205-CD-002-006

EOSDIS Core System Project

Science User's Guide and Operations Procedure Handbook for the ECS Project, Volume 4:

Software Developer's Guide to Preparation, Delivery, Integration and Test with ECS

Revision 2

February 1998

Raytheon Systems Company
Upper Marlboro, Maryland

Science User's Guide and Operations Procedure Handbook for the ECS Project, Volume 4:

Software Developer's Guide to Preparation, Delivery, Integration and Test with ECS

Revision 2

February 1998

Prepared Under Contract NAS5-60000
CDRL Item #025

RESPONSIBLE OFFICE

R. J. Plante /s/	2/13/98
Robert Plante, Director of Science Department EOSDIS Core System Project	Date

SUBMITTED BY

Patrick M. O'Connell /s/	2/15/98
Patrick O'Connell, Project Manager EOSDIS Core System Project	Date

Raytheon Systems Company
Upper Marlboro, Maryland

This page intentionally left blank.

Preface

This document is intended as a guide to the developers of Science Data Production Software (SDPS/W) to be run using the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) Processing Services of a Distributed Active Archive Center (DAAC). This guide is the fourth volume of the *Science User's Guide and Operations Procedure Handbook for the ECS Project.* Volumes 1-3 describe the ECS Client and how to access the ECS data services.

A broad overview of the ECS Planning and Data Processing Subsystems is provided, along with a discussion of several issues concerning the design, development and delivery of SDPS/W to the DAAC. References to the essential and background documentation are provided, and directions on how to obtain these materials.

Revisions to this document introduce new material which may be essential to the development of the science software. To assist the science software developer in identifying changes which are relevant to his/her work, the following major changes are highlighted:

Section	Title	Change
1. Introduction		
Section 1.3	Science Data Production in EOSDIS	Clarification on running non-standard production versions
4. The Processing Environment		
Section 4.1.7	What is "Mode Management?"	Description is updated, mechanism employed for running test mode PGEs
5. Developing Science Software		
Section 5.3.1.2	Optional Tools	Toolkit changes include the addition of Digital Elevation Model (DEM) tools
Section 5.2.4	Scripts	Limitation on use of standard input and output is added
Section 5.2.6	Log Files	Clarification on log file contents
Section 5.5.3	Granule Level Metadata	ECS-supplied ESDT Descriptor referenced
Section 5.5.4	Services	Important information on subsetting is added
6. Delivery of Science Data Production Software		
Section 6.3.2	Summary Information (Delivery Contents)	List of PGE exit codes and associated messages to be provided

8. Developer Interaction with Operational Science Software

Section 8.4	QA/QC of Data Products	Revised detailed description per current strategy and ECS support mechanism
Section 8.5	Changing Production Scheduling Information	Hight-level description of Production Rules added

The following changes were made per Government request in the letter dated 08 January 1998, Subject: - NAS5-60000; DID 205-CD-002-006, Science Users' Guide and Operations Procedures Handbook, Volume 4: Software Deveoper's Guide to Preparation, Delivery, Integration and Test, Revision 2, October 1997 .

In order to aid the reader in locating this set of changes, their locations are indicated with change bars.

Page	Section	Title	Change
4-11	4.4	Quality Assessment	Term "validation" is replaced by product QA
4-16	Table 4-1	SSI&T Tool Capabilities	Term "kftp" is replaced by "ftp"
5-10	5.5.3	Granule Level Metadata Paragraph 3	Operational scenario for generation of MCFs by ECS is corrected.
5-11	5.5.3 Paragraph 2	Granule Level Metadata	Operational scenario for generation of MCFs by ECS is corrected.
8-4	8.7	Metadata Updates	Corrected to indicate generation of MCFs by ECS.

This document, as a formal contract deliverable with an approval code 1, required Government review and approval prior to acceptance and use. It was reviewed and approved, with comments, per GSFC Code 505 contracts letter dated March 19, 1997. Comments received with the approval letter have been incorporated, and this document is now considered accepted for use; no further review is required.

Contractor changes to this document are handled in accordance with Class I and Class II change control requirements described in the EOS Configuration Management Plan. Any changes shall be made by document change notice (DCN) or by complete revision.

This document is under ECS Project Configuration Control. Any questions or proposed changes should be addressed to:

Data Management Office
The ECS Project Office
Raytheon Systems Company
1616 McCormick Drive
Upper Marlboro, MD 20774-5372

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Revision 2
iii through xiv	Revision 2
1-1 through 1-4	Revision 2
2-1 and 2-2	Revision 2
3-1 through 3-4	Revision 2
4-1 through 4-16	Revision 2
5-1 through 5-14	Revision 2
6-1 through 6-10	Revision 2
7-1 through 7-8	Revision 2
8-1 through 8-4	Revision 2
A-1 through A-4	Revision 2
B-1 through B-4	Revision 2
C-1 through C-8	Revision 2
AB-1 through AB-4	Revision 2
GL-1 through GL-4	Revision 2

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
193-205-SE1-001	Preliminary	August 1993	
205-CD-002-001	Pre-approval	January 1995	
205-CD-002-002	Final	August 1995	95-0594
205-CD-002-003	Revision 1	February 1997	96-1376
205-CD-002-004	Revision 1	April 1997	97-0547
205-CD-002-005	Revision 2	October 1997	97-1411
205-CD-002-006	Revision 2	February 1998	

This page intentionally left blank.

Contents

Preface

1. Introduction

2. Related Documentation

3. ECS Processing Overview

4. The Processing Environment

5. Developing Science Software

6. Delivery of Science Data Production Software

7. Overview of the Integration with ECS and Acceptance Tests

8. Developer Interaction with Operational Science Software

Figures

Tables

Appendix A. Getting Assistance

Appendix B. Checklists

Appendix C. Annotated Outlines for Deliverable SDPS/W Documentation

Abbreviations and Acronyms

Glossary

1. Introduction

1.1 Identification

The successful integration of science algorithms into the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS) is a critical activity in the EOSDIS Project. This document contains information necessary for developers of Science Data Production Software (SDPS/W) to successfully integrate their software into the ECS.

This document is submitted as required by Contract Data Requirements List item 025, Data Item Description 205/SE1 of the ECS Contract (NAS5-60000).

1.2 Scope

This document is intended as a guide to the developers of SDPS/W to be run using the operational ECS Processing Services of a Distributed Active Archive Center (DAAC). A broad overview of the ECS Planning and Data Processing Subsystems is provided, along with a discussion of several issues concerning the design, development and delivery of SDPS/W to the DAAC. References to the essential and background documentation, including resources on the World Wide Web (WWW) are provided along with directions on how to obtain these materials.

This guide is the fourth volume of the *Science User's Guide and Operations Procedure Handbook for the ECS Project.* Volumes 1-3 describe the ECS Client and how to access the ECS data services.

1.3 Science Data Production in EOSDIS

It is the goal of EOSDIS to provide end-to-end services from EOS instrument data collection to science data processing to full access to EOS and other Earth science data holdings. NASA has established several Distributed Active Archive Centers (DAACs), chosen based on expertise in specific disciplines (indicated in Figure 1-1) and demonstrated long-term commitments to the corresponding user communities. The DAAC acronyms are defined in the list of Abbreviations and Acronyms. [Not shown in Figure 1-1 is the Socio-Economic Data and Applications Center (SEDAC).]

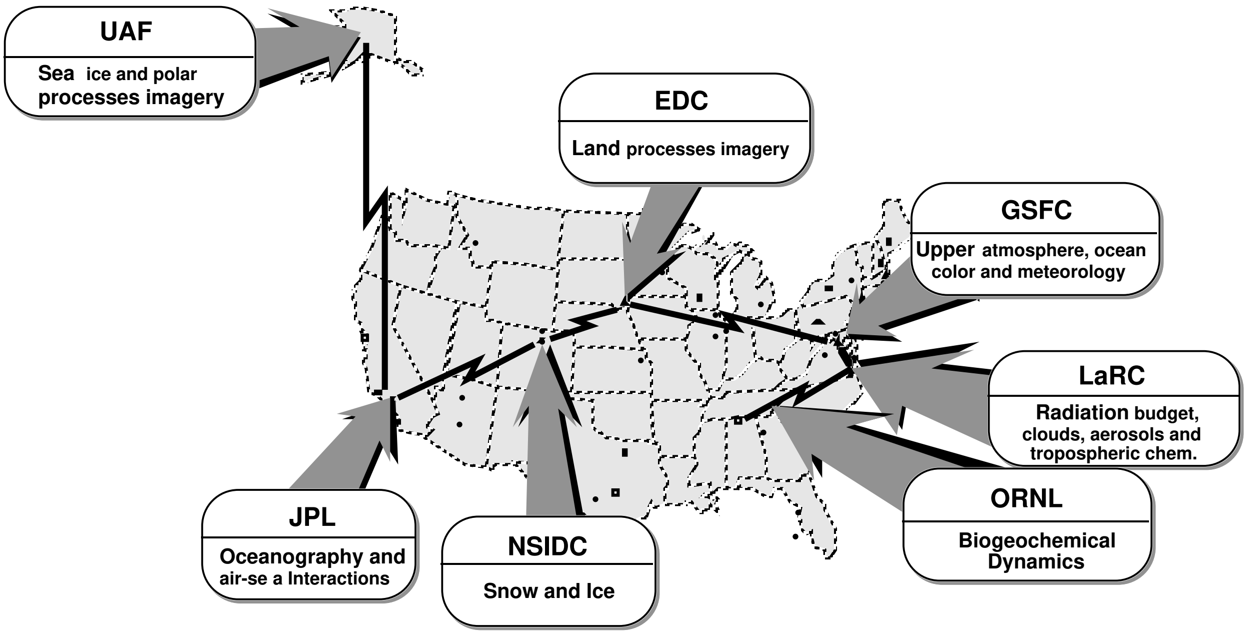

Figure 1-1. Locations of Distributed Active Archive Centers

The DAACs provide the facilities, the management and operations support for the production, archive, and distribution of EOS Standard Products. At the DAACs, users can expect a level of service which would be difficult to maintain in a single data center attempting to serve the extraordinarily wide range of scientific disciplines encompassed by the EOS program. A user interacting with any given DAAC can access data from all the DAACs.

In addition, Science Computing Facilities (SCFs) located at EOS investigators' home institutions are used to develop and maintain scientific algorithms and software, calibrate the EOS instruments, validate algorithms and products, generate Special Products, provide data and services to other investigators, and analyze EOS and other data in pursuit of the Mission to Planet Earth (MTPE) science objectives. The SCFs are established directly by the EOS investigators, and may consist of a single workstation or a large data center. Software toolkits are provided to the SCFs and other users to facilitate data access, transformation and visualization, and for algorithm development.

Several types of SDPS/W can be run at a DAAC. These include:

- Standard Product Generation, with the product archived
- Environmental Monitoring, i.e., not a standard product but an alert message

  (e.g., a volcanic eruption)
- Specific Processing, e.g., in support of a field experiment
- User Methods, e.g., non-standard subsetting of an archived product

Approval must be obtained for the use of DAAC processing services. The nominal approval hierarchy is indicated in Table 1-1.

Table 1-1. Approval Authority for DAAC Product Processing Services

SDPS/W Type	Approval Authority
Standard Products	EOS Program and Project Scientists
Environmental Monitoring	EOS Project Scientist
Specific Processing	EOSDIS Project Scientist
User Methods	DAAC Manager

The SDPS/W may run in one or more versions:

- Production Version (i.e., product is publicly available)
- Engineering or Test Version (i.e., algorithm not validated, output may not be publicly available but is temporarily saved or sent to investigator)

    This version may be run sporadically or run on every granule; the code may initially be inefficient, becoming more efficient as performance tuning takes place.

    The method by which test versions of products will be created will be through a "test/engineering" mode using Mode Management.

    All products are archived, but would be archived in a test mode to a separate test archive. A test archive would not have the persistence of the production archive, nor would it have the subscribability for the public. A test product would be released only at the investigator's discretion to another investigator.

    For a "short-term" product, there is a choice on whether to run in production mode or in a test mode. The choice is made based particulars of the situation and on whether or not the product is to be made publicly available.

- Special SDPS/W (i.e., short-term production, output may or may not be permanently archived)

As the scientific algorithms mature, it is likely that reprocessing of the earlier observations will be necessary in order to maintain consistency within the long-term data set. The approval to undertake the reprocessing of an entire data set involves both operational and scientific issues, especially in cases where interdependencies on products from different instruments are involved. In the case where a reprocessing request involves an entire data set, review and approval may be required from the EOS Program and EOSDIS Project. Routine reprocessing by the DAAC of specific data files in response to operational problems, however, is a normal operational function and does not usually need specific approval.

1.4 Getting Started: A Quick Guide

The essential documents to the development of SDPS/W and how to acquire them are listed in Section 2. An overview of ECS is given in Section 3, and Section 4 describes the ECS science data processing environment in greater detail. Section 5 discusses issues related to the development of SDPS/W. Procedures for getting assistance are described in Appendix A.

The procedures to follow when delivering SDPS/W to the DAAC are delineated in Section 6. An overview of the steps in integrating the SDPS/W with ECS is given in Section 7, along with a brief description of the acceptance testing. Checklists for the various steps in planning, developing, integrating with ECS, and testing can be found in appendix B. Annotated outlines for the documentation to be delivered with the SDPS/W are suggested in appendix C.

Following installation of SDPS/W at the DAAC, the SCF will often have continued interactions with the operational science software. These are briefly described in Section 8.

2. Related Documentation

Current submittals of all ECS project documentation are available on the World Wide Web via the ECS Data Handling System at http://edhs1.gsfc.nasa.gov. Hard copies or archived copies may be obtained from the ECS Project Office.

2.1 Parent Documents

The parent document is the document from which this Software Developer's Guide to Preparation, Delivery, Integration, and Test with ECS' scope and content are derived.

194-207-SE1-001	System Design Specification for the ECS Project

2.2 Applicable Documents

The following documents, referenced within and directly applicable to this Guide, are likely to be essential to developers of the SDPS/W.

333-CD-004-001	Release B.0 SCF Toolkit Users Guide for the ECS Project
194-815-SI4-001	SDP Toolkit Primer for the ECS Project (available in electronic format only at http://newsroom.hitc.com/sdptoolkit/primer/tkprimer.html)
175-WP-001-001	HDF-EOS Primer for Version 1 EOSDIS, White Paper for the ECS Project
420-WP-006-002	Establishing Science Software Exit Conditions for the Production Environment, White Paper for the ECS Project
420-TP-005-002	Preliminary User View of Release A Data, Technical Paper for the ECS Project
423-16-01	Goddard Space Flight Center, Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, 10/96
445-WP-001-002	Production Rules, White Paper (draft available on request)
505-41-33	Goddard Space Flight Center, Interface Control Document (ICD) Between EOSDIS Core System (ECS) and Science Computing Facilities (SCF)
170-TP-007-001	Writing HDF-EOS Grid Products for Optimum Subsetting Services, Technical Paper for the ECS Project
170-TP-008-001	Writing HDF-EOS Point Products for Optimum Subsetting Services, Technical Paper for the ECS Project
170-TP-009-001	Writing HDF-EOS Swath Products for Optimum Subsetting Services, Technical Paper for the ECS Project

2.3 Information Documents

2.3.1 Information Documents Referenced

The following documents are referenced herein and amplify or clarify the information presented in this document. These documents are not binding on the content of this Guide. Working iterations of these documents will be made available to the user community as they are drafted.

162-WP-001-002	Operations Concept for Integration and Test of Science Data Production Software, White Paper for the ECS Project
152-TP-001-003	ACRONYMS, EOSDIS Core System (ECS) Project
World Wide Web	EOS Instrument Team Information Page at http://ecsinfo.hitc.com/iteams/
	Metadata in EOSDIS http://ecsinfo.hitc.com/metadata/metadata.html
	The Population Process for ECS Metadata in Release A at http://ecsinfo.hitc.com/metadata/tp420.html

2.3.2 Information Documents Not Referenced

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

505-41-12	Goddard Space Flight Center, Interface Requirements Document between EOSDIS Core System (ECS) and Science Computing Facilities (SCF)
NP-215	Goddard Space Flight Center, 1995 MTPE (Mission to Planet Earth) EOS Reference Handbook PDF softcopy of this document is available at http://eospso.gsfc.nasa.gov/eos_reference/handbook95.pdf
World Wide Web	ECSinfo [EOSDIS Core System Information for Scientists] at http://ecsinfo.hitc.com/
	PGE Design Information Page at http://ecsinfo.hitc.com/iteams/Science/pge_wp.html
	EOSDIS Core System Project SDP Toolkit Home Page at http://newsroom.hitc.com/sdptoolkit/toolkit.html
	ECS Frequently Asked Questions at http://ecsinfo.hitc.com/iteams/faq/
	Metadata in EOSDIS at http://ecsinfo.hitc.com/metadata/metadata.html

3. ECS Processing Overview

3.1 The Role of ECS

MTPE is a long-term, multi- and inter-disciplinary NASA research mission to study the processes leading to global climate change, and to develop a predictive capability for the Earth system on time scales of decades to centuries. MTPE comprises three major components:

- The EOS collects Earth science data, with emphasis on long-term, sustained data sets from carefully calibrated instruments on satellites in low Earth orbit.
- The EOSDIS provides the Earth science community with easy, affordable, and reliable access to EOS and other Earth science data.
- It provides an integrated scientific research program to investigate processes in the Earth system and use this information to improve predictive models.

The EOSDIS provides a broad community of users with a unique resource for enhancing their understanding of global change issues and for acquiring data for use in other applications. The EOSDIS user community includes not only NASA-supported researchers, but other scientific, educational, resource management, and policy communities.

The EOSDIS Core System (ECS) is a major component of the EOSDIS. ECS will control the EOS spacecraft and instruments, process data from the EOS instruments, and manage and distribute EOS data products and other selected data sets. The essence of the data flows involved in end-to-end transmission of EOS data is shown in Figure 3-1 and is described briefly below:

In general, NASA EOS satellites transmit their instrument data in a telemetry stream through the Tracking and Data Relay Satellite System (TDRSS) to the receiving station at White Sands, New Mexico. The data are then processed by the EOS Data and Operations System (EDOS) to create the Level 0 "raw" instrument data. It is planned that future missions beyond AM-1 will use the EOS Polar Ground Stations (EPGS) to replace TDRSS in the primary telemetry relay role. TDRSS will then become a backup to EPGS.

