules needed by a SIPS as input. An automatic re-ordering capability implemented by the SIPS would reduce operator workload in such situations.

4. The SIPS may temporarily fall behind in its processing. As a consequence, data initially staged via subscriptions may need to be restaged because they are no longer in the interface server cache.

The Machine-to-Machine gateway capability provides an automated ordering capability to cover situations such as these. It will be driven by commands submitted by a remote SIPS via ssh, and provide the following capabilities:

1. Ability to submit searches to ECS and specify the metadata to be returned. For example, this permits a SIPS to verify the availability of required data in ECS. The SIPS interface will support the native ECS metadata model (rather than the V0 model).

2. Ability to order individual ECS data granules based on UR, GranuleID or LocalGranuleID. For example, the SIPS might submit searches for inputs needed for reprocessing on a regular basis (e.g., daily). The SIPS then selects granules from these search results individually for ordering as needed, based on production schedule or current availability of processing resources.

3. Ability to stage products that are selected by a search criterion supplied in the request (this is called integrated search and order). The search result is not returned to the caller. Instead, the gateway will immediately order the found granules. For example, a program or script run by the SIPS on a regular basis may discover that inputs for a certain time range are missing. The script is not interested to track the individual granule IDs. Rather, it submits an integrated search and order, telling ECS to stage this data if available.

The interface will be based on the syntax that was designed in support of JEST/Mojo. The search capabilities will be similar to what the V0 Gateway supports:

1. Searches can include spatial and temporal data types and the full range of searchable science metadata (including local granule ID and PSA).

2. Search terms correspond to metadata attributes.

3. A search term can include lists of values. This will be interpreted as an OR-list.

4. Multiple search terms (i.e., criteria for different attributes) are assumed to be ANDed

Search results will be formatted using XML, in accordance with a DTD that follows the ECS core science data model and allows the return of product specific attributes (PSA).

Landsat data are not used as input into any SIPS reprocessing. Since they require special order handling due to billing and floating scene subsetting, they will not be accessible via the Machine-to-Machine gateway. DAAC operations will be able to specify which data types can be ordered via the SIPS interface. They must ensure that the Landsat data types are excluded.

DAAC operations will have configuration parameters to keep control over the concurrent workload at each gateway. The gateway will not queue any work, but rely on the Science Data Server and Distribution for workload queuing and recovery. The gateway will support graceful shutdown and cold restart. Orders received via the gateway will be tracked via MSS order tracking. External clients can assign their own request Ids to assist in a coordinated recovery (the gateway will not process an external request if a request with the same identifier is found in the order tracking database as already submitted into ECS).

A DAAC can run one or several gateways (e.g., a separate gateway for each SIPS). Each can be configured separately. If the DAAC is running multiple Science Data Servers because it has partitioned the Inventory, it must run separate gateways for each.

The machine-to-machine gateway will rely on ssh for the authentication of a connection. Ssh security is based on the use of a public/private key pair.

Each request must be associated with an MSS User ID. The operator will have the choice of configuring a default MSS User ID for each MTMGW server. If there is no default for a server, then the server will reject requests that do not specify a User ID.

The 209-CD-035-001 Interface Control Document for the Machine-to-Machine Search and

Order Gateway for the ECS Project should be referred to for the design and operation concepts.

This ICD is the interface specification for the ECS Machine-to-Machine search and order Gateway (MTMGW). The gateway supports searches and orders such as may be required

by a Science Investigator-led Processing System (SIPS) for obtaining large volumes of data

from the ECS archive for reprocessing. The document provides detailed operational context as well as specifications for the MTMGW message syntax and for the Extensible Markup Language (XML) used for search results. (jp)

References:

209-CD-035-001	Interface Control Document for the Machine-to-Machine Search and Order Gateway for the ECS Project
611-CD-600-001	Mission Operation Procedures Release 6A Section 28, Maintenance of Configuration Parameters
910-TDA-022 rev 03	Custom Code Configuration Parameters for ECS Release 6A.05.04. Section 1.4.6 Machine-To-Machine Gateway Parameters
625-CD-612-001	Configuration Management Training Lessons
625-CD-611-002	Database Administration-Configuration Registry

ECS Environment

The MTMGW is installed is INTHW. That is, it typically resides on host **ins01 [g0ins01, e0inso1, n0ins01, l0ins01]**, although multiple instances of MTMGW can coexist in the same mode on the same and/or different platforms.)

The configuration of an instance of MTMGW server to restrict which ESDTs are accessible to a user of that server is achieved through setting configuration parameters in the Configuration Registry. Specifically, MTMGW is installed with multiple configuration nodes in the EcCsRgRegistry Server database, under multiple modes. The multiple nodes in EcCsRgRegistry Server appear individually as follows:

EcCsMtMGateway_<instance number>

As noted in 910-TDA-022, the restriction of access is achieved by specifying, for an instance of MTMGW server, those ESDTs that are NOT accessible -- i.e., those for which access is to be excluded. These are identified to the Registry database using the ECS Registry Graphical User Interface (GUI). The specification is made using the ESDT short names. A "wild card" may be used for version number (e.g., to exclude ESDTs for L70R.*). When MTMGW is installed at the DAAC, three instances are set up, permitting selection of different access restrictions. Using ECS Assistant, each is assigned to a port, and the ports may then be restricted to particular SIPS.

Nevertheless, DAAC operations may want or need to configure the MTMGW. The Configuration Management training lesson (625-CD-612-001, March 2001) references the Configuration Parameters document (910-TDA-022) and contains a section addressing the use of the Configuration Registry GUI to view and edit configuration data in the database. It includes a procedure for launching the GUI and displaying parameters in the database, noting that a user with an authorized account that has appropriate permissions can change or delete parameters. Information on the Configuration Registry is also being added to the 6A.05.XX training lesson on Database Administration (625-CD-611-002) for delivery in July 2001. Also the 611-CD-600-001, Mission Operation Procedures Release 6A Section 28, Maintenance of Configuration Parameters has a section on

the Configuration Registry.