Satellites operated by international partners, however, downlink directly to the appropriate International Partner Ground System (IPGS) via their receiving stations. Data from an IPGS are typically transferred directly to the DAAC via media or other electronic means.

The Level 0 instrument data are then distributed to the designated DAACs. These data centers house the ECS computing facilities and operational staff needed for product generation and to manage and store EOSDIS data, as well as the associated metadata and browse data required for effective use of the data holdings. When necessary, the DAACs exchange data via a dedicated EOSDIS backbone network (EBNET) to support processing which requires ancillary data produced at another DAAC. The DAACs also house systems for processing and/or storage of non-EOS Earth science data.

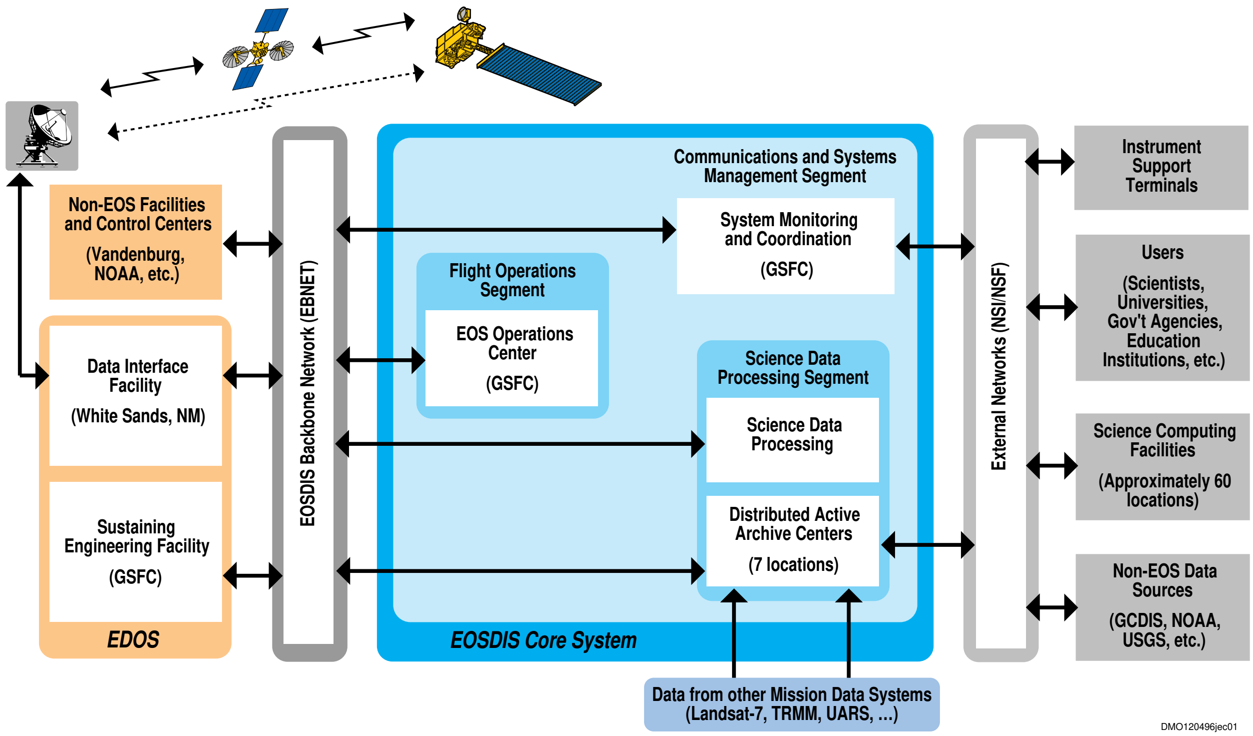

Figure 3-1. EOS Mission Science and Engineering Dataflows

EOS Flight Operations (including Spacecraft and Instruments) are conducted from the EOS Operations Center (EOC). Non-U.S. Instruments on U.S. Platforms are operated and monitored through International Partner (IP) Instrument Control Centers (ICC).

Most science users access EOS data products via external networks such as the NASA Science Internet (NSI) or the National Science Foundation (NSF) Internet. Open access to the EOS data by all members of the science community distinguishes the EOS from previous research satellite projects, where selected investigators have had proprietary data rights for a number of years after data acquisition.

Generally, a "Standard Product" is generated as part of a research investigation, of wide research utility, accepted by the Investigator Working Group (IWG) and the EOS Program Office, routinely produced at a DAAC, and typically spatially and/or temporally extensive. SCFs located at EOS investigators' home institutions are responsible for developing, validating and maintaining the science algorithms and software for the standard products. Instrument teams are also responsible for the calibration of the EOS instruments.

The SCF may generate data products at their facility as part of a research investigation. These may be produced for a limited region or time period. These "Special Products" can be transferred to a DAAC for archive and dissemination.

3.2 Functional Description of ECS

ECS is composed of three segments (see Figure 3-2), defined to support three major operational areas: flight operations, science data processing, and communications/system management. A brief description of the ECS segments follows. (For more information, see the *System Design Specification for the ECS Project*.)

- The Flight Operations Segment (FOS) manages and controls the EOS spacecraft and instruments. The FOS is responsible for mission planning, scheduling, control, monitoring, and analysis in support of mission operations for U.S. EOS spacecraft and instruments. The FOS also provides investigator-site ECS software (the Instrument Support toolkit) to connect a Principal Investigator (PI) or Team Leader (TL) facility to the FOS in remote support of instrument control and monitoring. PI/TL facilities are outside the FOS, but connected to it by way of the EBNET or the NSI/NSF Internet.

- The Science Data Processing Segment (SDPS) receives, processes, archives, distributes and manages Level 0 data and higher level products from EOS and other NASA Earth Probe flight missions. It provides support to the user community in accessing both Standard Products and products resulting from research activities that utilize these EOS data (e.g., special products). SDPS also promotes, through advertisement services, the effective utilization and exchange of data within the user community.

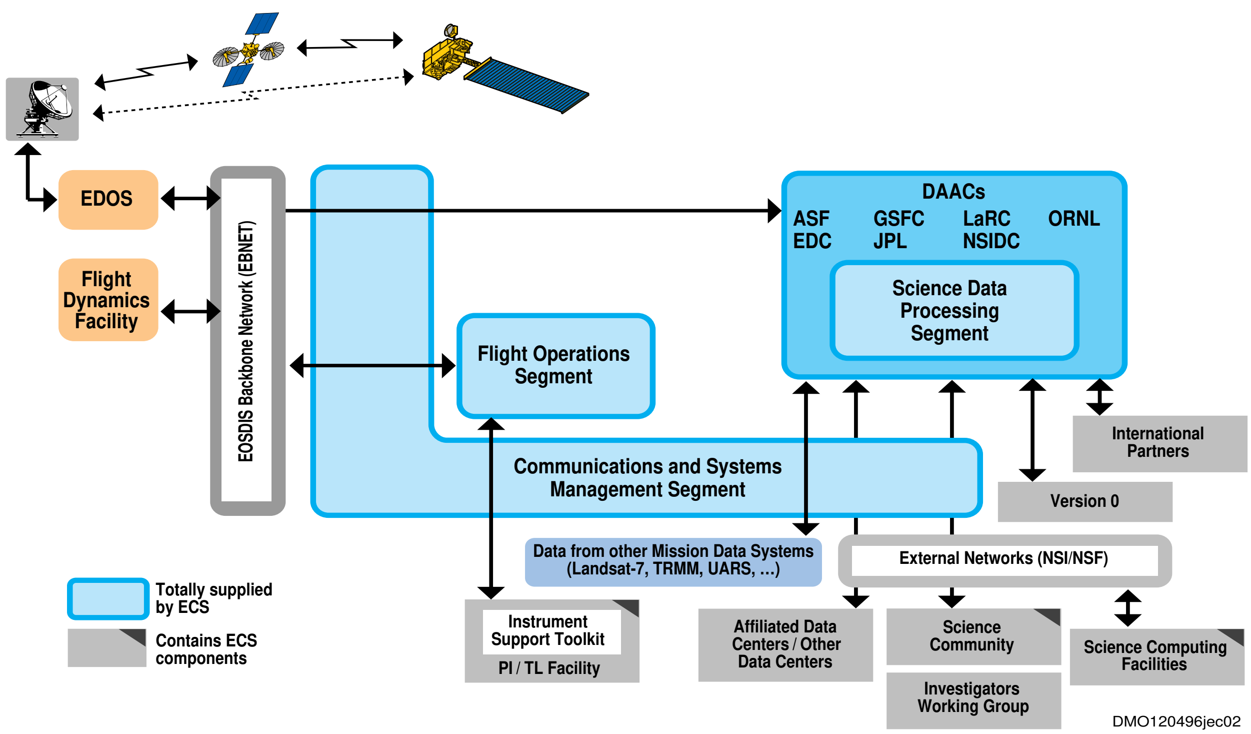

Figure 3-2. ECS System Architecture Context Diagram

- The Communications and System Management Segment (CSMS) consists of the system components involved with the interconnection of user and service providers and with system management of the ECS components. The CSMS is composed of the Communications Subsystem (CSS) and Management Subsystem (MSS). The MSS includes decentralized Local System Management capabilities at the individual DAAC sites and the separate System Monitoring and Coordination Center (SMC). The SMC provides system management services for the EOS ground system resources.

3.3 Long-term Obligations of Science Data Production Software Developers

Investigators developing SDPS/W assume obligations which extend well after the time the software is placed into production at a DAAC. These may include:

- Maintaining instrument calibration over the life of the instruments and archive information related to calibration.
- Maintaining the SDPS/W and associated documentation. Software maintenance involves modification of existing operational software for updates resulting from changed functional specifications, corrections to processing or implementation failures, adaptations to changes in the processing or data environments, and enhancing performance or maintainability. It is recommended that the software maintenance be done within the context of a change control process.
- Assessing the quality of the standard products
- Assisting DAAC operations in solving problems related to the operation of SDPS/W
- Maintaining the tables and coefficients required by the SDPS/W
- Providing expert consultation to the DAAC with regard to the standard products

4. The Processing Environment

4.1 Concepts

The purpose of this section is to describe some of the concepts that are important for ECS processing. Two major players in science software development and product generation are the SCF and the DAAC, and their roles and responsibilities are discussed. The primary interfaces between the DAAC and other sites are described. The top-level planning and processing operations concept is explained.

In addition, the ideas of "strings", "PGEs", and "subscriptions" are discussed. These concepts are fundamental to understanding how science product generation is done. In general, (1) science software is packaged into Product Generation Executives, or **PGEs**, (2) the processing hardware on which the science software runs is organized into sub-networks called **strings**, and (3) science software processing is scheduled in response to **subscriptions**, submitted by the DAAC operations personnel (and others), for the data products that are generated by the science software.

4.1.1 Roles and Responsibilities

The **developer** is the person or team who actually develops the science software. The developer may reside at the SCF, or may deliver software to the SCF for integration with other software before delivery to the DAAC.

The roles and responsibilities of a specific science software developer and the DAAC at which the science software will be executed need to be discussed between those parties, and an agreement documented (see Section 5). In general, however, the developer and/or the SCF will be responsible for:

- Developing the science software and delivering it to the DAAC.
- Assisting the DAAC in planning for production readiness, including the integration and test of the science software.
- Supplying documentation (see Section 6) of the SDPS/W to the DAAC for production, to serve as an archived record in case maintenance responsibilities later are transferred to someone else, and as information for users to determine that the data meets user requirements.
- Supplying the mandatory core and product specific metadata (see Section 5.5)
- Supplying test cases and input test data and expected test output to the DAAC in order to verify that the software runs correctly in the operational environment.
- Participating in the integration and test of the science software at the DAAC, in particular the evaluation of the results of integration and test.

- Making corrections to the science software that are indicated by integration and test at the DAAC.
- Validating the algorithms and data products.
- Assessing the quality of the data products.
- Enhancing the science software.

Nominally, the DAAC will be responsible for:

- Integration and test of the science software, in order to verify that the software will run safely, i.e., will not interfere with other software or DAAC operations. The integration and test activity will also allow the SCF to verify that the software will run correctly (i.e., produce scientifically correct results) in the production environment.
- Archiving all source files, documentation, test information, and other files associated with the science software.
- Supplying the operational environment for the science software and running the software in an ongoing production mode to generate the data products.
- Maintaining the science software in response to evolving ECS hardware and/or software environments. Changes to science software will require prior approval from the developer.
- Archiving and distributing the data products.
- Distributing source files, documentation, test information, and other files associated with the science software to authorized users.
- Accepting, archiving, and distributing special products generated by other facilities.
- Advertising data and providing user support.

4.1.2 Interfaces

What follows is not an exhaustive list of the interfaces between the DAAC and other sites. Rather, it is intended to serve as a general description of the interfaces that are especially relevant to science processing.

- Between the DAAC, and the SCF and/or developer: The SCF/developers assemble the PGE source code, control files, makefiles, documentation, scripts, test input data, comparison data and other related files into Delivered Algorithm Packages (DAPs). After notification and response, they send the DAPs to the DAAC. The DAAC receives science software deliveries from the SCF/developer. For each type of science data input and science data output product, the Instrument Teams are responsible for defining an Earth Science Data Type (ESDT), which includes supplying collection and granule level metadata. The DAAC and the SCF/developer work together during integration and test of the science software. During production, the DAAC may send data products to the SCF/developer for quality assurance and receive back quality assurance flags. [See the *ICD Between ECS and Science Computing Facilities (SCF).*]

- Between the DAAC and EDOS: The DAAC receives Level 0 data from EDOS.
- Between the DAAC and other DAACs: The DAAC receives data products generated at other DAACs which it needs as inputs to its own processing. The DAAC likewise distributes the data products which it produces to other DAACs which need those products as inputs.
- Between the DAAC and non-EOS data suppliers: The DAAC receives data products from non-EOS suppliers such as the National Oceanic and Atmospheric Administration (NOAA) when these products are required as inputs to the DAAC's processing. These data are called ancillary data (see the *ECS Glossary*).
- Between the DAAC and users: The DAAC receives requests for data products and other archived information from the users, and distributes the requested data. When SSI&T for a PGE has been successfully completed, the latest version of the PGE source code, control files, makefiles, documentation, and related files from the DAPs for this PGE are labeled with a version identification, assembled into a Science Software Archive Package (SSAP), and inserted on the Science Data Server (SDSRV). Users may also request the SSAPs. Note that version identification will be accomplished with ClearCase labels rather than normal ClearCase version numbers. ClearCase labels are attributes associated with particular ClearCase elements and offer more flexibility than standard versions *per se*.

4.1.3 Planning and Processing Concept

The subsystems which provide the processing environment are known as Planning and Data Processing (PDPS). As part of the Science Data Processing Segment of ECS, the primary job of the PDPS is to convert data production requests into data processing requests, execute the required science software to satisfy the data processing requests, and pass the resulting data products to the data server(s) for distribution and archiving. This process is illustrated in Figure 4-1.

A production request is a request to generate a particular data product. There are three kinds of production requests: (1) an on-demand production request, (2) a standard production request, and (3) a reprocessing production request.

An on-demand production request is an “unplanned” request to generate a particular data product, and is usually automatically initiated by a data server responding to a request for data from a user for a product that does not already exist. It may also come from the DAAC operations staff, for example, if a processing chain has been interrupted and needs to be restarted. Although on-demand requests are usually handled as soon as they are received, they may be deferred until the next time a production plan is created if they will cause too large of a perturbation in the expected resource usage.

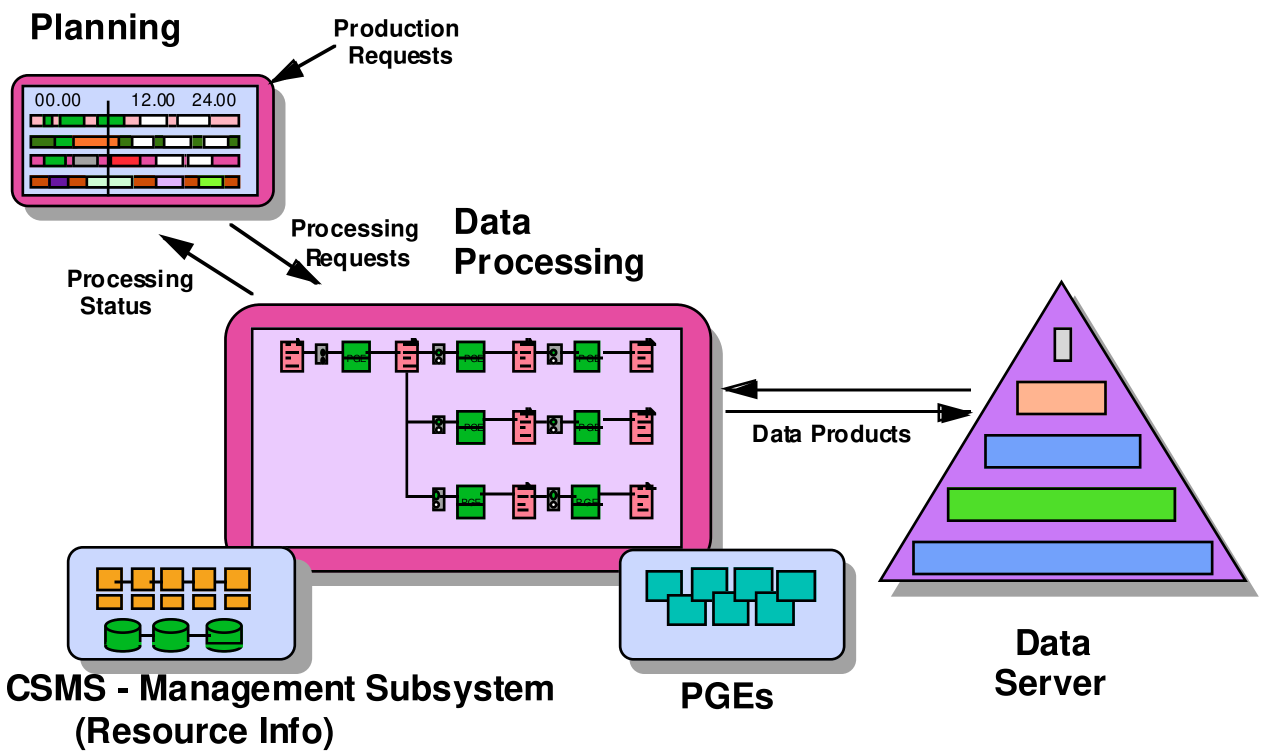

Figure 4-1. ECS Planning and Processing

A standard or reprocessing production request is a "planned" request to automatically generate a particular data product, and is initiated by the DAAC operations staff. A standard production request is for a product that is based on at least some "new" inputs. These input data might not yet exist.