26.27 Reprocessing TBD

26.28 Additional information on the Preparation of Earth Science Data Types (ESDT's)

Every science data product generated and archived by the ECS must be described to the system by metadata which are put into an inventory and then used to retrieve and distribute the data to users of the system. The ECS Earth Science Data Model organizes the metadata into groups of related attributes and services to be performed on the data products. Granules of the same type of science data are grouped into collections. Every collection is described by an Earth Science Data Type (ESDT) and is made known to the system by an ESDT descriptor file and associated software code which is built into the Data Server's dynamic link library (DLL) to perform the services. The ESDT descriptor is composed of sections containing the following information:

Collection level metadata attributes with values contained in the descriptor.
Granule level metadata attributes whose values are supplied primarily by the Product Generation Executives (PGEs) during runtime.
Valid values and ranges for the attributes.
List of services for the data and events which trigger responses throughout the system.

The ESDTs for all data collections to be input to or output from the PGEs must be built and registered into the ECS before any of the PGEs can be run under the automated processing system.

During the past year, the ECS has collected information from the Instrument Teams on the ESDTs needed for their science software in the Release 6A.05. This information has been baselined and a set of ESDT descriptor files have been built and tested according to this baseline. The baselined ESDT descriptor files reside in the Release 6A under ECS configuration management in a VOB called the ECS Configuration Area. Since science software has been developed to the baseline, any changes to the baselined ESDT descriptor files should be rare.

26.28.1 Comparing Granule Level Metadata

A PGE accesses granule level metadata attributes and values via a Metadata Configuration File (MCF). There is typically one MCF for each output data set. The ESDTs which have been built and registered in the ECS contain a section for granule level metadata attributes and values for each data set. In terms of content, the MCFs and the granule level metadata section of the corresponding ESDT descriptor files have to be in sync.
Few changes are expected in the Inventory section of the MCF. Changes are more likely to be expected in the Archive section of the MCF and in the Product Specific Metadata. If there are any changes, a new version of the baselined ESDT descriptor file must be generated.

26.28.2 Installing/Removing (ESDT/DLL) using SDSRV Operator GUI

Before the ECS can process data, an Earth Science Data Type must be installed into the system via the Science Data Server (SDSRV). The ESDT allows the system to recognize a particular data type and also provides services for accessing the data in the form of a Dynamic Link Library (DLL). The following procedures give step-by-step instructions on configuring the ESDT and installing the ESDT using the Science Data Server GUI., see Figure 26.28.2-1. Science Dataserver Operator GUI.

26.28.2.1 Installing a single ESDT with Dynamic Link Library (DLL)

1 Copy the ESDT descriptor file and ESDT/DLL file from the source directory to the directory under the current mode of operations, The ESDT descriptor files are installed in the specified mode.

DLL's located : /usr/ecs/<mode>/CUSTOM/lib/ESS
ESDT Descriptors Located: /usr/ecs/<mode>/CUSTOM/data/ESS

2 Ensure that the following servers are currently executing: **Advertising Service** on the appropriate ADSHW HWCI server machine, **Data Dictionary Service** on the appropriate DMGHW HWCI server machine, **Science Data Server** on the appropriate ACMHW HWCI server machine and the **Subscription Service** that operates on the appropriate CSS server machine.

3 Start the **SDSRV GUI** by entering the following at the UNIX prompt on the SDSRV GUI workstation:

On workstation **x0acs##,** at the UNIX prompt in a terminal window, type as in step a, below, then enter your user id and password.
NOTE: The x in the workstation name will be a letter designating your site:
g = GSFC, m = SMC, l = LaRC, e = EDC, n = NSIDC, o = ORNL, a = ASF, j = JPL, p = PVC; the ## will be an identifying two-digit number (e.g.,g0acs03 indicates a Science Data Server Subsystem (SDSRV) workstation at GSFC).

4 Then rlogin, and enter **setenv DISPLAY <local_workstation IP address>:0.0**. The **<ipaddress>** is the ip address of **x0acs##**, and xterm is required when entering this command on a Sun terminal.

26.28.3 Quick Start Using the Science Data Server

To invoke the ECS Science Data Server GUI, the user types the following command line:

1 **telnet** to (SDSRV) **p0acs03** [e.g.]
2 login: **ID, password:**
3 cd /usr/ecs/<mode>/CUSTOM/utilities/EcDsSdSrvGuiStart <mode> Enter:
>EcDsSdSrvGuiStart <mode>

- <**mode**> is the ECS Mode for the execution, e.g., **OPS, TS1.**
- This command will bring up the GUI with the main screen appearing. See Figure 26.11.3-1. The user can then initiate the actions described in the following section.

4 On the main screen select the **Data Types** tab. A list of the ESDTs that have already been installed on the SDSRV will be displayed.

5 Click the **Add** button, this will bring up a first smaller gui. Go below to bring up the **Add Data Type** window, this will bring up a second small gui.

6 **Descriptor Filename:** enter path to where the ESDT/DLL is located, including the full ESDT descriptor. Click **OK**. Control goes back to the first gui.

Note: Within the ESDT descriptor there is a DLL Filename identifying which DLL is to be used with this ESDT.

The descriptor filename and DLL Filename will require the complete directory path name as part of the file name which is the same directory as was specified in step 1 above. (isolate the particular Data Type from the larger List, by using a unique sequence of letters or numbers at the end of the full path to better identify the Data Types ie;**/*___*).**

To specify specific directories, the File button to the right of the Descriptor Filename and DLL Filename data entry fields will bring up a standard file selection GUI for this purpose. Also note that the Archive ID field will be constructed using the DSS Storage Management Staging Server UR that is found in the Science Data Server configuration file. The Science Data Server Configuration file is located in:

/usr/ecs/<mode>/CUSTOM/cfg/EcDsScienceDataServer.CFG.

Example: If the **DSSSTMGSTAGEINGSERVERUR** field was set to **DRP1_OPS:VG1** then the Archive ID fields will automatically be set to **DRP1_OPS.**

7 Click the **Ok** button, this will cause the **Add Data Type** window to initiate installation of the ESDT/DLL into the Science Data Server.

8 The Science Data Server GUI will respond in a short time with a window stating that: **MM/DD/YY HH/MM Finished adding ESDT's**. Also, the ESDT will appear alphabetically on the **Science Data Server - Data Types** list under the **Data Types** tab.

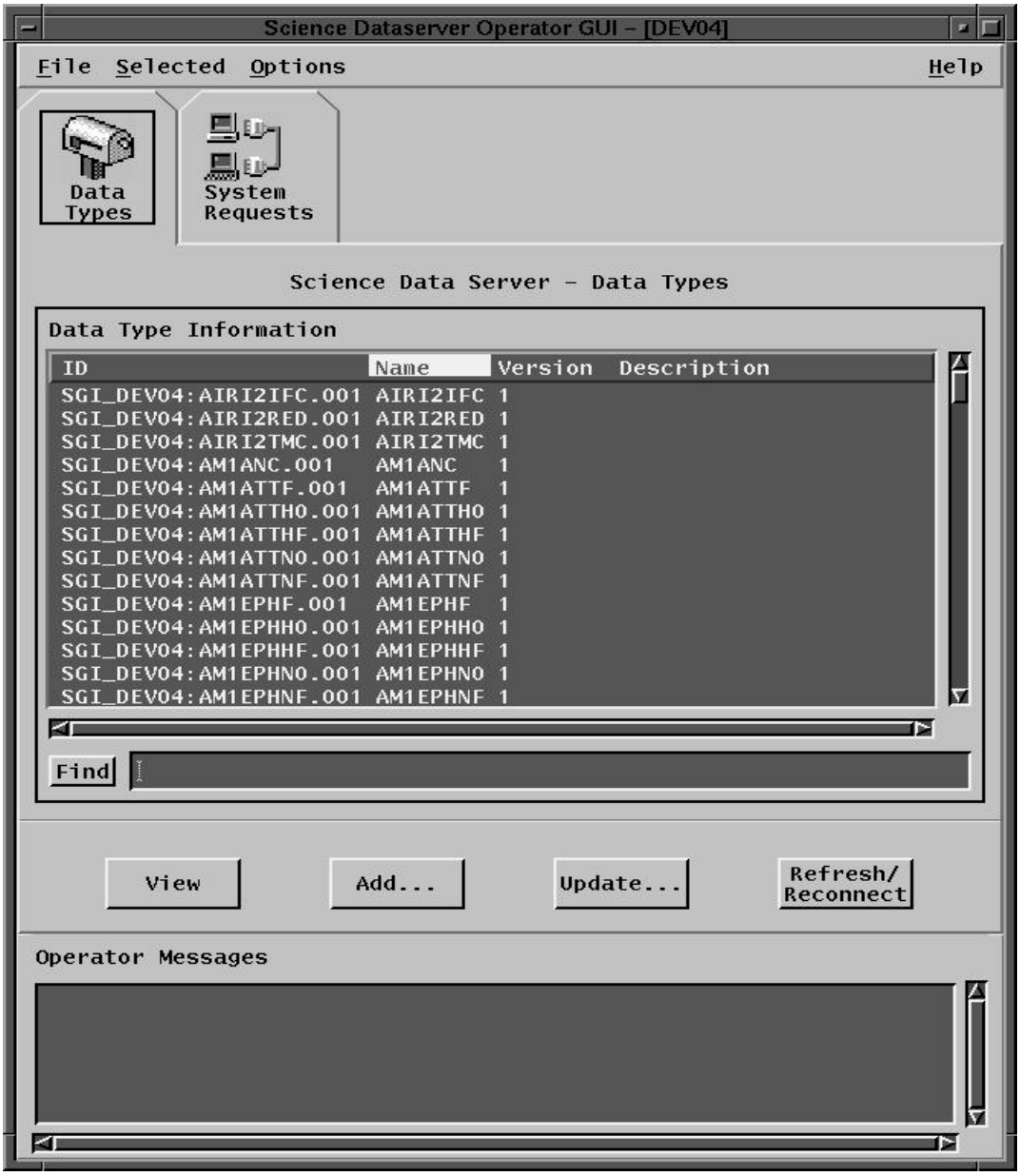


Figure 26.28.3-1. Science Dataserver Operator GUI Shown with Default Data Types Tab.

26.28.3.1 Validating Successful ESDT Installation

Criteria for success:

The **SDSRV** will display an Event ID to the fact that a new ESDT has been installed successfully.

The following servers will also need to have acknowledged a successful ESDT Event ID before additional work can be done: **ADSRV, DDICT, & SBSRV**.

26.28.4 Science Data Server

The SDSRV provides the SDPS with a catalog of Earth Science Data holdings, and the Earth Science Data Type services that operate on the data. The SDSRV provides a catalog of metadata describing the archived data holdings of the SDPS and provides mechanisms to acquire the data from the archive. The SDSRV also provides data type service on the catalog and a data reduction or sub-setting and reformatting service.

The Science Data Server Operator GUI provides the operator two major functions, the management of Earth Science Data Types and the management of all types of requests the Science Data Server Operator is involved with. Further details on these two functions are given in Table 26.28.4-1.

Table 26.28.4-1. Common ECS Operator Functions Performed with the Science Data Server Operator GUI

Operating Function	GUI	Description	When and Why to Use
Manage Science Data Server Earth Science Data Types (ESDTs)	Data Types Tab	Allows operators to manage the ESDTs offered by the Science Data Server	As needed, to manage data type descriptor information and add and update ESDTs
Manage Data Server System Requests	System Requests Tab	Allows operators to manage all the requests within each data server component	As required, to manage requests in each data server component

26.28.4.1 Science Data Server Main Screen

The ECS Science Data Server operator GUI, shown in Figure 26.28.4-1, has two tabs that provide access to each one of the components' screens.

- The Earth Science Data Type Manager is accessed through the **Data Types** tab
- The System Request Manager is accessed through the **System Request** tab.