A reprocessing request is the mechanism to generate a new version of a product based on inputs, normally using an updated version of the science software or updated calibration coefficients. A reprocessing request may have a major impact on the system, due to data dependencies. Reprocessing a product which is used to create many other products may necessitate a good deal of additional reprocessing. For this reason, reprocessing may need approval before it will be done. (See Section 4.5.)

Standard and reprocessing production requests, as well as deferred on-demand requests, are handled via a production plan. The production plan is created off-line by the DAAC operations staff, with the help of the planning software. Inputs to the planning process are: all of the planned production requests and deferred on-demand requests, information about the resources required to satisfy those requests, estimates for when required inputs will be available, resource availability information, and local DAAC production guidelines (defining priorities for different types of processing).

The resulting plan is a "schedule" of data processing requests which must be executed to satisfy the production requests. A data processing request is a request to execute a particular PGE with specific input and ancillary data to generate the designated products.

A production plan is created off-line as a "candidate plan", but does not affect actual processing until it is activated (at which point it is called the "active plan"). At any particular time, there may be any number of candidate plans, but only one active plan. Although the production plan is, in a sense, a "schedule", processing is triggered by availability of required inputs and resources, rather than occurring at a predetermined time.

4.1.4 What is a "String"?

The design of the processing environment utilizes a hardware "string" architecture to support the unique requirements of each site. In the string architecture, processing resources are configured into either dedicated or virtual sub-networks, or "strings", and each string supports some subset of the production or testing tasks for the site.

The goals of this approach are (1) to minimize contention for resources by different processing tasks and optimize network traffic; (2) to modularize the hardware architecture, so that hardware may be upgraded or added, with minimal impact to the rest of the system; and (3) to insulate ongoing product generation from other activities, such as SDPS/W integration and test or preventative maintenance.

A string is configured to support the particular processing and reprocessing requirements of the product generation tasks assigned to it. The choice of which tasks to assign to which strings is based, in part, on coupling and cohesion considerations: tasks which are tightly coupled (via data dependencies) are assigned to the same string, while tasks which are loosely coupled are assigned to different strings.

Some of the hardware resources support activities other than product generation activities, such as SDPS/W integration and test, production of prototype products, and training. These strings also serves as backup for the other strings, as well as providing additional processing resources when necessary.

4.1.5 What is a "PGE"?

Science software is packaged into one or several PGEs. Each PGE consists of one or more compiled binary executables and/or shell scripts, and is the smallest scheduled unit in PDPS processing. Conceptually, a PGE produces one or more standard or intermediate product files or completes a processing step on a data set.

In the production system, PGEs are scheduled automatically. They can also be initiated manually, outside of the production environment, as UNIX shell level commands. (For development purposes at an SCF, temporary scripts can be built which invoke one or more PGEs.) A PGE can contain an arbitrary amount of processing. However, the following design considerations should be kept in mind:

- Because PDPS schedules processing at the PGE level, there is a certain amount of overhead associated with multiplying the number of PGEs. Also, it is more difficult for the developer to fine-tune the ordering of processing between PGEs than within a PGE. Hence, it is undesirable to have numerous, small PGEs.
- On the other hand, PDPS only saves files at the end of a PGE and, in case of a failure, will restart at the beginning of the PGE. Therefore, it is also undesirable to have PGEs that are too large.

A suggested rule of thumb is to design each PGE to generate a limited set of products (perhaps only one). In addition to the output product(s), the PGE must generate the metadata, i.e., information which describes each product output file. Metadata generation is more straightforward when the number of output products is few.

A "quality assessment executive" (QAE) is a special kind of PGE whose purpose is to perform automatic quality assessment (QA) on the product outputs of a PGE. A QAE will update the metadata for the corresponding product files. QAEs will be included in the production planning.

4.1.6 What is a "Subscription"?

In general, a subscription defines an interest in changes in the capabilities or data offered by a service. An ECS user, for example, sends a Subscription Request to the DAAC to automatically notify the subscriber when data meeting specified criteria are available. When a subscription is triggered, the notification to the subscriber includes a universal reference (UR) to the data so that the recipient can easily retrieve it. “Standing Orders” could be created via subscription, for example, for data to support product QA.

For general access to the data products, the ECS Scientist’s Workbench is the mechanism by which a user will directly access the ECS services supporting subscriptions. These services include:

- The ability to inquire whether a subscription has been triggered.
- The ability to receive notification that a subscription has been triggered.
- The ability to specify the actions to be taken when a subscription has been triggered.

The *Science Users Guide and Operations Procedure Handbook (Release B.0)*, document 205-CD-004-001, provides more information.

For product generation, data subscriptions are established for each PGE for the input and ancillary data and for product quality assessment. In this manner, the PDPS is automatically advised concerning the availability of the data inputs.

4.1.7 What is "Mode Management"?

Mode management enables the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) Maintenance and Operations (M&O) Staff at the DAAC to perform testing and/or training activities while production activities continue uninterrupted. It addresses the planning, initiation, execution, monitoring, and control of various system activities. These activities include production (operations), testing, and training. Each unique activity that requires process and data separation is classified as a mode. Mode management enables the execution of multiple modes, such that each mode functions without interfering with the other modes and each mode maintains data integrity throughout execution. An example is testing a data server application within the same system that is supporting production activities. The test copy of the data server must not interfere/interact with the production copy of the data server on both the data and process levels. In addition, it will only see and have access to interface components that have been specifically set up and initiated under the same test mode. Science software which has successfully passed SSI&T and has been made operational, should only read test/training data sets in test mode. Data sets produced using test input data should not be archived with the operational production data sets.

The mode management design does not limit the number of concurrently executing modes. However, performance considerations need to be addressed prior to the initiation of an additional mode. The design will support multiple test and training mode instances, but due to concurrent data access and data preservation issues there can only be one operational mode of execution at any given time.

DAAC personnel responsible for resource management will have a view of all the components supporting each mode. This view is provided through HP OpenView and can be configured to display mode specific application and process-level components through the use of maps and submaps. Software components will be duplicated, and hardware resources will be isolated whenever possible, to support an additional mode. However, there will be shared resources, both hardware and software, that require special consideration to enable mode management support. Refer to Appendix A in the document *Transition To Releases B.0 and B.1*, 410-TP-010-004, April 1997 for additional details. There is also the *Mode Management White Paper*, N09530V3, November 1995, which is currently under revision.

4.1.8 What is an ESDT?

An ESDT is an Earth Science Data Type and is a representation of how the "*data are data*" paradigm is managed within the ECS. An ESDT must be defined for every granule managed by the ECS. For each granule, the ESDT defines collection level metadata, granule level metadata, and the services that can be performed on that granule. Collection level metadata define the characteristics of the data collection to which a granule belongs. Granule level metadata define the characteristics that are unique to that granule. Services define what ECS can do with the granule. Standard services include Insert (inserting the granule into the Data Server) and Acquire (getting the granule back from the Data Server). More advanced services include subsetting and subsampling. See section 5.5 for more details on metadata and ESDTs. See Section 5.5.4 for information on how to organize products to use ECS standard subsetting services.

4.2 Science Data Processing within ECS

4.2.1 Putting Science Software into Production

Once science software has passed acceptance testing at the DAAC and is ready to be run operationally, the planning databases are updated with the information that the system needs to be able to run the software. This includes information (for each PGE) such as: required inputs and expected outputs; memory, disk space, and nominal CPU time requirements; and which platforms it can run on.

Additionally, the use of metadata based activation rules, exit condition rules, and other PGE specific production rules will be supported by the Production System. PGEs which take advantage of these methods have additional information passed to the Planning database.

Information about science software production constraints and PGE production rules are incorporated into the local DAAC production guidelines, as well as into subscriptions established by the DAAC operations personnel. For example: a particular instrument's Level 1 processing is to be run whenever new Level 0 data are received by the DAAC; or another product is to be generated only once a month on or after a particular date; or a third product is only to be generated in response to an on-demand request. Further information on PGE production rules can be found on the *EOS Instrument Team Information Page*. An updated *Production Rules White Paper* (draft) is also be available as of June 1997.

4.2.2 Ancillary Data Preprocessing

Ancillary data required by certain science software may require preprocessing at the DAAC before it is used as an input by the SDPS/W. Preprocessing software will be supplied either by the SCF or by ECS, depending on the particular case. The usual rule is: if the preprocessing is required by multiple development teams, then ECS will supply the required software; if it is required by only one team, then that team will supply it along with the rest of its science software.

ECS-supplied preprocessing will never change the science content of the ancillary data. For example, while ECS preprocessing may reformat an ancillary data set, or may subset out certain parameters, it will never do things like change units, regrid the data, or average to fill in gaps.

Preprocessing software supplied by the SCF must be packaged into PGEs in the same fashion as the science software, and the appropriate databases and production guidelines will contain the same sort of information as for the science software PGEs and products.

Preprocessing software will, in some cases, run when a new ancillary data set is received by the DAAC, as part of the ingest processing. In other cases, it will run immediately prior to the PGE that requires it. The approach used in a particular case will depend on resource tradeoffs, and needs to be decided jointly by the developers and the DAAC.

4.2.3 Executing the Science Software

The science software is executed in response to either the production plan or the receipt of an on-demand production request.

A PGE in the production plan will be started by PDPS when the initial conditions specified in the plan have all been satisfied. In most cases, this means that all of the required inputs for the PGE exist and are present at the local DAAC, and that the required processing resources are available. Other types of initial conditions may be specified as well, in some cases: for example, the plan may specify that the PGE is not to be run until after a particular date, or the plan may indicate certain priorities that must be followed.

Upon receipt of an on-demand production request, PDPS translates the request into a list of PGEs which must be executed in order to satisfy the request. If the additional load on system resources is not too high, required inputs are requested (as necessary) and the PGEs are started when the inputs become available. Otherwise, the on-demand request is added to the planning database and is delayed until the next time a production plan is created.

Upon completion of a PGE, PDPS sends the output data products to the data server for storage and distribution.

4.3 Configuration Management

Configuration management (CM) deals with identifying, controlling, reporting and auditing hardware, software and documentation. Once baselined at the DAAC, no changes can be made without the approval of all responsible parties.

The benefits of CM of science software are to provide a definitive current “bill-of-materials” for the production software system at the DAAC and to maintain a record of previous versions of the science software system and test configurations. For any product generated at any given time, the exact configuration of science software, coefficient files, scripts, documentation, and control files is known and can be reconstructed.

ClearCase®, by Atria Software, Inc., is the CM tool provided with the ECS to the DAACs. ClearCase provides version control of objects including source code, binaries, executables, documentation, test suites, and libraries in heterogeneous UNIX development environments. A versioned object base (VOB) is created, consisting of a UNIX directory tree containing a database and three storage pools. The VOB is used to manage multiple versions of objects (e.g., source code, coefficient files, documentation), track which versions of software routines were used in software builds of individual programs and coefficient files used in production, maintain an annotated history of changes, allow recreation of past configurations, etc. This can be accomplished through either a graphical user interface or a traditional UNIX command line interface.

It is important to note, however, that (a) ECS does not impose any CM requirement on a SCF, and (b) ClearCase will not be provided to a SCF by ECS. The choice of a CM tool for use by a SCF should be guided by the requirements for the CM of the science software development and the available resources. There may be some efficiency in the SCF and DAAC using the same tool, but this is not required for good CM.

The SDPS/W, documentation and coefficient files, however, will be placed under CM at the DAAC when a delivery is made (see Section 6.4). Coordination with the DAAC on CM issues prior to a delivery is encouraged.

4.4 Quality Assessment

The assessment of product quality is likely to occur at several stages. First, there may be automated assessments that are made during the PGE execution at the DAAC or by means of a QAE. There can also be a manual assessment performed at the DAAC, as the SCF may direct. The DAACs will be able to update the operations QA flag in the metadata. QA at the SCF may be either manual or automatic. The user may also provide an assessment of the quality of a product.

The capability for QA is provided for every standard product PGE. If available, an automated QAE is run as part of the processing chain, normally following execution of the corresponding PGE.

Manual QA is controlled via subscriptions, i.e., the QA personnel at the SCF or DAAC register a subscription to receive the output product to be examined, specifying under what conditions they wish to QA the data. Once the subscription is triggered, the QA user is notified and may retrieve the data. It is recommended that manual checks include data that are randomly distributed over time and geographical areas in order to detect unanticipated anomalies. The SCF likely will also QA data which fail preliminary QA checks at the DAAC for unknown reasons.

Any product requiring manual QA, at either the SCF or the DAAC, is tagged as unvalidated, but is stored immediately and made available for distribution per existing subscriptions and subsequent requests. (Subscribers can specify whether or not they want to wait for product QA.)

4.4.1 QA at the DAAC

It is expected that the bulk of DAAC QA will consist of automated QAEs which are supplied by the developers and which run as part of the normal processing chain; and that, initially, most manual QA will be done by the SCF, who are the experts on what the data should look like. Certain routine manual QA procedures may gradually migrate to the DAAC as they become well understood, preferably in the form of automatic QAEs. In summary:

- Automated QA may be done as part of the PGE, in a separate QAE or in both.
- Manual QA follows any automated QA. It is performed by the DAAC quality assurance monitor who performs any manual QA procedures prescribed in the QA policies and procedures manual provided by the developer.
- It is also possible that, over time, portions of the QA procedures are understood well enough to be encapsulated in an automated system.

- The product metadata includes quality assurance codes for each stage of the QA process: any QA within the PGE, any results of running a QAE, the evaluation of the QA monitoring staff, and any other steps that are required. Separate QA codes ensure that a complete QA history is maintained within the product metadata.

4.4.2 QA at the SCF

QA at the SCF will normally involve the following steps on data products specified by the SCF, or on data containing anomalies that may be discovered by the DAAC or users:

- The product, accompanying metadata, and any output QA files are made available to the SCF, either by "pushing" it to the SCF or by notifying the SCF of the availability of the data so that they could "pull" it from the DAAC. The method of accessing data (either "push" or "pull") is specified in the subscription.
- The developers evaluate the product at the SCF.
- SCF recommends updates to the QA flags in the product metadata using a GUI. Following DAAC approval, the recommended QA updates are committed in the Data Server databases.

4.5 Reprocessing

As the scientific algorithms mature, it is likely that reprocessing of the earlier observations will be necessary in order to maintain consistency within the long-term data set. The EOS system supports the reprocessing of standard data products, their browse data products and metadata. Reprocessing is driven by a reprocessing request which can be generated in the event of the availability of improved input data, new/improved calibration data, and/or algorithm updates.

Note, however, that production requests may be resubmitted during routine operations for a variety of reasons to “reprocess” individual data input files. These situations are governed by the individual DAAC procedures, and do not require extensive review or impact assessment.

The approval to undertake the reprocessing of a large volume of data involves both operational and scientific issues, especially in cases where interdependencies on products from different instruments are involved. In the case where a reprocessing request involves an entire data set, review and approval may be required from the EOS Program and EOSDIS Project.

Approved reprocessing of data may (depending on their assigned priority) be scheduled to fill slack time slots on computers of similar architectural design during lulls in initial standard processing to insure that optimum utilization is made of available DAAC hardware resources. In brief:

- The reprocessing request is initiated by the instrument team responsible for the algorithm. The requester forwards the reprocessing request to the DAAC. The DAAC operations staff, working with stored accounting information, generate a cost estimate, a potential completion date/time, and a resource impact assessment and returns these to the requester.

- The requester makes the decision to proceed with or to stop the request process based on the estimate and its accompanying information. If the decision is to proceed, the requester forwards the reprocessing request with the impact assessment to the EOSDIS Project Scientist.
- The EOSDIS Project Scientist grants or denies approval based on science considerations, the allocations available for either the requester or the allocations for the topic covered by the request. The approved reprocessing request is then incorporated into the production plan at the DAAC. If the reprocessing request can not be accommodated within existing resource envelopes, the DAAC Manager and EOSDIS project personnel will consider how best to accomplish the reprocessing.
- The operations staff will handle approved reprocessing requests by collecting them into batches. Each batch will be processed using "head of chain" order; products will be processed from lowest order to highest order to take advantage of data dependencies.

General information concerning reprocessed products is available to all users through the Advertising Service. However, if a user wants to be specifically informed about changes (like reprocessing) to a given product, they can place a subscription with the Data Server requesting notification in case of product change.

4.6 The Planning and Data Processing System

The Planning Subsystem (see Figure 4-1) supports the operations staff in developing a production plan based on a locally defined strategy, reserving the resources to permit the plan to be achieved, implementing the plan as data and processing requests are received. It also allows the site operations staff to negotiate on a common basis with other DAACs and EOSDIS System management if any change to their production plan causes conflict with other provider sites' plans (e.g., where dependencies between processing algorithms cannot be fulfilled). This subsystem provides the functions needed to pre-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 Data Processing Subsystem is responsible for managing, queuing executing, and monitoring processes on the processing resources at a DAAC. Requests for processing are submitted from the Planning Subsystem, which in turn have been triggered by data arrival or user request (Data Server) or through Planning itself (e.g., reprocessing). The Data Processing Subsystem provides the functions needed to execute all processing requests in a fully automated fashion according to a pre-planned schedule. The subsystem allocates processing resources to requests according to priority and within the scheduling parameters set by the plan. It stages and de-stages data, and re-allocates resources to adjust processing load. It provides interfaces to operations users to manage processes, schedules, resources, and resource allocations; and makes processing status and history available to operations and science users.

4.6.1 Planning

Planning provides the DAAC with the ability to create, modify, and implement a production plan. The production plan is generated by expanding production requests into individual data processing requests, using production rules (e.g., priorities, order), predicted resource availability, and predicted input data availability. Multiple candidate plans can be created for consideration, but only one plan is activated. Planning implements the selected production plan by submitting the data processing requests in the plan to the Data Processing Subsystem as required inputs become available. PGE execution status is recorded against the plan to assess the progress of data processing.

Production requests are entered by DAAC operations personnel to handle standard processing and reprocessing. These requests are maintained in a planning database and are included in the next candidate plan when it is generated. On-demand requests are accepted by Planning from the Data Server Subsystem. On-demand requests are either submitted immediately for processing or added to the planning database for inclusion in the next candidate plan, depending on whether certain predetermined criteria are satisfied.

Data processing requests are automatically submitted by Planning for processing when the required input data are available. For standard processing, input data may not be available when a plan is activated, for example, for the lowest level of processing ("Level 0"). Data is expected to arrive from EDOS during the plan's time frame. Planning is informed of the data availability via data availability notices.

Planning tracks the status of all production requests entered and all data processing requests generated. Management reports are generated (either periodically or upon request) that provide information concerning the planning workload and the status of requests processed.

4.6.2 Processing

Processing is responsible for initiating, managing, and monitoring of the generation of data products. A data product is generated through the execution of PGEs which are provided by the instrument teams. Processing supports the execution of a PGE by performing the following activities :

- Supports operations staff interfaces to monitor the processing environment.
- Interfaces with the Data Server Subsystem to stage data required by a PGE for execution.
- Allocates hardware resources, i.e. CPU, memory, and disk space, required by the PGE for execution.
- Interfaces with the Data Server Subsystem to de-stage the data generated by the execution of the PGE.

A request to generate an ECS data product is received from Planning in the form of a data processing request. A data processing request contains the information, such as input data identification, output data identification, priority, hardware resources, and so on, that Processing needs to execute the PGE. Generally, a processing job is related to the generation of Data

Products, but these jobs may include other types of processing, such as pre-processing of input data, quality assurance processing of generated Data Products, and possibly resource maintenance.

During PGE execution, Processing monitors the execution of the PGE and informs the operations staff and Planning of current status. Status may include current processing event history (what is happening, i. e. data staging, execution). Also, monitoring will be needed to make sure that the processing activity is executing properly. Upon completion of the execution of a PGE, Processing informs Planning and initiates the transfer of the generated data product(s) (if necessary) to the Data Server Subsystem.

4.6.3 Science Data Processing (SDP) Toolkit

The purposes of the SDP Toolkit (formerly the PGS Toolkit) are primarily (1) to provide an interface to the ECS system, (2) to allow science software to be portable to different platforms at the DAAC, and (3) to reduce redundant coding at the SCF.

The science software interface to the SDP Toolkit is the same in both the SCF and DAAC environments. The implementations differ in that the DAAC version of the SDP Toolkit is instrumented to interface with the MSS and CSS components of ECS, whereas these connections are absent in the SCF version of the SDP Toolkit.

The SDP Toolkit is divided into two groups: **mandatory** tools (see Section 5.3.1.1), which are required to be used in science software to accomplish the designated functions; and **optional** tools (see Section 5.3.1.2), whose primary intention is to reduce duplication of effort at multiple SCFs.

4.6.4 Science Data Pre-Processing

The purpose of Science Data Pre-Processing is to prepare science data, ancillary data and/or orbit/attitude data for subsequent use in product generation. The data are pre-processed either as it enters the system, as it is stored, or as it is being staged.

Pre-processing may take place in the Ingest, Data Server, or science data Processing subsystems. Which subsystem handles which pre-processing tasks will probably be determined on a data type by data type basis. The following general pre-processing tasks have been identified:

- Validate ingested data.
- Extract metadata from ingested data.
- Assess quality of ingested data.
- Subset data.
- Reformat data.

4.6.5 Science Software Integration and Test Tools

The Science Software Integration and Test (SSI&T) "Tools" have been assembled for use at the DAAC in order to facilitate the transition of the science software and user methods which have been developed outside of ECS into the operational environment of the DAAC. Most of the tools comprising the SSI&T environment, listed in Table 4-1, are commercial off-the-shelf (COTS) items, with a few custom ECS-developed tools to handle ECS specific issues.

Table 4-1. SSI&T Tool Capabilities

ECS Service	Capability	Tool
Data Ingest	Receive science software delivery	ftp, DCE
Management	Configuration manage delivered science software SSI&T problem tracking	ClearCase Sybase DDTs
Data Processing	Check for compliance to standards	ECS custom developed scripts, compiler switches, FORCHECK (for FORTRAN 77) Lint (for C)
	View documentation	WABI , MS Office [1], Ghostscript/Ghostview (for postscript), Netscape (for HTML) Adobe Acrobat (for PDF)
	Compile and link delivered source files	CASEVision (i.e., compilers, debuggers, profilers, etc. on SGI)
	Run test cases	Custom developed manager, GUI
	Examine test outputs, including metadata	Custom developed file comparison tools, IDL, EOSView
	Collect resource requirements statistics	CASEVision
	Update system databases	Custom developed scripts
	Write reports and maintain logs	WABI , MS Office
	Write additional ad hoc tools	SPARCworks (i.e., compilers, debuggers, profilers, etc. on SUN), CASEVision, code snippets

1 - MS Word will also be used to view WordPerfect and ASCII documents

This page intentionally left blank.

5. Developing Science Software

Many issues arise in the process of designing and developing SDPS/W. This section addresses a number of these which are broadly applicable. Early contact and consultation with the DAAC and ECS support personnel is necessary to resolve the specific issues related to the individual SDPS/W. Contact information is given in Appendix A.

5.1 Coding Standards

The coding standards for science software source code and scripts are described in *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines* (see Section 2). The purpose of the coding standards is to ensure that the science software will be portable between the development platform at the SCF and the target platform at the DAAC. Portability of the science software will also facilitate future hardware, compiler, system software or ECS software upgrades at the DAAC.

Another purpose of the coding standards, related to portability, is to ensure that SDP Toolkit functions are used for any activities that interface with the rest of the processing environment. This allows development of and upgrades to the processing environment to proceed independently of the science software development. Changes in the processing environment can be encapsulated in the Toolkit routines, rather than requiring changes to the science software.

One of the Integration and Test activities at the DAAC will be to run standards checkers which will check that the science software follows all required coding standards. Source code that fails to follow these standards will either have to get a waiver via the Earth Science Data and Information System (ESDIS) Project Science Software Manager, or will have to be modified to meet standards. Waiver requests that are accepted are temporary, with an expiration date (nominally, the next SDPS/W delivery). All code delivered to the DAAC after the waiver expires must adhere to standards.

5.2 Issues and Guidelines

The goal of portability is to minimize costs associated with rehosting the SDPS/W, or even just upgrading the version of the compiler being used. Over the operational lifetime of any SDPS/W, it is very likely that several upgrades to the DAAC hardware and language compilers will occur. Reasons for these changes can be: the discontinuation of vendor support, difficulty or expense of obtaining spare parts, poor reliability, or changing mission requirements of the DAAC.

5.2.1 Using Heritage Code

Many lines of executable code already exist and it is only reasonable to expect to reuse established, tested code where applicable. This reduces immediate development costs.

Much of this code, however, is not written to ANSI standards and may not compile using the compilers available at the DAAC. Even if the heritage code does compile at the DAAC, the use of non-ANSI standard code or structures can mean that the SDPS/W developer will need to make major code revisions to accommodate future changes in operating system, compilers, or hardware.

This could mean a large effort to (re)familiarize the SCF staff with the details of the design and source code if a significant period of time has elapsed since it was originally developed. An extensive effort to redesign, modify and retest the SDPS/W may also be involved.

It is the responsibility of the SDPS/W developer to make modifications to heritage code to bring it up to ESDIS standards (e.g. adhere to ANSI language standards, comply with the code header standard) before delivery to the DAAC. Third-party heritage code that is part of the SDPS/W may possibly be waived for part of the ESDIS standards (e.g. code header standard). Waiver requests should be submitted to an ESDIS Science Software Manager before detailed implementation of the SDPS/W begins. It is also the responsibility of the SDPS/W developer to include the mandatory SDP Toolkit function calls in the heritage code (see Section 5.3.1.1). From the perspective of long-term responsibility, it is advisable to expend effort during the time of the development of the SDPS/W to make heritage code more portable.

5.2.2 Using COTS

Use of COTS packages in the SDPS/W involves many of the same issues of long term support and compatibility with the PDPS as using heritage code. In addition, it is the SDPS/W developer's responsibility to obtain and provide the DAAC with the license to use the COTS package.

5.2.3 Prohibited Functions

The use of certain functions in SDPS/W is prohibited. This applies to SDPS/W written in FORTRAN 77, Fortran 90, C, and Ada as well as to the script languages Bourne Shell, C Shell, Korn Shell, Perl, and POSIX-compatible shell language. A complete list of all prohibited functions can be found in the *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996*.

5.2.4 Scripts

A PGE may consist of a script invoking binary executables. Perl, C Shell, Korn Shell, Bourne Shell, and POSIX Shell script languages are supported by ECS, and can be used to provide the logical framework around the binary executable(s) that produce the science data products. As much as possible, however, data processing functions should be coded using a compiled language, such as Fortran or C, since there are ANSI standards for these languages. Prohibited functions and utilities in scripts are documented in the ESDIS standards and guidelines document (*Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996).*

The actual PGE script, as initiated in the production environment, must not take arguments from the command line. Instead, script calls to command versions of some SDP Toolkit “Process Control” tools must be used to retrieve pertinent runtime information. (Note: These parameters are static rather than dynamic.) It is preferred, however, that the information be obtained from within the binary executables (using the program-callable version of the same tool) rather than passing this information through the shell interface. This allows for easier configuration of executables within the PGE script in the event that modifications are required at some point in the future.

When testing for the exit status of a binary executable within the PGE script, a zero (0) should be returned from the binary executable (using the programming language “exit” statement or “return”) to indicate success. Any valid return code may be used, however, provided that the SDPS/W developer is aware that some codes may already have a predefined meaning (see “errno. h” in the SDP Toolkit) .

Similarly, the PGE script must set the exit status employing the script language “exit“ command. A value of zero (0) must be used to indicate success. Non-zero values for indicating failure and possible response actions are documented within a white paper (Heroux, Dave, *Establishing Science Software Exit Conditions for the Production Environment,* White Paper).

Script languages provide a great flexibility to manipulate text, files and processes. A PGE script, for example, may use pipes to convey data to and from standard-in and standard-out for a sequence of binary executables. Looping and conditional commands are also allowable. However, the standard input and output must not be directed to or from the PGE script. That is, no redirection at PGE boundries.

5.2.5 DCE Restrictions

In a Distributed Computing Environment (DCE), processing for a single program can occur across multiple machines. While this technology may help to improve processing efficiency of some classes of problems and at the same time take advantage of underutilized processors, the constraints that DCE places on the ECS system architecture presently preclude the use of certain of its features.

Specifically, the use of remote procedure calls (RPCs) is not allowed. The creation of RPCs to perform some segment of processing for a science algorithm cannot presently be generalized into extended SDP Toolkit functionality. Since it is the Toolkit's charter to isolate the science software from the system architecture, the SDP Toolkit's inability to mask this feature prohibits its direct usage in the actual production software.

Also prohibited are remote file mounts. As described in the SDP Toolkit User’s Guide, all input or output files must be designated in a Process Control File and the Process control tools by the program or script to access the physical filenames and file attributes.

5.2.6 Log Files

There are 3 log files that can be generated using the SDP Toolkit routines: the Status log file, the User log file, and the Report log file. The SDP Toolkit writes to each log file in "append" mode. In the SCF environment, this means that the log files exist until deleted. This is done so that the log files can accept input from several executables within a single PGE. In the production environment, log files are still written in append mode, but are removed following every PGE run and placed in a unique Production History file on the Data Server. Therefore, a PGE's log files cannot be overwritten by other PGEs in the production environment.

More detailed information about the log files can be found in the *SDP Toolkit Primer for the ECS Project* (see Section 2). Every log file contains a banner which, among other things, identifies the Production Run ID, SDPS/W version ID, and start time.

Briefly, the Status log file is used to capture all pre-defined error and status information concerning a program. The SDP Toolkit errors are also written to the Status log. Each Status log entry consists of two lines, followed by a blank line. The first line gives the name of the function or subroutine that set the message, and mnemonic label of the message. The second line of a Status log entry consists of the text of the message. For developer messages, this is the message string obtained from the developer-provided status message text file.

The User log file is automatically updated every time the SDPS/W calls the appropriate SDP Toolkit functions. Thus, this file will contain only the developer-defined error and status messages that are specifically written to the User log file.

The Report log file is updated each time the SDPS/W makes a call to the appropriate SDP Toolkit function, which takes as input any string, and simply writes it to this file. The SDPS/W developer is in complete control of what gets written to the Report log file.

The developer has great flexibility in defining the information that is written in these log files. It is important that these error codes and status messages be clear and unambiguous. When confronted with a large number of log entries, the operator or maintenance programmer is presented with the problem of finding those messages that deal with a particular situation. The following are suggestions concerning information to include in status and error messages:

- If not automatically included, the Production Run ID, the SDPS/W version ID, and the name of the function or subroutine that trapped the error. The severity of the error is contained in the mnemonic label of the message (see the SMF tools description in the *Release B.0 SCF Toolkit Users Guide*.
- Date and time that the error was trapped.
- Names of input data file(s) (listed once at beginning of execution).
- Identifier for input record(s) being processed when the error was trapped, if appropriate.
- Identifier for processing group (e.g., scan cube, bin, etc.) and location within processing group, if appropriate.

5.2.7 Operational Considerations

The operational staff at the DAAC will interact with the SDPS/W over a period of many years. They will be the first to deal with errors and failures. Consequently, they need:

- Clear, concise, consistent and correct operational procedures and documentation.
- Understandable error codes and unambiguous corrective actions.
- SDPS/W design incorporating breakpointing or other strategies allowing for easy restart.
- Logical file layouts, with parameters unambiguously identified; units and scaling factors where appropriate.
- Friendly user interface.

These items are also important to minimizing the SDPS/W developers long-term maintenance costs.

5.3 Development Tools

5.3.1 The SDP Toolkit

The SDP Toolkit consists of a set of fully tested, fast, efficient and reliable C and Fortran language functions, customized for application to ECS. Formerly called the PGS Toolkit, it provides the interface between the SDPS/W and the ECS system, including PDPS, CSMS, and Information Management. It also promotes the ability to port the SDPS/W to different platforms at the DAAC, and can reduce duplication of coding effort among the disparate SDPS/W developers.

It is essential to understand the concepts that distinguish the SCF development environment from the PDPS production environment at the DAACs. While the science software and interface to the SDP Toolkit is preserved in both environments, there may be slightly different implementations and behavior in the toolkit functions and peripheral components (e.g., shell-level development and testing tools). The version of the SDP Toolkit that runs in the SCF environment is known as the SCF Toolkit; the version that runs in the DAAC environment is the DAAC Toolkit.

The Toolkit divides into two groups: mandatory tools, which the system requires in science software, and optional tools, whose primary intention is to save SCF development effort by reducing redundancy. Use of the mandatory tools will be verified manually during science software integration and test.

5.3.1.1 Mandatory Tools

Use of SDP toolkit routines is mandatory for file open/close, error and status messaging, process control and dynamic memory allocation. For details, see Section 6.2 of the *Release B.0 SCF Toolkit Users Guide.* These tools include:

- Generic Input/Output (**IO_Gen**) tools provide the means for non-HDF ancillary open and close support, temporary and intermediate duration files. The actual reads and writes are accomplished using standard C and FORTRAN commands.
- Error and Status Message (**SMF**, for Status Message Facility) tools provide general error reporting and status log messaging capabilities and interfacing to CSMS services. The Toolkit takes no action with regard to errors that are trapped; this is left to the science software.
- Process Control (**PC**) tools provide the primary interface to the PDPS. A major use of these tools is to access physical filenames and file attributes. In addition, they retrieve SDPS/W developer-defined parameters used in processing (e.g., selecting between variations of an algorithm that have been included in the source code).

Other mandatory toolkit routines include:

- HDF access tools read and write standard HDF and HDF-EOS product files.
- Level 0 access (**IO_L0**) tools access Level 0 data from EOS spacecraft.
- Metadata (**MET**) access tools allow science software to access, alter, write and append metadata.
- EOS spacecraft ephemeris and attitude access (**EPH**) tools read ephemeris and attitude data.
- Time and Date (**TD**) tools perform time and date conversions between selected time systems.

5.3.1.2 Optional Tools

The remaining tools are optional. They perform functions that are commonly needed in processing remotely sensed data. Individual science algorithms, however, may have unique requirements that will require customized routines to be developed to perform a given function. These tools include:

- Ancillary data Access (**AA**) functions access such data as NMC data.
- Celestial Body Position (**CBP**) tools locate the spacecraft, sun and other celestial bodies.
- Coordinate System Conversion (**CSC**) tools allow celestial reference frame coordinate conversions, and perform related tasks such as locating the sub-satellite point.
- Constant and Unit Conversion (**CUC**) tools allow access to physical constants and unit conversions.

- Digital Elevation Model (DEM) functions access DEM data.
- The IMSL package provides mathematical and statistical support.
- Graphics support in the production environment is provided using IDL.

There are also some test tools, which are for use during development at the SCF only. These include an ephemeris and attitude simulator and a Level 0 data packetizer.

For the most part the optional tools are independent of each other, though all depend on the lower level tools, including SMF (all tools), PC, IO_Gen, and TD.