The operator can select from the menu bar items at the top of the Science Data Server Operator window for getting help and activating less-frequently used functions. The menu bar capability is available on all Science Data Server Operator GUI screens. The following menus are available:

- **File** - which includes the following item:

– **Exit** (Ctrl-Q) - Exit application (graceful exit).

- **View** - functionality has not been determined as of this time (TBS).
- **Options** - This menu includes the *System Settings* item that opens the Server Polling Options window. Polling of the data server can be switched On/Off and the SdSrv Polling rate can be adjusted through this window shown in Figure 26.28.4-1
- **Help** - which provides context sensitive help.

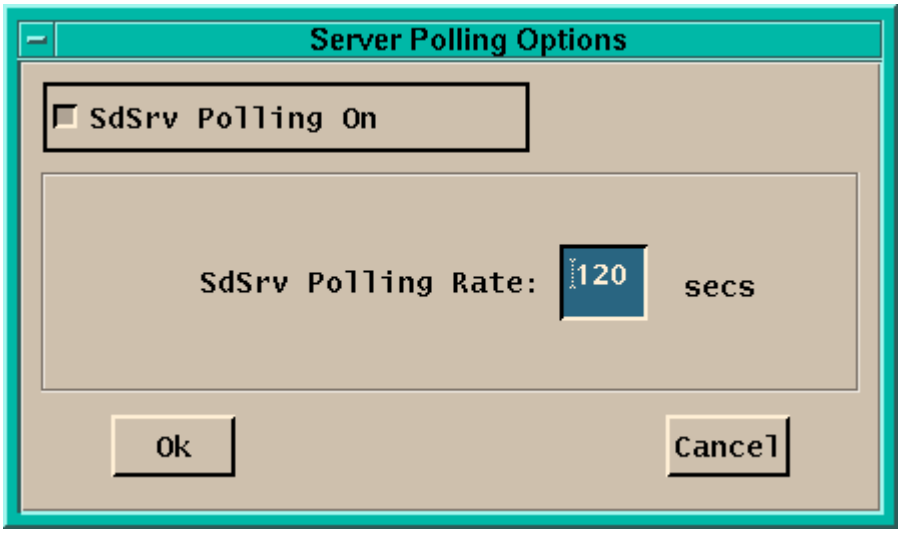


Figure 26.28.4-1. Science Data Server - Server Polling Options

Table 26.11.4-2 describes the information fields on the Server Polling Options screen.

Table 26.28.4-2. Science Data Server - Server Polling Field Description

Field Name	Data Type	Size	Entry	Description
Polling Rate	integer	4 digits	Optional	Specify the rate at which the Science Data Server Operator GUI is updated with data coming from the Data Server. The polling rate default is 120 seconds.

26.28.5 Data Types Tab

The Data Types Tab is the default screen of the Science Data Server Operator GUI shown in Figure 26.28.5-1. This window provides operations personnel at the DAAC the capability to view descriptor information, add new ESDTs and update ESDTs. A list of currently installed ESDTs is shown along with a version number and a brief description of the structure for an ESDT. Additional information that describes the structure, contents, and services that each existing ESDT provides can be viewed by selecting the data type and clicking on the *View* button.

Table 26.28-5-1. describes the Science Data Server Operator - Data Types fields.

Table 26.28.5-1. Science Data Server Operator - Data Types Field Description

Field Name	Data Type	Size	Entry	Description
Data Type ID	character	8	System generated	Uniquely identifies the specific type of ESDT.
Name	Character	25	System generated	Name of ESDT.
Version	Integer	3	System generated	Version number of ESDT, assigned starting at 1.
Description	Character	255	System generated	Includes structure and services available for an ESDT.
Find	Character	255	Optional	This functionality is provided in order to help the user browsing very long ESDT lists.

In addition, the following buttons are provided:

- **View** displays ESDT descriptor information (read-only) and its associated dynamic data link library (DLL) filename. Descriptor information consists of groups, objects, and keywords that define an ESDTs metadata, advertised services, subscribable events, data dictionary information, validation criteria, and science parameters. Descriptor information is necessary for the Science Data Server to properly configure itself to perform services related to an ESDT. A DLL is an executable library that is loaded dynamically when needed to fulfill ESDT services. The Science Data Server - Descriptor Information Dialog (see Figure 26.28-5-1 below) provides the following buttons:
  - **Close** exits the dialog without performing any operations.
  - **Help** displays on-line help information.

Figure 26.28-5-1. Science Data Server - Descriptor Information Dialog

- **Add** opens the Data Type Dialog (see Figure 26.28.5-1) which is used to add a new ESDT to the existing installed list of data types based upon input information. The SDSRV GUI has the capability to install multiple ESDTs. Click on the **File…** button to display a list of descriptor filenames to choose from instead of typing them in. Multiple descriptor files can be selected. Click the **OK** button to add the data type. If no error messages appear, then the operation has been successfully completed. Click the **Cancel** button to close the dialog without performing an operation. Click the **Clear** button to start all over again the process of filling in new information. Click the **Help** button to display on-line help information.

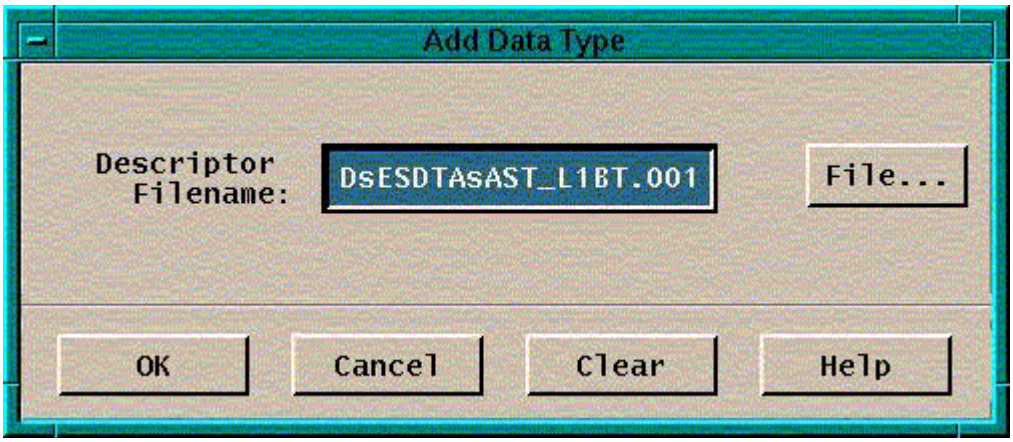


Figure 26.28-5-2. Science Data Server - Add Data Type Dialog

Table 26.11.5-2, describes the Science Data Server - Add Data Type fields.