5.3.1.3 Toolkit Languages

The Toolkit is written in the C language. A macro package provides bindings to the C code from FORTRAN 77 (a few exceptions are coded directly in FORTRAN 77). These bindings appear to have no effect on processing speed. Where possible, the same application program interface (API), i.e., calling sequence, has been used for both C and FORTRAN. Support of Fortran 90 requires no special bindings, since FORTRAN 77 is a subset of Fortran 90 (this has been confirmed by testing).

5.3.1.4 Toolkit Documentation

Documentation for the Toolkit can be found in the documents *Release B.0 SCF Toolkit Users Guide* (333-CD-004-001) and *SDP Toolkit Primer for the ECS Project* (No. 194-815-SI4-001), as well as on the ECS Project WWW SDP Toolkit Home Page (URL http://newsroom.hitc.com/sdptoolkit/toolkit.html).

5.3.2 Other ECS Tools Available to Developers

The DAAC will have a number of tools installed for the purpose of integrating and testing science software into the production environment, as outlined in Section 4.6.5. In general, these tools are provided only to the DAACs.

License agreements for some tools have been obtained, however, allowing the commercial tools to be used by a limited number of science software developers at their own facilities. Please contact the ESDIS Science Software Manager concerning the availability of tools and licenses. The developer, however, must assume the burden of license administration for any COTS tools that may be provided.

5.3.3 Obtaining and Using the Tools

Currently, the SDP Toolkit is available via ftp. Send electronic mail to ***pgstlkit@eos.hitc.com*** for further information. General information about the toolkit is provided in *SDP Toolkit Primer for the ECS Project* (see Section 2).

5.4 Constants and Coefficients

It is a common and recommended programming practice to define constants symbolically in a separate include file. These types of constants may include:

- Physical and geometric constants.
- Threshold values.
- Dimension(s) of arrays.
- Dimensions and offsets associated with input or output data.
- Loop limits.
- Error and other types of flag values.

Dimensions of arrays and loop limits are particularly well suited for include files when they are used by multiple software modules or are expected to be changed in the future.

Coefficients and threshold values which may change over time, however, should be read from a file at time of execution, however, rather than embedded into the source code. This approach simplifies the CM and I&T process when updates to these values are required. The source code does not need to be recompiled and the testing burden is reduced.

5.5 Metadata

Metadata is descriptive information about individual input and output product files (i.e., granules) and about a “collection” of files that make up the data set. This includes information such as:

- Parameters contained in the output product.
- Spatial and temporal coverage.
- Identifier(s) of the data inputs.
- Quality assessment of the output product.
- Version of the science software used to generate the granule.

The metadata is essential for an EOSDIS science data user to locate and acquire products based on a selection of time, space and other attributes. Some metadata attributes are common to all products at a given level (i.e., L0-L4), and are referred to as the “Core” metadata. Metadata that uniquely applies to a product are referred to as “product-specific” metadata. These are implemented in a manner which allows complex yet efficient search operations to be performed. More detailed information regarding the data model can be found in the Technical Paper *Preliminary User View of Release A Data* (420-TP-005-002). Detailed information on metadata population can be found in the documents *An ECS Data Provider’s Guide to Metadata* (163-WP-001-001) and *The Population Process for ECS Metadata in Release A* (420-TP-014-001).

5.5.1 Earth Science Data Types

The ECS Science Data Model, which is described in Release B *Science Data Processing Segment (SDPS) Database Design and Database Schema Specifications for the ECS Project* (311-CD-008-001), contains a description of all of the metadata attributes which are necessary to identify, interpret, and perform services on granules and collections. Valid values and ranges are specified for Core attributes when appropriate. Product-Specific valid values and ranges are specified by the data providers when they define the attributes. This composite of database information is known as the Data Dictionary. The set of attributes describing a collection and the services to be performed define an Earth Science Data Type (ESDT) for the collection.

To establish a collection of data granules, the metadata attributes relevant to the collection and granules are organized into an ESDT descriptor file. Services available for the data are also specified in the descriptor file. Thus, the ESDT descriptor file can be viewed as a composite of several parts, with an underlying, implicit Data Dictionary containing the description of the attributes and valid values and ranges:

- Collection level metadata attributes
- Granule level metadata attributes
- Valid values and ranges
- Services available for the data

The code to implement the services exists separately in the Dynamic Link Library (DLL).

In general, the various parts of the ESDT descriptor will be built using different methods and associated tools. Therefore, deliveries to the DAACs may occur via different processes and at different times. Once all the parts of a new ESDT descriptor are delivered, the SSI&T staff will concatenate the files and possibly add or modify sections. The descriptor files must be in Object Description Language (ODL) format for insertion onto the Science Data Server. The ODL files are ASCII files and may be edited.

Metadata contained in the ESDT descriptor file will be validated by Data Server at three different points. First, when the descriptor ODL file is parsed, basic checking of attribute names and membership in proper metadata groups will be performed. Secondly, prior to insertion into the database tables, core attributes will be validated against validation rules or “valids.” Valids can be one of three types: boolean expression, range (numerical value between specified bounds), and match from an enumerated list. Valids checking will be applied to product-specific attributes if the validation rules have been supplied in the ESDT descriptor file. The final checking of metadata is done upon insertion of the data into the database tables. Here, relational constraints will be checked such as links between attributes.

5.5.2 Collection Level Metadata

MetaDataWorks is a metadata entry tool that extracts the metadata attributes and valid values from the online data dictionary. This tool has been developed by ECS to support the generation of ESDT descriptor files. It uses a WWW browser interface to collect metadata information in HTML forms. Using these Web-based forms, users may set defaults for collection attributes which are readily incorporated into subsequent ESDTs. Information entered for each ESDT can be saved, viewed, and later edited. MetaDataWorks also enables rapid entry of the attributes to be used at the granule level. Its output is the ODL file comprising the ESDT descriptor, as well as the Metadata Configuration File (MCF) required by the SCF Toolkit for processing.

For Pre-Release B Testbeds, the ECS SSI&T team will used an early version of this tool to support the generation of ESDTs for the Instrument Teams. In later releases of ECS, there will be a formalized set of procedures for developing ESDTs.

5.5.3 Granule Level Metadata

The SDP Toolkit metadata tools are to be used by each PGE to generate the granule level metadata, and are described in Appendix J of the *Release B.0 SCF Toolkit Users Guide*. The allocation of optional parameters between Core metadata and product specific metadata needs to be established through dialog between the science software developer, ESDIS and ECS project personnel well in advance of SSI&T at the DAAC.

The collection level Core and product-specific metadata will be populated before the time of SSI&T. The granule level Core and product-specific metadata are populated during product generation or may be specified as fixed values in the Metadata Configuration File (MCF) .

Appendix J of the *Release B.0 SCF Toolkit Users Guide* delineates the mandatory and optional granule level metadata, how these metadata are provided to the product header, how the metadata tools assist this process, what the PGE has to do, and the normal outcome of running the metadata tools. In brief, the SDPS/W developer provides ECS (later this will be done by the DAAC) with product specifications. ECS (at a later time, the DAAC will do this) will then build the ESDT Descriptor and submit it to the DSS. Using this information, the DSS generates the ESDT and MCF. The ECS (the DAAC will do this later) then provides the ESDT and MCF to the SDPS/W developer who provides feedback/suggestions. If changes are needed, ECS (at a later time, the DAAC will do this) incorporates these into another iteration of ESDT and MCF generation. These are, again, provided to the SDPS/W developer who will use the MCF for development testing at the SCF since the SDP Toolkit metadata routines require the MCF. Other values, obtained from information supplied in the Process Control Files and entered into the PDPS database, are automatically extracted by the metadata tools during the specific PGE execution within the PDPS. Finally, the granule Core and product-specific metadata is written to the HDF file containing the product output as a product header.

The association of a granule with a particular collection is through the ShortName and VersionID metadata attributes. These attributes, therefore, occur in both the collection level metadata and in the granule level metadata for granules belonging to that collection. The granule level

ShortName and VersionID values as set in the MCF *must* therefore match the corresponding collection level ShortName and VersionID values as set in the ESDT descriptor file.

Early generation of MCFs and ESDT descriptors for the collection level metadata at the ECS or DAAC before SSI&T begins would allow the entire ESDT descriptor to be constructed and tested on the SDSRV in time for the PGE delivery. The editing and concatenating of the descriptor files for new ESDTs may require a considerable level of effort by the SSI&T staff at the ECS or DAAC.

5.5.4 Services

Services refer to actions that can be done on the granules of a particular collection by the ECS. The Insert and Acquire services are the most basic. Insert allows granules to be inserted to the Data Server and to be associated with a particular collection. Granules are said to be *added* to a collection on insert. Acquire allows granules to be retrieved from the Data Server. More advanced ECS services include subsetting and subsampling. These services allow granules to be subsetted or subsampled as they are retrieved from the Data Sever.

Services to be performed on the data collection have to be designated and inserted into the ESDT descriptor files before the descriptor is loaded onto the SDSRV. The ODL descriptor files may have to be edited at the DAAC. The ODL syntax for services will be available at SSI&T.

ECS provides a core set of services that can be configured for each ESDT. However, some ESDTs may need specialized services that require the ECS development team to add additonal DLL code. For further information on specialized service needs, see *Section A.2 Other Support* in this document. A point of contact is listed under the heading *Other SDPS/W Development and SSI&T Questions*.

Higher-level services such as subsetting (which allows extraction of a portion of the data), and subsampling (which allows coarse reduction of the resolution of the data) are available for specific ESDTs. These services are limited initially to HDF-EOS format files. Also, multi-dimensional files are not supported by default. The default services support selection by latitude, longitude, altitude, time and by parameter. Data structures supported are: swath, grid and point. For details on guidelines for organizing swath, grid and point data to a common standard in order to achieve optimum performance in the service, the reader is referred to: *Writing HDF-EOS Swath Products for Optimum Subsetting Services*, 170-TP-009-001; *Writing HDF-EOS Grid Products for Optimum Subsetting Services*, 170-TP-007-001; and *to Writing HDF-EOS Point Products for Optimum Subsetting Services*, 170-TP-008-001.

5.6 Recommended Directory Structures for Development and Delivery

The manner in which the directories and sub directories are organized can have an effect on the process of transferring the SDPS/W to the DAAC and on the SSI&T process. Various approaches are equally valid, however. The specific structure will probably be a part of the agreements reached between the SDPS/W developer and the DAAC. The following is one recommended directory structure. Directory names are shown in **bold** font. Subdirectories are shown indented from their parent directories.

- **doc/** This directory contains all system level documents.
- **run/** A directory containing (within subdirectories) all executables, test data and scripts needed for running the supplied test cases.
    - **exec/** This subdirectory will contain the scripts and binary executables, including those used in install procedures, and other miscellaneous executables.
    - **anc/** This subdirectory will contain the supplied ancillary data files required for testing
    - **log/** This directory contains subdirectories for each PGE. Log files generated during test case execution are placed in the appropriate subdirectory.
    - **output/** This directory contains subdirectories for each PGE. The output data products are placed in the appropriate subdirectory.
    - **pgecfg/** The default directory for Process Control Files and Metadata Control files.
    - **test/** This directory has two subdirectories, one for test inputs and one for expected test outputs.
- **store/** This directory contains source and compressed data files within subdirectories for each PGE plus subdirectories for libraries and utilities common to more than one PGE.
    - **cg/** Common and Global functions.
    - **cfg/** Configuration management tools and functions.
    - **dif/** Data ingest and formatting functions.
    - **utl/** Utility programs - required for testing and installation.
    - **lib/** Contains sharable object libraries (if applicable).
    - **pge*X*/** Separate subdirectories for each PGE (*X* referring to an identifier for each).
        - **doc/** Contains a README file describing PGE*X*.
        - **include/** Contains the include files which are part of PGE*X*.
        - **source/** Contains the source code and make files for PGE*X*.
        - **test/** Contains a README file describing the test cases, test procedures, and how to evaluate the test outputs.
        - **coeff/** Contains coefficient files tables.

5.7 Test Data

The need for test data derives from the goals of SSI&T of testing the portability and production readiness of the SDPS/W. Additional details are provided in the White Paper *Science Software I&T Requirements for Test Data,* 162-WP-002-001.

5.7.1 Portability

The SDPS/W developer must provide test data for standalone, portability testing of each PGE, including:

- Input data
- Granule level metadata values so that the input test data may be Inserted to Data Server for standalone testing. These are the values that a PGE producing the data would have set. The metadata attributes should already be known from the ESDT Descriptor files associated with the PGE that will be used to produce the data in the production system.
- Ancillary Data, if applicable
- Coefficient files, if applicable
- Expected Test Results

If the SSI&T file comparison tools available at the DAAC are to be used, then additional information must be provided:

- For HDF files, an ASCII file containing the allowed tolerances for each output parameter.
- For intermediate files in binary format, the allowed tolerance for each parameter, and either
  - The file and record layout, and/or
  - A routine to read records from the binary file.
- Metadata must be supplied in HDF files.
- A type reference file generated by executing an ECS-supplied program.

5.7.2 Resource Profiles

Information needed for the Planning and Processing Subsystems is entered into the Planning Subsystem Data Base during the science software I&T. Some of the items are supplied by the instrument team (e.g., data dependencies, production rules, product file sizes), but the nominal and maximum resource profiles for each PGE must be determined first during SSI&T.

The appropriate elements relating to the resource profile that needs to be established for each PGE are:

- PGE CPU time
- Elapsed time

- Average amount of shared memory used
- Maximum memory used

These values are used to establish nominal and extreme values which will be monitored during actual product generation. They will also be continually updated during operational processing.

These elements can be measured during the standalone, portability testing described in Section 5.7.1. After the PGE is linked with the SCF Version of the SDP Toolkit, it can be run from the command line in a script which uses the DpPrRusage utility to do the profiling. DpPrRusage will be available at the DAACs. To be representative of normal resource utilization, however, at least some of the input and ancillary test data provided by the SCF must:

- Represent the normal range of input file lengths
- Contain, for each parameter, values which span the appropriate, reasonable range.

5.7.3 Production Readiness

The principal goal of the SSI&T is to test the production readiness of the science software. This includes verification of the entries for each PGE in the Planning and Data Processing database concerning the PGE profile and the information needed to generate a Process Control File.

Verifying that the Planning and Processing database entries are correct requires sufficient test data for activation of each PGE, e.g.,

- A PGE processing a single granule must have a minimum of one granule and appropriate ancillary data (if any) provided by the SCF.
- A PGE generating a weekly composite must have a minimum of one week of input data and appropriate ancillary data (if any) provided by the SCF.
- A PGE generating a monthly product must have a minimum of one month of input data and appropriate ancillary data (if any) provided by the SCF.

Other tests of production readiness need to be performed to ensure that science software runs to normal completion repeatedly over the normal range of data inputs and run-time conditions (including nominal excursions) and executes without interfering with other S/W or DAAC operations. Therefore, some additional test data which includes excursions or data corruptions need to be supplied by the SCF.

6. Delivery of Science Data Production Software

The material in this section is presented as a template of basic intent, and not as rigid rules. The procedures and mechanisms need to be defined for the specific SDPS/W being developed and the DAAC environment within which the product generation will be done.

6.1 Prior to Delivery

Early and frequent communication between the SDPS/W developer and the ESDIS Project and DAAC personnel is a key factor in the successful integration of the SDPS/W with ECS. This section describes some of the activities that lead up to the delivery of the SDPS/W to the DAAC.

6.1.1 SDPS/W Resource Specification

For new SDPS/W and for extensive modification or enhancement of existing production SDPS/W, an assessment of the impact on EOSDIS operational, computational, storage and communications resources will be necessary. The developer will be asked by the DAAC to provide information about the SDPS/W in order to permit impact assessments to be performed.

The requested information about the SDPS/W will likely include:

- PGE Information
  - PGE ID
  - Baseline Estimate of Wall Clock Time for PGE Execution
  - Number of Floating Point Operations (FLOPs) for PGE Execution
  - List of Input Products and their ESDT ShortNames
  - List of Output Products and their ESDT ShortNames
  - List of Intermediate files and their ESDT ShortNames

    List of temporary, intermediate or product files, each with file name, volume (MB) produced by a single PGE execution, spatial and temporal coverage
  - Production rules for the PGE that describe the rules in PDPS for activating the PGE. This should be a text description of the conditions under which the PGE is activated. The information required depends upon the complexity of the production rule and upon the capabilities of the ECS release to which the science software will be delivered.

- File Transfer Templates
  - Data Transfer ID
  - Product ID to be Transferred
  - Source Location
  - Destination Location
- QA Activity Templates (at the same time resolution as PGE)
  - List of Persons and Locations Involved
  - List of Products Required and their ESDT ShortNames
  - List of QA Products Generated and their ESDT ShortNames
  - Estimated time from start to finish
- Process Activations
  - List of Processes (PGEs activated, Files transferred, QA activities done)
  - Frequency each process activated
- How Do Descriptions Change With Time?

6.1.2 Operations Readiness Plan

Before a delivery of new SDPS/W is planned, the developer needs to contact the DAAC and ECS support staff. A plan must be started by the developer and the DAAC as soon as feasible, and updated as the scheduled date of the delivery approaches, and as a result of the acceptance testing at the DAAC. The Operations Agreement (OA) serves to document the understandings of the involved parties concerning the sequence of steps to be performed leading to the SDPS/W being put into production at the DAAC, and the responsibilities of each involved party.

6.1.3 Other Pre-Delivery Activities

The determination of the acceptance criteria for a delivery of new SDPS/W or enhancements to existing production SDPS/W and agreement regarding the test plan needs to be accomplished prior to the delivery. The ESDT descriptor files, including the collection level attributes, granule level attributes, and services, must be edited and concatenated into a complete ESDT descriptor file. The descriptor file is then loaded onto the SDSRV before SSI&T. Other items such as standards checking may be accomplished prior to delivery.

In addition, developers at SCFs may use remote access interfaces at DAAC discretion to carry out remote integration and test of their SDPS/W in the DAAC test environments. If the DAAC authorizes use of these interfaces, an SCF member can remain physically located at the SCF while using an SCF machine for interactive access to ECS software. Refer to *ICD Between ECS and Science Computing Facilities (SCF)* for further details on this interface.

6.2 SDPS/W Deliveries

The items comprising the entire SDPS/W system are indicated in Table 6-1. A delivery will typically contain more than just source code. All items require some degree of review or test before they can be accepted at the DAAC. These items are generally packaged into a tar file for delivery to the DAAC. The package is called the Delivered Algorithm Package (DAP).

All deliveries of SDPS/W are made to the DAAC with appropriate discipline expertise (indicated in Figure 1-1), or alternately delivered to the DAAC identified in a Working Agreement with NASA (or similar memorandum). Following delivery of SDPS/W source code to the DAAC, it will be integrated with ECS (see Section 7), and acceptance testing will be performed prior to the SDPS/W being placed into production. Updates to coefficient or control files require a test to ensure that they have been correctly installed.

A delivery package may merely consist of a single updated module, a control or coefficient file or a documentation update page. At the other extreme, a delivery package may consist of the entire SDPS/W system and supporting documentation.

Although the Algorithm Theoretical Basis Documents (ATBDs) are listed as optional in Table 6-1 with regard to elements that may be delivered to the DAAC, the ATBDs are a requirement of the EOS Project Science Office. The ATBDs are currently available from URL: http://spso2.gsfc.nasa.gov/spso_homepage.html; reference to where the ATBDs may be obtained must be included in the provided documentation if the ATBDs are not delivered to the DAAC.

Table 6-1. Elements of Science Data Production "Software"

Data	Documentation
Coefficient Files	Algorithm Theoretical Basis Documents (optional)
Control Files (PCF and MCF)	System Description Document
Test Data	Operations Manual
Expected Test Results	File Description Document
PGE Production Rules	Test Plans, Test Procedures, and Test Results
	Data Product Catalog
Source Code	
Module - e.g., subroutine, function, main, includes, commons, block data, make files, shell scripts	
Subsystem - one or more main programs	
PGE - a single binary executable or a script linking several binary executables	
Science Data Processing Software (SDPS/W) - one or more PGEs	

6.3 Delivery Contents

A delivery of SDPS/W may include all of the items listed in Table 6-1, or may only involve some of the elements listed there. The elements that may be required in a delivery depend on the purpose of the delivery and the phase of development of the SDPS/W. As shown in Table 6-2, a delivery in the early stages of development, intended to test portability and SDP Toolkit usage, has fewer required elements than the Launch-ready version.

With each delivery, however, a delivery memo is required containing the applicable items described in Section 6.3.1 below. It is recommended that the actual delivered elements be grouped by PGE, containing the applicable items listed in Sections 6.3.2 and 6.3.3. Elements that are common to several PGEs may be grouped together, and need not be repeated with each PGE.

Table 6-2. Recommended Elements by Delivery Type (1 of 2)

Type of Delivery	Purpose of Delivery	Source Code, etc.	Data	Documentation
Early Testing e.g. a β-version)	To demonstrate portability and exercise the SDP Toolkit interfaces at the DAAC.	Source code (may contain stubs) for one or more PGEs, make files, scripts	Test Data, Control File(s), Coeff. File(s), Expected Test Results	Test Plan(s), Compilation Listings, Data Flow Diagram(s), First Drafts File Descr. Doc. Data Product Catalog
Engineering (e.g., Ver. 1)	To test all major functional capabilities, interfaces, and standard error and message services. The performance and resource utilization of this version will be measured to obtain an estimate of the final operational version.	Complete (but not final) source code for *all* PGEs, make files, scripts.	Test Data, Control File(s), Coeff. File(s), Expected Test Results, PGE Production Rules	Test Plan(s), Compilation Listings, First Drafts of System Descr. Doc. Operations Manual, Second Drafts File Descr. Doc. Data Product Catalog

Table 6-2. Recommended Elements by Delivery Type (2 of 2)

Type of Delivery	Purpose of Delivery	Source Code, etc.	Data	Documentation
Operational, Launch-Ready (e.g., Ver. 2)	To install the complete, verified operational SDPS/W that will support the post-launch data processing.	Complete, verified source code for *all* PGEs, make files, scripts.	Test Data, Control File(s), Coeff. Files, Expected Test Results, PGE Production Rules	Test Plan(s), Compilation Listings, Second Drafts of System Descr. Doc. Operations Manual, Final File Descr. Doc. Data Product Catalog
Operational Documentation	To provide the DAAC with the Final versions of the Operational Documentation.	Not Applicable	Not Applicable	Final System Descr. Doc. Operations Manual.
Operational Enhancement	To make substantial improvements to the SDPS/W reflecting experience gained during the months following operational implementation.	One or more PGEs, make files, scripts.	Control File(s), Coeff. Files, Expected Test Results †, PGE Production Rules	Test Plan(s), Update Pages to documentation
S/W Patch	To correct a minor error in the source code, not affecting performance or resource utilization.	Corrected source code.	Expected Test Results†	Test Plan(s), Update Pages to documentation, if needed
Coefficient File	To replace the existing production coeff. file(s) to obtain improved product accuracy.	Not Applicable	Coeff. File(s), Expected Test Results. †	Test Plan(s)

† Static Test Data assumed to already reside at the DAAC under CM

6.3.1 The Delivery Memo

A memo describing the purpose and contents must accompany each delivery. It is recommended that the delivery memo (one memo per delivery) include the following information:

1. Investigator or Team
2. Point(s) of contact

   For questions regarding only the delivery (include name, telephone, email)
3. Purpose of the delivery

   e.g., an initial release, modification.
4. Delivery context diagram(s), if delivery is an update

   e.g., block diagram(s) of the SDPS/W, with the subsystems that are being changed clearly identified.
5. Descriptions of problem(s) resolved, if any

   List problem reports resolved with this delivery and method of resolution
6. List of delivery contents (see Section 6.3.2)
7. A summary of impacts or changes to the operations, e.g., resource requirements, procedures, interfaces.
8. Other pertinent information, as judged by the SDPS/W developer or the DAAC.

6.3.2 Summary Information

It is recommended that the following information be provided for each of the delivered items as applicable, organized by PGE. Elements that are common to several PGEs may be grouped together, and not repeated with each PGE.

1. Name and function performed
2. PGE Identifier

   Each PGE is to have a unique identifier assigned by the SDPS/W developer. This unique identifier may be one component of a longer name that includes instrument acronym, PGE version number, and release date.
3. PGE Components and Developer-assigned Version Numbers

   At the DAAC, each PGE will receive a new version number when any component is changed, e.g., scripts, source code, coefficient files, or control files. The SCF version number of each component will be referenced. When SSI&T has successfully been completed for a PGE, the DAAC assigns a version identification. This version is entered as a ClearCase label (not to be confused with ClearCase's standard versioning) for the PGE source and related files in ClearCase. It also is entered as a label for the PGE executable and SSAP which are then inserted on the SDSRV. The DAAC and Instrument Team should come to an agreement on the PGE version identification.

Consequently, there is no need for the SDPS/W developer to adopt or accommodate the version numbering scheme employed at the DAAC. It is recommended, however, that a new PGE release be given a new version level (e.g., V2.0 to V3.0), and that updates to individual components be given new sublevels (e.g., V2.0 to V2.1)

4. Point of contact

    For technical questions specific to this PGE

5. SCF test site configuration (need only be provided once if applies to all delivered source code)

    Hardware

    Operating system and version number

    Memory

    Software libraries, tools and version numbers (e.g., IMSL)

6. Compilation Information

    Source code compiled listing

        To assist in troubleshooting if problems arise when trying to compile, and to show which compiler options were used

    Information from the relevant manual pages describing the compiler options used

        Explains options used to assist in porting to other compilers

7. Listings of the first few records of binary coefficient files

8. List of references to associated documentation.

    Each document referenced must either be included with the delivery, or information needs to be provided detailing how to obtain the document.

9. Summary of files associated with this PGE

    For each file utilized by this PGE, enter

        Name

        Format (e.g., ASCII, postscript, etc.)

        Type (e.g., C source code, makefile, test data, defect list, etc.)

        Version Number

File size in bytes

Short text description

10. Estimate of resources required for execution
11. List of Processing Dependencies

    Ancillary Data

    Pre-processing

12. List of known defects
13. List of exit codes and their associated messages

In addition, either new documents or update pages are required for deliverable documentation, as delineated in working agreements with the Flight Project or the DAAC. Annotated outlines for recommended documents for operational use at the DAAC are provided in appendix C, and attempt to organize the SDPS/W documentation that is identified in many working agreements into documents that are more useful to operational staff. The recommended operational documents are:

- System Description Document (SDD)

  The SDD is intended to explain the overall workings of the software system. The SDD may also serve as a reference as to the more detailed information provided in the other documents. This document can be considered as Part 1 of a User's Guide.

- Operations Manual

  Operations procedures are instructions to the DAAC staff on the manual activities that need to be either routinely performed, or performed in reaction to specific circumstances. This document can be considered as Part 2 of a User's Guide.

- Processing Files Description Document

  This document details the file and record layouts for each PGE of the SDPS/W.

- Test Plans

  Description of the testing performed at the DAAC to ensure safe and reliable production (e.g., stress testing, a commissioning period, a parallel test period).

The following documentation may or may not be required by the individual DAAC, but is strongly encouraged:

- Scientific Documents (ATBDs, etc.)
- Interface Definition Document
- Detailed Performance Testing Results

- Detailed Design and/or Implementation Documents
- COTS User and/or Programmer Guides

6.3.3 PGE Software, Control and Data Files

The following deliverable items need to be provided for each PGE, as applicable:

- Science software source code, including make files and scripts (required)
- Testing software source code, including make files and script files (optional)

  (e.g., programs to display the contents of binary intermediate files)
- Test input data (required) and expected output (required)
- Coefficient Files (required)
- Process Control File (required)
- Metadata Configuration File (optional, but highly recommended)

6.4 Configuration Management

The DAAC will only baseline the launch-ready version of the SDPS/W for long-term CM. Earlier deliveries will be placed under CM during the integration and testing effort, and as-needed per DAAC policy.

SSI&T will be independently controlled at each DAAC respecting ECS CM procedures, nonconformance reporting, and tracking of SPS.W configurations. Small changes to PGEs will be readily accommodated, though still controlled. However, changes to a PGE that may require reprocessing could have broader impacts that require careful assessment and approval.

6.5 Delivery Procedures

As noted in Section 6.1, it is recommended that the activities leading up to the delivery of one or more PGEs to the DAAC begin well in advance of the actual delivery. The effort and resources required for integration and test of the SDPS/W (i.e., SSI&T) depend upon the number of PGEs involved in the delivery, and the stage of the SDPS/W development when the SSI&T is being performed. The actual delivery procedures will need to be agreed to by the SDPS/W developer and the DAAC, but will typically include:

1. At some point before the intended start of SSI&T (suggested minimum: one month), it is recommended that the SDPS/W Manager send a request to the DAAC to indicate the need for user-accounts, investigator directories, and resource access permissions. Whenever possible, it is desirable for the SDPS/W developers to perform some testing of their software at the DAAC prior to the DAAC's SSI&T campaign, either in person or remotely.

2. The DAAC will typically confirm the request, and provide (if requested) user-account information as well as information concerning the Investigator Directory created to support this developer activity. The planned System Testing and Preventative Maintenance schedules affecting the allocated hardware may also be included in the reply.

3. The SDPS/W team prepares all the files needed for transfer. All source code files, make files, test data, test plans, expected test results, and documentation associated with a PGE may be combined in a single compressed, tar file (see Section 5.5 for recommended directory structure). This file might be labeled:

   *name*_*pge*_V*number*_*date*.tar.Z

   where *name* indicates the instrument (or interdisciplinary investigation), *pge* is a unique identifier for the PGE, *number* is the PGE release version number, and *date* is in the month-day-year numerical format (i.e. mmddyyyy) indicating when the tar file was created. For example,

   CERES_PGE3_V2.0_12151998.tar.Z

   for a CERES tar file created on December 15, 1998. If inclusion of the test data in the tar file results in a file that is too large for effective network transfer (e.g., more than 50-60 megabytes), the PGE test data can be placed in tar file(s) separate from the other PGE items. If separate compressed, tar files are used to transfer the test data, this fact needs to be clearly noted in the delivery memo, and the file names must include the PGE identifier.

   Elements that are common to several PGEs may be grouped together, and not repeated with each PGE. A *pge* of COMMON could be used for these items.

   It is recommended that all of the delivered SDPS/W documents be placed in a separate compressed tar file, labeled

   *name*_DOCS_V*number*_*date*.tar.Z.

   For example,

   MODIS_DOCS_beta_01101996.tar.Z

   for a MODIS tar file created on January 1, 1996.

4. The SDPS/W and test data are transferred via ftp or media to the DAAC.

7. Overview of the Integration with ECS and Acceptance Tests

The principal goals of the I&T of SDPS/W are to (1) test the ability of the SDPS/W to run to normal completion repeatedly over the normal range of data inputs and run-time conditions, and (2) ensure that the SDPS/W executes without interfering with other DAAC operations. This section outlines the general process employed at the DAAC during the SSI&T of delivered SDPS/W. The specific procedures are delineated in agreements established between the SDPS/W developer and the DAAC.

SSI&T is a very instrument specific process, and a single operations concept cannot cover all permutations. There are several approaches or models to performing I&T of SDPS/W, three of which are described in Table 7-1. These three models differ in whether some or all of the subsystems of a SDPS/W are delivered at one time and in the responsibilities of the participants in the SSI&T process.

Table 7-1. Three† Models for SSI&T

I&T Model	Description	I&T with ECS
The Big Bang Delivery	All subsystems of the SDPS/W are developed at the SCF and then transferred to the DAAC in a single delivery.	Done by DAAC/ECS staff. Supported by SDPS/W developer. Acceptance by DAAC Manager
The Multi-stage Delivery	Individual subsystems of the SDPS/W are transferred to the DAAC at scheduled times.	Done by DAAC/ECS staff. Supported by SDPS/W developer. Acceptance by DAAC Manager
Infiltration or Infusion	The SDPS/W elements are entirely or gradually transferred to an investigator work area at the DAAC.	Stand-alone testing led by SDPS/W developer, assisted by DAAC/ECS staff. Operational Testing led by DAAC/ECS staff, assisted by SDPS/W developer. Acceptance by DAAC Manager

† Other SSI&T models and variations of these three are also possible.

In the Big-Bang Delivery model, all subsystems developed at the SCF are transferred to the DAAC in a single delivery. Alternately, individual subsystems are transferred separately at scheduled times (i.e., the multi-stage delivery). In both of these approaches, the SSI&T is performed at the DAAC by the SSI&T staff with the support of the SDPS/W developer.

If the DAAC and the SDPS/W developer agree, however, the SDPS/W developer may opt to perform some part of their final development, the integration and/or the testing of their software at the DAAC (i.e., the infiltration or infusion model) with the support of the DAAC staff. Remote testing by the SDPS/W developer may be possible, or this testing may be done on-site. Final operational testing will be performed by the DAAC SSI&T staff, with the support of the SDPS/W developer.

Regardless of SSI&T model followed, the SSI&T process includes three broad phases which are applicable to all versions of new SDPS/W systems, revisions to previously accepted production software, and changes to coefficient or control files:

- Inspection - the delivery package is received, examined, and reviewed for testing.
- Integration - source code is compiled and linked in the operational PDPS environment.
- Acceptance Testing - the software is tested in the bounds of the production environment, leading to production status.

7.1 SSI&T Roles and Responsibilities

The developer is at all times responsible for the validity of the products produced by the SDPS/W and the science algorithms employed. As such, the SDPS/W developer is responsible for any changes made to the SDPS/W at the DAAC, even if the developer authorizes DAAC or other personnel to make specific changes to the software, coefficient, or control files.

During SSI&T, corrections to the software may be identified by developer or DAAC personnel that are needed to complete the SSI&T process. It is important, therefore, that the SDPS/W development team be represented at the DAAC during the SSI&T of delivered SDPS/W.

The developer is also responsible for correcting all deficiencies in the SDPS/W documentation that may be identified during the SSI&T process. This may be accomplished by submitting update pages to the affected documents.

7.2 Integration of Software with ECS

This phase of the SSI&T process will often be led by the SDPS/W developer. The SDPS/W source code transferred to the DAAC is first compiled and object code is linked with appropriate libraries. A temporary CM storage pool may be established for this phase, under the control of the SDPS/W developer or SSI&T personnel, in order to (1) allow code fixes to be readily accomplished without encumbering the DAAC CM policies and procedures, and (2) record the change history of the SDPS/W during this phase of testing.

Initially, the software may be linked with the SCF version of the SDP Toolkit appropriate to the DAAC hardware. The output files from the test runs are compared to the expected output files. Performance statistics are monitored and recorded during execution of the software, and compared with those obtained at the SCF. (The motivation for this step is based on the fact that software performance can be unexpectedly altered when executed on different hardware, or even the same hardware but employing different versions of compilers or operating systems. The information obtained by this step is valuable to understanding the effects of differences between the SCF and DAAC environments on the SDPS/W. This step will likely be performed only during SSI&T of the first delivery or two of the SDPS/W.)

The software object modules may then be linked with the Toolkit version resident at the DAAC for further stand-alone testing, i.e., testing which does not involve the planning database or the ECS PDPS. When the standalone testing is completed, a DAAC-led testing phase is performed. At the point of this transition, the SDPS/W elements which are to be used are marked as "delivered," forming the Delivery Package.

7.3 Inspection of the Delivery

The delivered SDPS/W is checked against the Delivery Memo and the list of required elements as previously agreed to by the SDPS/W developer and the DAAC. Supplied documentation is checked for completeness, comprehensibility, consistency and correctness to the extent possible. The DAAC will probably also run the source code and scripts through the standards checkers to check compliance to coding standards. The SDPS/W will likely be placed under DAAC configuration control at some point after the stand-alone testing described in Section 7.2 and the inspection by the DAAC personnel.

7.4 Production Testing Readiness Review

After the inspection of the delivered elements and initial stand-alone testing are done, a review may be held involving the SDPS/W developer (perhaps remotely), DAAC management, SSI&T personnel, ESDIS Science Software Manager (perhaps remotely) and possibly other ESDIS and EOS Project representatives (perhaps remotely). The aim of a review would be to provide an assessment of the delivered software's readiness for testing in the production environment. Success criteria for this review may include the delivery of all required files, resolution of any deviations from the established standards, and the viability of the software for further operational integration and testing. The decision of the DAAC management may be to continue with SSI&T, or to stop until the identified deficiencies have been eliminated.