Table 26.28.5-2. Science Data Server - Add Data Type Field Description

Field Name	Data Type	Size	Entry	Description
Descriptor Filename	Character string	25	required	Name of an ASCII file containing the ESDT descriptor file.

- **ADD ESDT** GUI is similar in use to the Update ESDT GUI and guideline.
- **Update** opens the Update ESDT Dialog (see Figure 26.28.5-3) which is used to update a ESDT to the installed list of data types based upon input information. The SDSRV GUI provides the capability to update multiple ESDTs at one time. The Science Data Server needs to be running in **Maintenance mode** to accept this operation. Click on the **File…** button to display a list of descriptor filenames to choose from instead of typing them in. Multiple descriptor files can be selected. Click the **OK** button to update the data type. If no error messages appear, then the operation has been successfully completed. Click the **Cancel** button to close the dialog without performing an operation. Click the **Clear** button to start all over again the process of filling in new information. Click the **Help** button to display on-line help information.

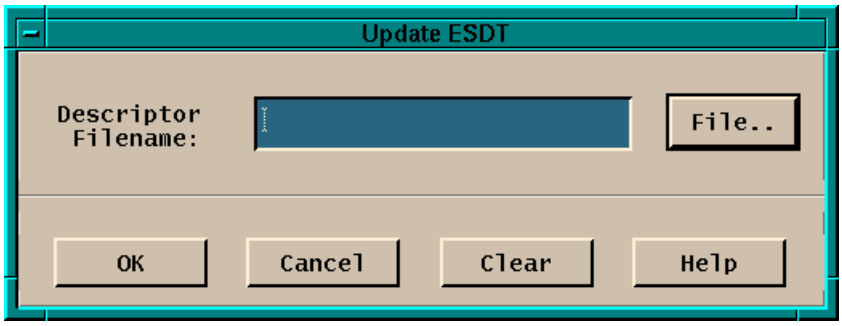


Figure 26.28.5-3. Science Data Server - Update Data Type Dialog

Table 26.11.5-3, describes the Science Data Server - Update Data Type fields.

Table 26.28.5-3. Science Data Server - Update Data Type Field Description

Field Name	Data Type	Size	Entry	Description
Descriptor Filename	character string	255	required	Name of an ASCII file containing the ESDT descriptor file.

- **Refresh/Reconnect** updates the data type information screen with current information.
- **Operator Messages** provides the functionality that displays informational and error messages to the user.

26.28.6 System Requests Tab

Clicking the System Requests tab will bring up the System Management Requests window (see Figure 26.28.6-1). This window provides operations personnel at the DAAC the capability to monitor requests the Science Data Server is working with. All requests within the Science Data Server are displayed. The columns of the list can be sorted by positioning the cursor and clicking on the appropriate column of interest. The requests can be filtered by positioning the cursor and clicking on the **Filter** button and entering the attributes on which to filter.

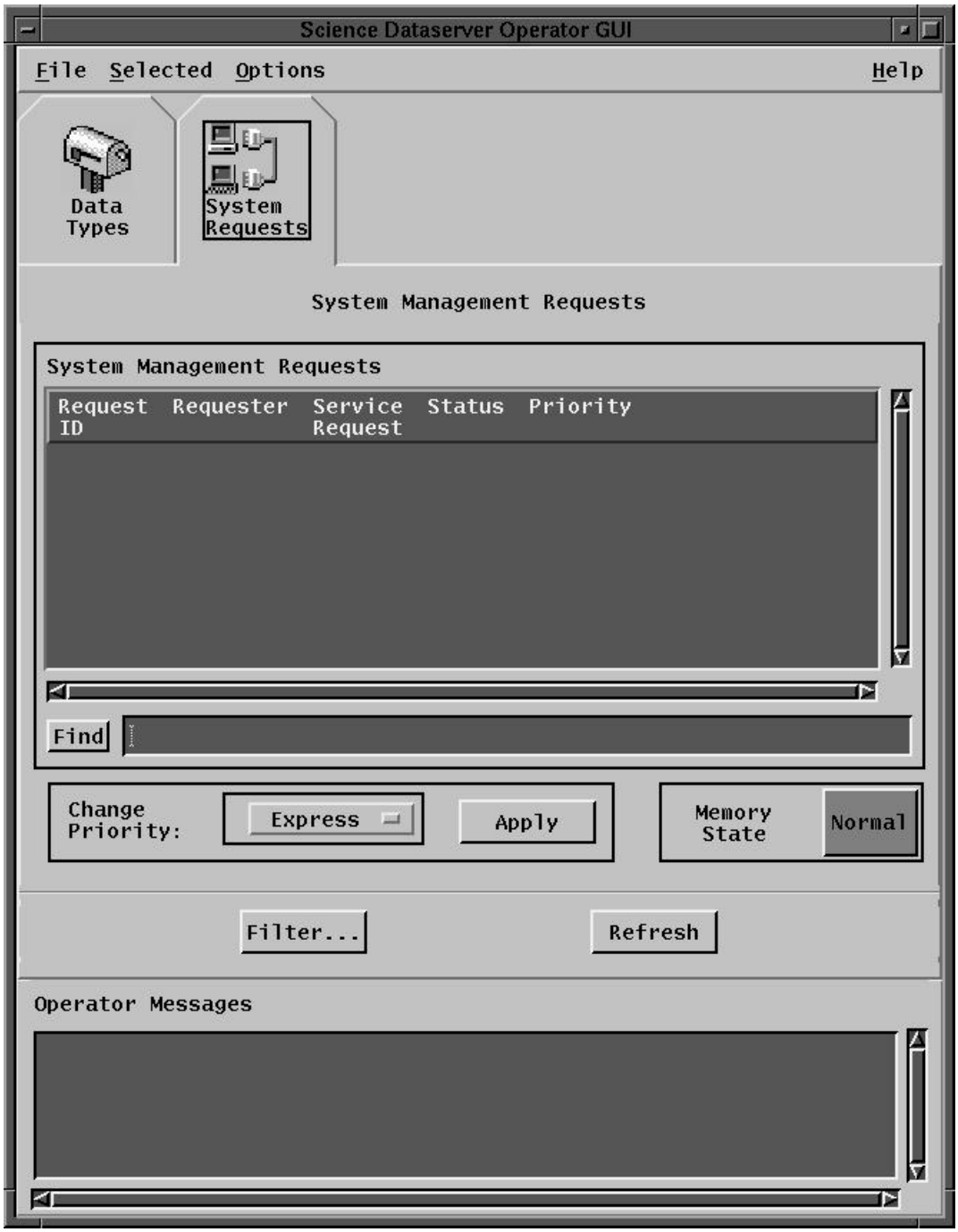


Figure 26.28.6-1. System Management Requests Window

Table 26.28.6-1, describes the System Management Requests Window fields.

Table 26.28.6-1. System Management Requests Field Description

Field Name	Data Type	Size	Entry	Description
Request ID	character	255	system generated	Unique identifier for the request.
Requester	variable character	100	system generated	Identifies the user that submitted the request.
Service Request	character	25	system generated	Types of requests handled are Insert, Acquire, and Delete
Status	character	20	system generated	Possible states are Submitted, Queued, Executing, Failed_Retryable, Failed_Fatal, Failed_Unknown and Done
Priority	variable character	20	system generated	Priority of the data server system requests, i.e., Express, Very High, High, Normal(default), Low.
Find	character	255	optional	If the list is too long, this field can be used to search for an entry

In addition, the following buttons are provided:

- **Change Priority:** changes the priority of each selected request through a pulldown menu. Possible values are: Express, Very High, High, Normal (default) and Low.
- **Apply** allows the operator to commit to the priority change selected through the change priority button.
- **Filter…** (see Figure 26.28.6-2) brings up the System Management Filter Requests dialog which provides a selection of attributes on which to filter for the list of System-wide requests. Filter on system management requests by entering the desired information, then clicking on the Request ID or Requester radio button for the desired attribute. Return to the original list of requests by clicking on the All Requests radio button. Click on other filters associated with State and Priority by clicking on the toggle button. Filter on every attribute associated with a category by clicking the **All** button or clear a category of filters by clicking on the **None** button.
- **Memory State:** it monitors the current memory state of the data server in regards to values that are set on the server side through configuration parameters. Possible values are: Normal(green color), Low(yellow), Very Low(red). This functionality will only be visible if the server's DSSMEMORYMONITORDISABLEFLAG is off.

Figure 26.28.6-2. System Management Filter Requests Dialog

Table 26.28.6-2 describes the System Management Filter Requests Dialog fields.

Table 26.28.6-2. System Management Filter Requests Field Description

Field Name	Data Type	Size	Entry	Description
Request ID	character	255	system generated	Unique identifier for the request.
Requester	variable character	100	system generated	Identifies the user that submitted the request.

In addition, the following buttons are provided:

- **OK** implements filter criteria, and the dialog closes.
- **Apply** implements filter criteria, and the dialog remains open for additional filtering.
- **Cancel** closes the dialog without saving
- **Help** displays on-line help information.

- Back to the System Requests tab description (figure 26.28.6-1), **Operator Messages** provides informational and error messages to the user.
- **Refresh** causes the Data Server to be polled for an update on Requests.

26.28.7 Required Operating Environment

For information on the operating environment, tunable parameters, and environment variables refer to the 920-TDx-013 "Custom Code Configuration Parameters" documentation series . The "x" refers to the installed location, e.g. 920-TDG-013 is for GSFC DAAC.

26.28.7.1 Interfaces and Data Types

Table 26.28.7.1-1, lists the supporting products that this tool depends upon in order to function properly.

(Change needed to replace DCE)

Table 26.28.7.1-1. Interface Protocols

Product Dependency	Protocols Used	Comments
SDSRV and all clients	DCE	via client libraries
SDSRV GUIs	X-11	via client libraries

26.28.7.2 Databases

The Science Data Server Operator GUI does not include the direct managing of any database. It has an interface with the Science Data Server Data Base: however this interface is based on a simple parameter passing function. For further information of the Science Data Server Data Base refer to 311-CD-107-005, *Science Data Server Database Design and Schema Specifications for the ECS Project*.

26.28.7.3 Special Constraints

The Science Data Server Operator GUI runs only if the Science Data Server is running in the background. Note also that at the moment the Science Data Server GUI is started through a command line that specifies the configuration file that is used to initialize the GUI Application.

26.28.7.4 Outputs

There is no processing associated with the operation of this GUI. The information provided to the operator are retrieved from the Data Server Database described in the previous sections and displayed through the screens discussed therein.

26.28.7.5 Event and Error Messages

Both event and error messages are listed in Appendix A.

26.28.7.6 Reports

No reports are produced by this tool.

26.28.8 Browser to View ECS Science SDSRV Database

1 Connect to the **SDSRV (P0acs03)** database with login information as follows:

Server Name: **p0acg01_srvr**

User Name: **sdsrv_role**

Password: **welcome**

5 The Browser lets you view all the tables in the **SDSRV** database with the mode you have selected. Example**: <EcDsscienceDataserver1_TS1**> for the PVC.