7.5 Testing

7.5.1 Executing the Test Plan(s)

Following successful stand-alone testing, the testing of the SDPS/W progresses to a phase where the PGEs are executed in a manner approaching how they would be run in operation. This involves entering gathered information about each PGE into the planning database and using the PDPS to perform further testing. This phase will normally be led by the DAAC.

7.5.2 Relevant Success Criteria

SDPS/W deliveries to a DAAC will occur at several different stages of development and production. Typically, pre-launch SDPS/W deliveries may include a β-version, an Engineering version (e.g., Ver. 1), and a Production-Ready version (e.g., Ver. 2). Other deliveries involve software, coefficient file, or documentation updates.

The criteria for "acceptance" depends upon the nature and scope of the delivered items, as well as the purpose of the delivery. Discussions between the DAAC and the SDPS/W developer concerning the applicable criteria relating to a specific delivery need to be completed before the actual delivery is made.

7.5.2.1 β-version Delivery

Delivery of a β-version SDPS/W to the DAAC will nominally occur approximately three years before the launch of the instrument relevant to the software. The SDPS/W is in the early stages of development, and may be based on heritage or prototype code which may not be fully compatible with the SDPS Toolkit routines. All files required for execution must accompany the software. (See Section 6.)

The purpose of this delivery is to demonstrate portability and exercise the SDP Toolkit interfaces at the DAAC, and the β-version SDPS/W will not be placed into production. Tests of the production capability of this version of the SDPS/W are not appropriate.

7.5.2.2 Engineering Version (e.g., Ver. 1) Delivery

The purpose of this SDPS/W delivery is to test all major functional capabilities, interfaces, and standard error and message services. The delivery of an Engineering Version will nominally occur 18 - 24 months before launch, and be substantially complete. (For some higher level standard products, this delivery may occur after launch.) The final science algorithms may still be undergoing development, however.

This version of the SDPS/W will not be placed into production. The testing of the production capability of this version, however, will be as extensive as possible. A thorough review of the first draft documentation will also be done. (The motivation for placing great emphasis on testing and review of this version is to allow adequate time to make all necessary revisions. A large amount of other system and interface testing will occur in the final year before a satellite launch, and it is likely that only one full SSI&T effort for each instrument's SDPS/W can be accommodated then.)

7.5.2.3 Production-Ready Version (e.g., Ver. 2) Delivery

Several months to one year before launch, the complete, verified version of SDPS/W intended for use in production will be delivered to the DAAC for SSI&T. For some higher level standard products, however, this delivery may occur well after launch.

The testing of the production capability of this version is done to confirm that any deficiencies identified in the prior SSI&T of the Engineering Version of the SDPS/W have been corrected, and no new problems have been created. A thorough review of the second draft documentation will also be done.

7.5.2.4 Software Update Deliveries

Updates to software already in production will occur for reasons ranging from minor bug fixes to extensive redesign. If the fraction of code affected is small (e.g., $< 2\%$ of the executable source lines for a PGE) and the software changes do not involve changes in resource utilization, then a sufficient test is to confirm that the output of the updated PGE at the DAAC reproduces the values obtained at the SCF. More extensive modifications require more thorough testing as in Section 7.5.2.2, including perhaps testing of the PGE in parallel with the existing production PGE (see Section 7.8).

7.5.3 Acceptance Review

The SSI&T personnel prepare a report on the results of acceptance testing, and a log of the testing procedures, input data and output products. This report is reviewed by the SDPS/W developer and the DAAC management. An Acceptance Review is convened with participants (perhaps remote) including the DAAC management, a SDPS/W representative, ESDIS Science Software Manager (perhaps remotely), the SSI&T personnel and perhaps others. If the software is accepted, it will be transferred to commissioning status.

7.6 Commissioning New SDPS/W

Before attaining full production status, new production software will go through a commissioning period. The commissioning period is the DAAC production equivalent to the post-launch period of science algorithm and product validation done by a SDPS/W developer, done using actual instrument data and normal processing schedules, typically in the period immediately following launch and instrument checkout.

The purpose of the commissioning period is to discover and correct deficiencies in the SDPS/W related to production safety and reliability, and to validate the PGE database entries used by the Production Planner. The software is run repetitively or continuously to test the production reliability, receiving close scrutiny. The commissioning period for a given PGE will depend on how frequently it is run: the more frequently run, the shorter the commissioning period.

7.7 Operational Implementation of SDPS/W

Following the successful SSI&T of a SDPS/W delivery intended for production, there are additional steps necessary. These include:

- Assessing Impacts, including
  - Reprocessing of products generated by the upgraded SDPS/W
  - Reprocessing of other products for which the outputs of the upgraded SDPS/W are used as ancillary inputs
- Complete Production Operations Readiness Plan
- Conduct Operations Readiness Activities
- Conduct Operations Readiness Review (ORR)
- SDPS/W promoted to Production

(One requirement may be receipt of the final versions of the SDPS/W documentation before the SDPS/W is placed in production!)

7.8 Parallel Testing of Software Upgrades

For SDPS/W upgrades or enhancements involving more than 2% of the executable source lines for a PGE or that involve changes in resource utilization, a period of parallel testing with the older production software may be needed. During the period of parallel production, the outputs of the old and new SDPS/W are compared to ensure that the desired improvements have been achieved. Typically, the output files from both versions will be sent to the SCF.

The parallel-test is also used to discover and correct deficiencies in the SDPS/W related to production safety and reliability, and to validate the PGE database entries used by the Production Planner. The software is run repetitively or continuously to test the production reliability, receiving close scrutiny. The parallel-test period for a given PGE will depend on how frequently it is run: the more frequently run, the shorter the parallel-test period.

7.9 Code Fixes, Updates to Coefficient/Control Files

Patches to individual source code routines do not need as formal an SSI&T process as described above. The delivery procedures are simplified by the fact that the unchanged elements of the SDPS/W, test data, and documentation already reside at the DAAC. Updates to coefficient or control files only need a test to be sure that the new values are being used in production.

A sufficient test in both cases is just to confirm that the output of the updated PGE at the DAAC reproduces the values obtained at the SCF. The DAAC CM procedures must still apply, however.

7.10 Rapid Installation

At the DAAC Managers discretion, the procedures outlined above may be abbreviated in a circumstance where new SDPS/W needs to be installed rapidly or modifications to existing SDPS/W need to be made quickly. Such rapid installations of SDPS/W will be rare, however.

This page intentionally left blank.

8. Developer Interaction with Operational Science Software

In production, the SDPS/W developer will likely be involved with the science data product production in several ways. This section briefly discusses several of these interactions with the production SDPS/W. For external interfaces (between ECS/DAAC operations and SCF staff or SCF-provided software), the detailed descriptions, interface methods, contents, and formats of each data flow between the SDPS/W developer's SCF and the DAAC are found in the *ICD Between ECS and Science Computing Facilities (SCF)*. Internal data flows (between ECS/DAAC operations and ECS software at the SCF) are described in the ECS design documentation.

8.1 Coefficient Files

These files would be infrequently updated over the lifetime of the instrument and would be placed under CM at the DAAC (e.g., calibration coefficients). Although the coefficients in this category might be updated weekly in the immediate period following launch, the interval between updates will increase thereafter so that updates become infrequent. Coefficient files are delivered to the DAAC in the same manner as SDPS/W.

8.2 Ancillary Data Inputs Generated at SCF

This flow involves any files that may be generated by the SCF on a frequent and regular basis and sent to the DAAC for use in the normal product processing of instrument observations for a specific spatial or temporal domain. As such, these ancillary files would not be tested at the DAAC nor be placed under CM.

All ancillary data are sent to the ECS Ingest subsystem via interfaces documented in the *ICD Between ECS and Science Computing Facilities (SCF)*. These data will be archived at the DAAC, and the appropriate identifiers will be included in the product history for any products which use these data.

8.3 Production Status and Error Messages

Messages are generated by the SDPS/W, providing the developer with statistics and any information concerning the processing of each execution of a PGE. From this information, software and data problems can be isolated, fixes needed to correct software bugs can be identified, and enhancements to the SDPS/W indicated.

The SCF initiates a request for the processing status and errors using the ECS Client and receives a Distribution Notice (DN) email message. Using information provided in the DN, the SCF can then obtain the Status and other log files. SCFs can similarly request resource usage, production history information and data availability schedules from ECS. Alternatively, a SCF user can use a WWW interface to obtain processing status or resource usage.

Production History includes 1) the historical trace of a product as needed by the general user community to discriminate among possible candidate data collections or granules based on the inputs used in the processing, 2) information that must be captured in order to regenerate a granule or granules either during reprocessing or using a user's local processor, and 3) information that helps monitor the health of both the information system used to process the data and the software and ancillary data used in the processing.

8.4 QA/QC of Data Products

As noted in Section 4.4, the capability for QA is provided for every standard product PGE that runs at the DAAC. QA can be either automatic or manual (or both) and can be performed at both the DAAC and the SCF.

Automatic QA may be basic or advanced. Basic automatic QA means that the PGE tests the science data against standard conditions and records the results in the granule's QAStats metadata. Using algorithm-specific tests, the PGE may also record QA results in the AutomaticQuality flag. Advanced QA applies to a PGE which tests the science data against product specific conditions and populates the Product Specific Attributes (PSA) for the granule. There is also a mechanism by which supporting information for automatic QA may be captured for subsequent analysis.

Manual QA is applied to the troubleshooting of automated QA failures. Any product failing automated QA and its associated Production History and supporting information would automatically be sent to the SCF for further manual analysis. Manual QA is also used for routine data inspection for quality analysis at the SCF. The Instrument Team would review a representative sample of the data which passed the automated QA tests. Based on the results of the manual QA performed by the Instrument Team, the ScienceQualityFlag would be updated. In troubleshooting, this may be done on a a granule by granule basis. For routine inspections, the update may be made to the ScienceQualityFlag for all granules similar to the representative samples analyzed.

8.4.1 QA Data Subscription Request

This request is the mechanism by which a SDPS/W developer establishes a QA data subscription. A dialog between the SCF and the DAAC staff will be necessary to establish and update subscriptions.

The SDPS/W developer may subscribe against core metadata and Product-Specific Attributes (PSA). The QA Data Subscription Request service may be either a "notification", an email message indicating which data matches qualifiers, or an "acquire" service whereby data is automatically pushed to the subscriber's site (in this case, the SCF).

8.4.2 QA Data Subscription Event Notification and Acquire Services

An ECS-generated message is the mechanism by which the SCF is notified whenever the established subscription conditions have been encountered if the notification service was selected. If the acquire service was selected, the data is automatically pushed to the SCF pull area.

8.4.3 Data to QA

If the SDPS/W developer has requested the notification service, and decides to perform QA on a science data product, the SCF obtains the data product by using its Universal Reference in the QA Data Subscription Event Notification message. In this case, the SCF pulls the data to the SCF ftp pull area and performs QA upon the data received from ECS. If the SDPS/W developer has requested the acquire service, the data will have already been pushed to the SCF's ftp pull area.

8.5 Changing Production Scheduling Information

The information in the PGE database used to generate production plans includes the production rules for each PGE. Changes to the production rules may be delivered to the DAAC in the same manner as SDPS/W.

Production rules are instructions that drive how a particular PGE is to be run. Most of the production rules relate to how the PGE's input data is selected., but other production rules relate to conditions for activation of an instance of a PGE. In the former category, production rules include basic temporal, advanced temporal, orbit-based activation, period specification, most-recent data, alternate input selection, activation by tile. In the latter category, production rules include conditional activations such as: metadata-based, event-based, and intermittent execution. As well, a PGE may require a combination of productioin rules to be applied. For detailed information about production rules, the reader is referred to the June 1997 *Production Rules White Paper (Draft),* 445-WP-001-002, available on request.

8.6 Requesting Reprocessing

As discussed in Section 4.5, the EOS system supports the reprocessing of standard data products, their browse data products and metadata. A developer would likely request reprocessing for specific data, for example, to assess the impact of updated SDPS/W that has been installed into the production. The reprocessing request should be sent to the DAAC Production Scheduler as an email message or via an X11 interface. As noted earlier, however, requests for reprocessing of substantial portions of the archived input data will require an impact analysis and appropriate authorization.

8.7 Metadata Updates

In order for ECS, and later the DAACs, to update MCFs and ESDTs when the need arises, the SDPS/W developer will interact with the ECS to provide comments/suggestion for the product specifications. ECS will then generate updated MCFs and ESDTs in an iterative process as described in Section 5.5.3.

Appendix A. Getting Assistance

As a developer of SDPS/W that will be run at a DAAC, you are a user of the DAAC production services. Each DAAC has support personnel to provide assistance concerning operational issues. The User Services Office of the appropriate DAAC will be able to direct you to the science algorithm support group. The addresses, telephone, Internet addresses, and disciplines of the DAACs are provided in Section A.1.

For information about or questions concerning usage of the SDP Toolkit or other questions concerning SDPS/W development issues can be directed to the ECS support personnel indicated in Section A.2.

A.1. DAAC User Services

ASF DAAC - User Services Office (Discipline: Polar processes, SAR products)

Geophysical Institute

University of Alaska

903 Koyukuk Drive

P.O. Box 757320

Fairbanks, AK 99775-7320

Voice: 907-474 -6166

Fax: 907-474 -5195

Internet: asf@eos.nasa.gov

WWW Home Page URL:

http://www.asf.alaska.edu/

EDC DAAC User Services (Discipline: Land Processes)

EDC DAAC User Services

EROS Data Center

Sioux Falls, SD 57198

Voice: 605-594-6116

Fax: 605-594-6963

Internet: edc@eos.nasa.gov

WWW Home Page URL: http://edcwww.cr.usgs.gov/landdaac/landdaac.html

Goddard DAAC User Services Office (Disciplines: Upper atmosphere, global biosphere, atmospheric dynamics, geophysics)

Code 902.2

NASA Goddard Space Flight Center

Greenbelt, Maryland 20771

Voice: 301-614-5224 or 1-800-257-6151

Fax: 301-614-5268

Email:daacuso@eosdata.gsfc.nasa.gov

WWW Home Page URL: http://daac.gsfc.nasa.gov/

JPL DAAC User Services (Discipline: Physical oceanography)

Jet Propulsion Laboratory

Mail Stop 300-320

4800 Oak Grove Drive

Pasadena, CA 91109

Voice: 818-354-9890

Fax: 818-393-2718

Internet: jpl@eos.nasa.gov or podaac@jpl.nasa.gov

WWW Home Page URL: http://podaac.jpl.nasa.gov/

Langley DAAC User Services (Disciplines: Radiation budget, tropospheric chemistry, clouds, aerosols)

NASA Langley Research Center

Mail Stop 157D

Hampton, VA 23681-0001

Voice: 757-864-8656

Fax: 757-864-8807

Internet: larc@eos.nasa.gov

WWW Home Page URL: http://eosdis.larc.nasa.gov/

NSIDC DAAC User Services (Disciplines: Snow and ice, cryosphere and climate)

National Snow and Ice Data Center

CIRES, Campus Box 449

University of Colorado

Boulder, CO 80309-0449

Voice: 303-492-6199

Fax: 303-492-2468

Internet: nsidc@eos.nasa.gov or nsidc@kryos.colorado.edu

WWW Home Page URL: TBD (http://www-nsidc.colorado.edu/NASA/GUIDE/index.html

ORNL DAAC User Services (Discipline: Biogeochemical dynamics)

Oak Ridge National Laboratory

PO Box 2008, Mail Stop 6407, Bldg 1507

Oak Ridge, TN 37831-6407

Voice: 615-241-3952

Fax: 615-574-4665

Internet: ornldaac@ornl.gov

WWW Home Page URL: http://www-eosdis.ornl.gov/

A.2. Other Support

SDP Toolkit Questions

SDP Toolkit Manager

NASA ECS Project

Hughes Information Systems

1616 McCormick Drive

Upper Marlboro, MD 20774

Voice: 301-925-0781 (leave voice mail message)

Internet: pgstlkit@eos.hitc.com

Other SDPS/W Development and SSI&T Questions

Science Software Support

NASA ECS Project/Science Office

Hughes Information Systems

1616 McCormick Drive

Upper Marlboro, MD 20774

Voice: 301-925-0898

Internet: rplante@eos.hitc.com

Appendix B. Checklists

B.1 Before Starting Development

Contact the DAAC User Services and ECS support personnel to get the latest information on the ECS release and the SDPS/W SSI&T procedures. ❑

Install the latest version of the SDP Toolkit at your local development site(s). ❑

Make sure that each developer/development site has the same revision number of the SDP Toolkit. ❑

Establish local coding guidelines for all developers/development site(s). ❑

Confirm that the SDPS/W design meets the EOSDIS requirements with regard to ...metadata generation and formats ❑

The purpose of this Appendix is not to provide a list of everything that must be done to develop an algorithm to meet the EOS science requirements, but simply remind you of those things that are related to the ECS integration and test activity

- ESDT generation and formats ❑
- browse product generation and formats ❑
- standard product generation and formats ❑
- quality assessment on ingest of external ancillary data ❑
- quality assessment for output products ❑
- product dependencies ❑
- SDP Toolkit interfaces ❑
- production rules ❑

B.2 During Development

Discuss details of DAAC/DPS interfaces with the DAAC/ECS science algorithm support group. ❑

Hold software reviews as needed. ❑

Use the mandatory tools of the SDP Toolkit for all SDPS/W and PDPS interfaces. ❑

Use the optional tools of the SDP Toolkit as applicable. ❑

Check SDPS/W source code and scripts comply with ESDIS coding standards. Use code checking tools. ❑

Delineate the various processing scenarios for the SDPS/W during production....

check informally with the DAAC/ECS science algorithm support group ❑

review formally with project staff ❑

Document PDPS interface assumptions for delivery. ❑

Define Test Plans and Acceptance Test Specifications with the DAAC/ECS science algorithm support group. ❑

Produce all test files for delivery. ❑

Run tests at the SCF. ❑

B.3 Before Delivery

Use code checkers to check code for software standards compliance of final deliverable. ❑

Determine to what extent the development team will participate at the DAAC during SSI&T. ❑

Confirm delivery date one month in advance. ❑

Assemble delivery package. ❑

Confirm that all needed files are present in the package. ❑

Transfer delivery package to the DAAC. ❑

B.4 During Integration and Test

Perform stand-alone testing at the DAAC to verify rehosting. ❑

Organize resources at the SCF for verifying of acceptance test results. ❑

Support DAAC operational readiness testing and activities. ❑

Participate in Reviews related to the SSI&T. ❑

This page intentionally left blank.