26.28.8.1 Using isql commands in the SDSRV

1. **ssh p0acs03**
2. **Password:**
3. **Isql -Usdsrv_role -Pwelcome -p0acg01_srvr**
4. **1> use EcDsScienceDataserver1**
5. **2>go**
6. **3>select * from <choose table_name>**
7. **4>go**

26.28.8.2 wisqlite is an isql Browser to View ECS Data Bases

1. login: **ssh p0pls02**
2. **cd /tools/tcl/bin**
3. Execute: **wisqlite &**
4. User ID: **pdps_role**
5. PW: **welcome**
6. Select server_name by clicking on Server button p0pls01_srvr
7. Select: **Sign on**
8. Enter: **pdps, objects, tables**
9. **Select * from <table your choice>**

26.28.8.3 Tables to Track .met files

The following tables are useful to track down the problems in insert ***.met**:

1 DsDeDictionaryAttribute

2 DsMdAddtionalAttributes

3 DsMdCollections - View with ECS Assistant only

4 DsMdGranules - View with ECS Assistant only

26.28.9 Removing ESDT's from Archive Area using Command Line Scripts

26.28.9.1 Removing ESDT's using Command Line Script Procedures:

1 **telnet** to (SDSRV) **p0acs03**[e.g.]

6 **login:, password**: <log into cmshared>

7 **cd /usr/ecs/TS1/CUSTOM/local**

5 Run script: **cleanesdt.csh <descriptor_file_name>**

- The ESDT will be removed from all four servers, SDSRV, SUBSRV, ADSRVR & DDICT. Check SDSRV Gui to see if removal was successful.

6 Now you have to log into each server and bring each of the four servers mentioned above, down by using the **KILL Script** and the **START Script** that have been specifically taylored for each server at the path identifed below in each subsystem:

cd/home/cmshared/bin/

- **SDSRV you are already on, p0acs03, SBSRV is on p0ins01, ADSRV & DDIC are on p0ins02.**

26.28.9.2 Staging ESDT's for Installation

1 Copy into a directory from which the Science Data Server will use to bring ESDT's into the Science Data Server Data Base

2 Once you have the descriptor file copied into <your_directory> in the **p0ins01** machine, you have to login into **p0acs03** and copy the file into **/usr/ecs/TS1/CUSTOM/data/ESS** directory. To do this, log into **p0acs03.** From the directory **/usr/ecs/TS1/CUSTOM/data/ESS,** Type command,
cp /home/your_directory/ESDTs/filename .

(Where /home/your_directory/ESDTs is the directory into which you stored the descriptor file.)

Now you have the copy of the latest descriptor file in the **ESS** directory.

26.28.10 Installing ESDT's using the Science Data Server GUI

Procedures:

1 **telnet** to (SDSRV) **p0acs03**[e.g.]

2 **login:, password**: <always log into **cmshared**> to enable use of isql commands and to bring servers up and down.

3 **cd /usr/ecs/TS1/CUSTOM/utilities**

4 If an ESDT has to be removed before replacing it, run script: **cleanesdt.csh <descriptor_file_name>**

- The ESDT will be removed from all four servers, SDSRV, SUBSRV, ADSRVR & DDICT. Check SDSRV Gui to see if removal was successful.

5 Now you have to log into each server and bring each of the four servers mentioned above, **down** by using the **KILL Script** and the **START Script** that have been specifically taylored for each server at the path identifed below in each subsystem:

cd/home/cmshared/bin/

- **SDSRV is on, p0acs03, SBSRV is on p0ins01, ADSRV & DDICT are on p0ins02.**

6 Once you know that all the 4 Science Data Servers are down, then log into **p0acs03** machine, go to directory **/usr/ecs/TS1/CUSTOM/utilities** and execute command **EcDsScienceDataServerStart TS1 StartTemperature cold**

(This script will bring up the four Data Servers.)

- Then check the status of the servers by using a Server Monitor GUI WHAZZUP. Once all the servers are up, then you can start installing the ESDT.

7 From **p0acs 03,** go to the directory, **/usr/ecs/TS1/CUSTOM/utilities,** type command: **EcDsSdSrvGuiStart TS1**

The Science Data Server GUI will be brought up. You will see the list of already installed ESDTs there. To add new one, click on the **Add** button. You will get the **Add ESDT** window. This window is pointing to: **/usr/ecs/TS1/CUSTOM/data/ESS** where ESDT's have been staged. Click on the **File** button to select the descriptor file. Select the correct .desc file and click on **OK**. Enter ArchiveID as **DRP1_TS1.** Click on **OK**. Wait and check the message at the bottom of the window. You will see message – **Successfully added**…Click the **Refresh** button and you will see recently added ESDT into the list.

26.28.10.1 Re-install or Update ESDT's using the Science Data Server GUI

- Only the Science Data Server (SDSRV) needs to be brought down/up using the Kill and Start scripts before doing step 4 below.

Procedures:

Note: If SDSRV is already running, you will have to Kill it first and then bring it up in the **maintenance mode**.

1 **telnet** to (SDSRV) **p0acs03**[e.g.]

2 **login:, password**:

3 **cd /usr/ecs/TS1/CUSTOM/utilities**

(Log into the (SDSRV) using **Maintenance Mode.** This locks out other users while

The data base is being update with ESDT's.)

4 Execute command :
EcDsScienceDataServerStart TS1 StartTemperature maintenance
(This script will bring up the Science Data Server.)

- Then check the status of the SDSRV using a Server Monitor GUI **WHAZZUP.** Once all the servers are up, then you can start installing the ESDT.

5 From **p0acs 03,** go to the directory, **/usr/ecs/TS1/CUSTOM/utilities,** type command:
EcDsSdSrvGuiStart TS1

6 The Science Data Server GUI will be brought up. You will see the list of already installed ESDTs there. To **Update** a ESDT, click on the Update button. You will get the **Update ESDT** window.
(This window is pointing to: **/usr/ecs/TS1/CUSTOM/data/ESS** where ESDT's have been staged.)

7 Click on the **File** button to select the descriptor file. Select the correct .desc file and click on **OK**. Enter ArchiveID as **DRP1_TS1.** Click on **OK**. Wait and check the message at the bottom of the window. You will see message – **Successfully added**…Click the **Refresh** button and you will see recently added ESDT into the list.

Note: The (enter Archive ID as entry will go away with Release 6A).

26.29 Appendix A. 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.*

A.1 Template ODL Files

There are five Template ODL files listed here. The specific or tailored ODL files listed in Sections A.2 through A.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

A.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 suposed 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 explination 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.                        */
/*************************************************************************/

TILE_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 ("<PGE name>#<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 ("<PGE name>#<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 = "TRWpl182"                          */
/*         implies file $DPAT_ESDT_SCIENCE_MD/ESDT_TRWpcal182.odl */
/*         already exists in the SSIT environment, and that       */
/*         ESDT Short Name "TRWpl182" 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 fromt he 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 unqiue naming of granules       */
/*            input 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.                    */
/*         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 constaints 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 only  */
```

```
/* has one file per granule.  Note that this is separate from the */
/* MIN/MAX_GRANULES_REQUIRED and MIN/MAX_GRANULE_YIELD parameters which */
/* 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 has */
/* 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 paramater 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 avaiable      */
/*              -- 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 avaiable   */
/*             -- 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 Attibutes (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 mult-containered metadata group. */
/*           -- For Product Specific Attibutes (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 mult-containered metadata group. */
/*           -- For Product Specific Attibutes (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 mult-containered metadata group. */
/*           -- For Product Specific Attibutes (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 mult-containered metadata group. */
/*           -- For Product Specific Attibutes (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 in not really inportant.  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 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.                       */
/*         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 occrus 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 flaged          */
/*            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
```

A.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_Defintion for         02/04/98    */
/*            Product Specific Attibutes.                     */
/*    -- 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    */
/*          DEFINTION 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 consided   */
/*            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 abberviation (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 abberviation (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 abberviation (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 detemine 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 Defintion 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 Defintion 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 "AdditionalAttibutes"                        */
/*         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 Attibutes (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 Attibutes (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          */
/*              -- The 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
```

A.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 configruation 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 */
/*             follwoing:                                              */
/*                "MMM DD YYYY HH:MM:SS", where                        */
/*                 YYYY=4 digit year, MMM=3 character abreviation 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
```

A.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. */
/*                                                                 */
/* Cooridnate 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            */
/*                ^                                   |            */
/*                |                                   |            */
/*                |                                   v            */
/*                o<----------------------------------o            */
/*             Cooridnate 4                    Coordinate 3         */
/*                                                                 */
/*    Or:                                                          */
/*
/*            Coordinate 2                    Coordinate 3         */
/*                o---------------------------------->o            */
/*                ^                                   |            */
/*                |                                   |            */
/*                |                                   v            */
/*                o<----------------------------------o            */
/*             Cooridnate 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
```

A.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 intrepeted 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 intrepeted 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
```

A.2 Typical ASTER PGE & ESDT ODL Files

Listings are provided for the following ASTER ODL files:

A.2.1 ASTER PGE ODL file for PGE_NAME BTS
A.2.2 ASTER ESDT ODL file for DATA_TYPE_NAME AST_LIB
A.2.3 ASTER ESDT ODL file for DATA_TYPE_NAME AST_ANC
A.2.4 ASTER ESDT ODL file for DATA_TYPE_NAME AST_04
A.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)

A.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 = O98
   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 = O97
   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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.3 Typical MISR PGE & ESDT ODL Files

Listings are provided for the following MISR ODL files:
A.3.1 MISR PGE ODL file for PGE_NAME MPGE1 (M1AN)

A.3.2 MISR ESDT MISANCGM ODL
A.3.3 MISR ESDT MIRCCT ODL

A.3.4 MISR ESDT MISL0AN ODL

A.3.5 MISR ESDT ActSched ODL
A.3.6 MISR ESDT MIANCSSC ODL
A.3.7 MISR ESDT MIANCAGP ODL
A.3.8 MISR ESDT MIANPPAN ODL
A.3.9 MISR ESDT MISL0SY1 ODL
A.3.10 MISR ESDT MISL0SY2 ODL
A.3.11 MISR ESDT MISL0SY3 ODL
A.3.12 MISR ESDT MIRFOIAN ODL
A.3.13 MISR ESDT MIB2GEOP ODL
A.3.14 MISR ESDT MIANCARP ODL (Version# 001)
A.3.15 MISR ESDT MIANCARP ODL (Version# 002)
A.3.16 MISR ESDT MICNFG ODL
A.3.17 MISR ESDT AM1EPHN0 ODL
A.3.18 MISR ESDT AM1ATTNF ODL
A.3.19 MISR ESDT MIANRCCH ODL
A.3.20 MISR ESDT MIL1A ODL
A.3.21 MISR ESDT MISBR ODL
A.3.22 MISR ESDT MISQA ODL
A.3.23 MISR ESDT MI1B2T ODL
A.3.24 MISR ESDT MI1B2E ODL
A.3.25 MISR ESDT MIRCCM ODL
A.3.26 MISR ESDT MI1B1 ODL
A.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)

A.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_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=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_GRANULES_REQUIRED = 1
   MAX_GRANULES_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
```

A.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
```

A.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
```

A.3.4 MISR ESDT MISL0AN ODL

```
DATA_TYPE_NAME = "MISL0AN"
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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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 Reseach 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
```

A.3.28 MISR ORBIT ODL

```
PLATFORM = "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
```

A.3.29 MISR PATHMAP ODL

```
PLATFORM = "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
```

A.4 Typical Terra MODIS PGE & ESDT ODL Files

Listings are provided for the following MODIS ODL files:

A.4.5 MODIS ESDT MOD03 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)

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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"
```

A.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"
```

A.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"
```

A.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"
```

A.5 Typical AIRS PGE & ESDT ODL Files

Listings are provided for the following AIRS ODL files:


```
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)
```

A.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 = "O003"
   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 = "O004"
   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 = "O005"
   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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.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
```

A.6 Typical Aqua MODIS PGE ODL File

Listings are provided for the following MODIS ODL files:

A.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