Appendix C. Annotated Outlines for Deliverable SDPS/W Documentation

The annotated outlines in this appendix are intended to provide a guideline on the division of information between these documents. They are not intended as a rigid outline for any document.

C.1 System Description Document

The System Description Document (SDD) can be regarded as an “Operator’s Guide, Volume 1”, in which the structure of software system is described along with the high-level workings of the software system. The SDD is intended to provide the reader with a basic understanding of what the software system does and how it accomplishes this. It is recommended that the specific operations procedures be delineated in a separate *Operations Manual* (see C.2 below for suggested outline) so that they may be easily located by operations personnel needing to perform a specific action without the interference of extraneous material.

The SDD should also serve as a basic reference, pointing to the more detailed information provided in the other documents such as an Algorithm Theoretical Basis Document (ATBD). The SDD need not be written on a highly technical level.

1. Introduction

A one-page executive summary of the material in the remainder of this document is recommended.

1.1 Background

Describe the reason the software was developed and the design approach that was used.

Reference any other documents to assist the reader (e.g., ATBDs).

1.2 System Concepts

Describe functional design of the SDPS/W system. This includes the design methodology (i.e., structured or object-oriented), the processing paradigms employed (e.g., sequential, concurrent), the granularity of the input data and frequency of processing, and the data granularity of the processing (if different than that of the input data).

1.3 System Structure

An overview of how the SDPS/W system is organized in terms of an object model or subordinate subsystems. Also, describe the context, extent of use and origin of any heritage code or COTS that are used. Include a simple diagram or two to illustrate the description. It is recommended that a history of changes be placed in the Appendix (see below).

1.3.1 Processing Components

Introduce briefly here the major components or subsystems that comprise this SDPS/W. Describe briefly how these components are related to the PGEs. It is recommended that the details be put in Section 3 below, however.

1.3.2 Data Flow

Describe the flow of data through the SDPS/W system at a high level.

1.4 Operational Scenario(s)

Describe broadly the conditions under which the various PGEs are executed

1.4.1 Environment

List the computational, communications and human resource requirements needed to run the SDPS/W effectively and efficiently. Include programming and script languages employed.

1.4.2 Interfaces

Describe any needed access or interface with other SDPS/W systems, ECS components (e.g., data server), ancillary data, and QA.

1.5 System Documentation

Reference other documents related to this system and where they can be obtained.

2. SDPS/W Outputs

Brief descriptions of the various of temporary, intermediate, or product outputs.

2.1 Product Output 1

Describe the generation and purpose or use of this item . Include product dependencies and metadata associated with this product output. Refer to *Processing Files Description Document* (see C.3 below for suggested outline) for file format details.

2.1.1 Scientific Basis

A very brief description of the scientific aspects, referring to other documents (e.g., ATBDs) for more detailed information.

2.1.2 Inputs

Describe the primary data input files used in generating this product.

2.1.3 Quality Assurance

Describe at an overview level how the product's integrity is maintained, and the conditions under which automated in-line QA, manual inspection at DAAC, and/or manual inspection by SCF are performed. The specific DAAC and SCF QA procedures should be delineated in the *Operations Manual* .

2.2 Product Output 2

As in 2.1 above.

3. Product Generation Executives (PGEs)

Identify and describe the subsystems and/or PGEs

3.1 PGE 1 (Subsystem 1)

A description of the processing performed by this PGE and key implementation decisions.

3.1.1 Purpose

What does execution of this PGE accomplish?

3.1.2 Structure

Describe the processing steps and binary executables which comprise this subsystem.

3.1.3 Data Files

Provide a tabular list of the input, output, and temporary files. Include the run-time logical identifiers, ESDT ShortNames, source (if input) and brief description (a few words) .

3.1.4 Production Rules

What are the conditions under which this PGE is executed?

3.2 PGE 2 (Subsystem 1)

Follow the guidelines as in 3.1 above.

3.3 PGE 3 (Subsystem 2)

Follow the guidelines as in 3.1 above.

4. System Performance

Describe nominal performance measures for each PGE.

4.1 Performance Factors

What criteria are appropriate to check that a PGE is performing nominally?

4.2 Resource Utilization

Include both SCF and DAAC values for disk, memory, cpu, computer type/class, if known.

4.2.1 Processing Time

Nominal ranges for CPU and wall clock

4.2.2 Memory Requirements

Direct access storage, shared memory, maximum resident memory size (from Rusage utility) ...

5. System Operation

What types of things will an operator be required to do for each PGE? Give a summary, but refer the reader to the *Operations Manual* for details.

Acronyms

Appendix: Change History

Structural changes are listed here (preferably in tabular form) with the date, version number and a brief explanation given for the change.

C.2 Operations Manual

The *Operations Manual* can be regarded as an "Operator's Guide, Volume 2", delineating the operator interfaces and specific operations procedures to be followed by the DAAC staff for the manual activities which must be either routinely performed, or performed in reaction to specific circumstances. This document needs to be written for an audience that will use it as a technical tool. It is recommended that this document be first written by the SDPS/W developer, and then annotated by the DAAC operations personnel.

It is recommended that the description of what the software system does and how it accomplishes this be put in the SDD (see C.1 above). Only specific, step-by-step operations procedures are recommended to be included here, permitting them to be easily located by operations personnel without the interference of extraneous material when needing to perform a specific action.

1. Introduction

The version of SDPS/W for which this document applies needs to be stated here, along with an overview of the document organization. This introduction may also provide a brief overview of the SDPS/W system, but the reader should be directed to the SDD for description of the SDPS/W.

Include a brief summary of DAAC-SCF operations activities, interfaces and roles/responsibilities. Provide points-of-contact (POCs), including telephone numbers for voice and fax as well as email addresses.

2. Routine procedures

Describe those procedures that need to be performed on a regular basis. Include screen layouts, menu selections and command language as appropriate.

2.1 Configuration Management issues

Identify any unique issues related to CM of source code, coefficient files, test data, test plans and test outputs. Suggest levels of change control for the various elements.

2.2 Installation

Identify instructions, procedures and/or scripts to be used for build/install of SDPS/W and any COTS, as well as setup/configuration. Include any libraries. It is recommended that a list be included or pointed to which gives all files, current version number and size (bytes).

2.3 Monitoring Procedures (Production and any QA)

What production and QA monitoring by operations personnel is required (e.g., execution time, disk space, data visualization). Include appropriate guidance instructions regarding limits, and trends.

2.4 Intermediate File Backup Procedures

Delineate frequency of backup of any intermediate files that are used for processing subsequent data.

2.5 Instrument Reflight Instructions

If the reflight of an instrument uses this same SDPS/W system, what accommodations must be made for processing data from each or both instruments.

3. Reactive Maintenance

Those procedures that are performed in reaction to operational problems are described.

3.1 Restart Instructions

Delineate the steps that need to be performed to restart the PGE or the entire SDPS/W system following (1) a PGE crash, (2) a PDPS crash and (3) a system crash. Include actions involving temporary files, intermediate files, and log files.

3.2 Shutdown Instructions

What steps are required to shut the SDPS/W system down gracefully? How should the SDPS/W system be shut down in a hurry?

3.3 Supplied Troubleshooting Tools

List and describe any tools that have been supplied to aid in troubleshooting

3.4 Error Handling Procedures

List any normal anticipated errors and how to correct them. Include a table of ALL possible error codes, what each means, and what action to perform. List all log files and what information can be found in each. Each log entry type needs to be documented, along with a description of each field and what it can contain.

C.3 Processing Files Description Document

This document details the file and record layouts for each major subsystem of the SDPS/W system.

1. Introduction

The version of SDPS/W for which this document applies needs to be stated here, along with an overview of the document organization. This introduction may also provide a brief overview of the SDPS/W system, and the reader directed to the SDD for description of the SDPS/W. A high level data flow diagram is helpful here.

2. Data Sets - External (i.e., produced by someone else)

Provide a list of the external input or ancillary data sets. Briefly describe their functions within the system.

2.1 Data Set Description

List and describe the data sets. This description will be at a level to contain the organization, and the physical and logical record lengths.

2.1.1 Data Set 1

Describe the data set attributes in a consistently followed format. Define (byte level) the file header information and data record descriptions. Indicate scaling factors and units.

If an HDF file or HDF-EOS file, describe instead the data model and attributes s.

2.1.2 Data Set 2

Follow the guidelines from data set 1 until all external data sets have been described.

...

3. Data Set - Internal (i.e., produced by this SDPS/W)

Follow the outline for Data Set - External. This section would include temporary files, intermediate files and product output files. If UNIX pipes are used in a PGE, delineate the information contained in each standard output record format.

If there are no internal data sets, include the section but write

NONE.

4. References

Describe any references made in the document in a bibliographic format. If you are going to include hard copies, do so at the end of the document and use a divider or a tab to mark them.

C.4 Test Plans

Description of the testing at the DAAC for safety and production reliability. It is encouraged that the various tests be written with a common look and feel for operator actions/commands.

0. Test Plan Description

Describe overall goal and success criteria of this test plan

0.1 Test 1

Describe goals of this individual test.

0.2 Test 2

Describe goals of this individual test.

...

1.0 Test 1

1.1 Test Description

1.1.1 Specific Test Instructions

1.1.2 Success Criteria

1.1.3 Estimated Time to Conduct Test

1.1.4 Participant, Roles & Responsibilities

1.2 Test Environment (i.e., resources requirements, assumptions , and stubs)

1.3 Necessary Input(s) and descriptions

1.4 Expected Output(s) and descriptions

1.5 Allowable Precision

1.6 Script(s) used

2.0 Test 2

(same information as for Test 1 above)

This page intentionally left blank.

Abbreviations and Acronyms

ANSI	American National Standards Institute
ASF	Alaska SAR Facility (DAAC)
ATBD	algorithm theoretical basis document
CM	configuration management
COTS	commercial off-the-shelf (hardware or software)
CPU	central processing unit
CSMS	Communications and Systems Management Segment (ECS)
CSS	communications subsystem
DAAC	Distributed Active Archive Center
DAP	Delivered Algorithm Package
DCE	distributed computing environment
DN	distribution notice
ECS	EOSDIS Core System
EDC	EROS Data Center (DAAC)
EDOS	EOS Data and Operations System
EOC	EOS Operations Center
EOS	Earth Observing System
EOSDIS	EOS Data and Information System
ESDIS	Earth Science Data and Information System Project (GSFC)
ESDT	Earth Science Data Type
FOS	Flight Operations Segment
GSFC	Goddard Space Flight Center
GUI	graphical user interface
HDF	hierarchical data format
HTML	hypertext markup language
ICC	instrument control center

ICD	interface control document
ID	identifier
IP	international partners
IPGS	international partner ground system
IWG	Investigator Working Group
I&T	integration and test
JPL	Jet Propulsion Laboratory
LaRC	Langley Research Center (DAAC)
MCF	metadata configuration file
MIL-STD	military standard
MSS	management subsystem
MTPE	Mission to Planet Earth
NASA	National Aeronautics and Space Administration
NOAA	National Oceanic and Atmospheric Administration
NSF	National Science Foundation
NSFNet	NSF Network
NSI	NASA Science Internet
NSIDC	National Snow and Ice Data Center (DAAC)
OPS	operations
ORNL	Oak Ridge National Laboratory (DAAC)
OR	operations readiness
ORR	operations readiness review
PDPS	Planning & Data Processing Subsystems
PDR	Preliminary Design Review
PGE	product generation executive
PGS	Product Generation System
PI	principal investigator
POSIX	Portable Operating System Interface for Computer Environments
QA	quality assurance

QAE	quality assessment executive
QDS	quicklook data set
RPC	remote procedure call
SCF	Science Computing Facility
SDD	system description document
SDPF	Sensor Data Processing Facility (GSFC)
SDPS	Science Data Processing Segment (ECS)
SDPS/W	science data production software
SDSRV	Science Data Server
SMC	System Monitoring and Coordination Center (ECS)
SSAP	Science Software Archive Package
SSI&T	Science Software Integration and Test
TDRSS	Tracking and Data Relay Satellite System
TL	team leader
TRMM	Tropical Rainfall Measuring Mission (joint US-Japan)
TSDIS	TRMM Science Data and Information System
UR	Universal Reference
VOB	versioned object base
WWW	WorldWide Web

This page intentionally left blank.

Glossary

Word or Phrase	Definition
ASCII File	A data file whose contents are encoded as ASCII characters.
Authorized User	Any user of ECS who has obtained authorization from the system to access certain restricted files or other parts of the system.
Binary File	A data file whose contents are in binary form (i.e., not encoded).
Code Analysis	Static/dynamic code checking for memory leaks, argument list mismatches, code coverage, and other coding errors.
Coding Guideline	Recommended, but not required, programming style.
Coding Standard	Required programming style.
Color Table	Table mapping pixel values to colors.
Configuration Management Tool	Software tool for doing automated configuration management of source code, scripts, documentation, and other computer files.
CPU Time	Elapsed time spent by a processor doing some task, not counting idle time (as opposed to wall clock time, which includes idle time).
DAAC SSI&T Environment	The hardware (processing platforms, other workstations, etc.) and software (compilers, linkers, diagnostic tools, etc.) used to integrate and test the science software at the DAAC.
DAAC SSI&T Staff	The personnel actually involved in doing the integration and test of the science software at the DAAC. This includes DAAC personnel, SCF developers who are helping with SSI&T, etc.
Data Server	Either the Data Server subsystem as a whole, or a specific instance of a data server. A data server is a (hardware/ software) entity which accepts, stores, and distributes EOS (and other) data, for both other subsystems within ECS and external users.
Data Type	A particular type of data handled by a particular data server. An example of a data type might be MODIS level 1a products, etc.

Data Visualization	The ability to examine a data file as a raw data dump, a plot, or an image.
Delivered Algorithm Package	An ensemble of PGE source code, control files, makefile, documentation, and other related files delivered in a package from the SCF to the DAAC.
Disk Space Usage	The amount of disk space that a process requires to run. This includes in particular the amount of space required to read and write any input, intermediate, or output data files.
Distribution Notice	An email message from the ECS to a data product subscriber notifying of the availability of the requested data. This notice contains the Universal Reference to the requested data.
Earth Science Data Type	The formally specified set of attributes and services that define and characterize a given ECS science data collection.
Edge Enhancement	A feature enhancement technique which enhances the display of outlines of objects in an image.
Electronic Mail	The ability to send and receive messages across the Internet.
Feature Enhancement	Various techniques for enhancing the display of an image so that certain types of features are more easily seen.
HDF File	A data file whose format follows the NCSA Hierarchical Data Format standard, as well as ECS-developed extensions thereto.
I/O Access	A read or write by a process to a data file.
Integration and Test Log	An ongoing record of integration and test activities.
Internet	A collection of government, educational, and commercial networks which cooperate to allow data transfer between their users. The Internet evolved out of an experimental DOD network called ARPAnet.
Make File	Input file for the UNIX make facility, which allows a user to specify dependencies between source (and other) files, primarily for compiling and linking programs, but also used for other activities, such as building documentation and installing software.
Memory Leak	An erroneous use of memory by a program, such as reading from an uninitialized address, or writing to an address that has not been allocated.
Memory Usage	The amount of memory that a process requires to run.

Metadata	"Data about data", used to facilitate database searches. Types of metadata include: product metadata (data describing a particular product, such as when it was generated, etc.) and algorithm metadata (data describing science software).
PGE	Product Generation Executive. The smallest scheduled unit of science software. A PGE may consist of a single binary executable or a large number of executables wrapped by a shell script. To the ECS system, a PGE is a completely black box.
PGE Database	The database of information about the science software, organized by PGE, used by the Planning and Data Processing subsystems to execute the PGEs. Includes disk space and memory requirements, estimated run time, etc.
PGE Installation Package	A single record in the PGE database, containing all of the required information about a PGE.
Problem Tracking Tool	Software tool for doing automated tracking of problems found in the science software, as well as the status of the resolution of those problems.
Procedure	A subroutine, function, or other module within a process.
Process	An executing program.
Processing Platform	A computer used to run science software. (As opposed to miscellaneous workstations and PCs used by the SSI&T personnel for other purposes.)
Product	A permanently archived set of output data generated by the science software.
Profiling	The measurement of the science software's resource usage requirements, such as memory and disk space requirements, CPU time, etc.
Report	Documentation of some automated (such as standards checking) or manual (such as evaluation of a science software delivery) activity.
Report File	An electronic copy of a report.
Runs Reliably	The software runs to normal completion repeatedly over the normal range of data inputs and run-time conditions.
Runs Safely	The software executes without interfering with other software or DAAC operations.

Science Software	Software developed at the SCFs to generate standard (and other) science data.
Science Software Archive Package	An ensemble of the latest version of a PGE source which has passed SSI&T, control files, makefile, documentation, and related files packed into a tar file and loaded onto the Data Server.
Science Software Delivery	All of the files delivered by the SCF to the DAAC which are associated with a particular set of science software. This includes: source code, include files, shell scripts, make files, documentation, test procedures, test inputs and outputs, etc.
Science Software Documentation	Documentation associated with a particular set of science software, in electronic form.
Science Software Script	Any shell script which is part of the science software and which must be run in order to produce output products (as opposed to scripts which are supplied to help in the integration and test process).
Science Software Source Code	Any source code which is part of the science software and which must be run in order to product output products (as opposed to code which is supplied to help in the integration and test process).
Screen Capture	A facility for taking a ""snapshot"" of what is being displayed on a workstation or PC, and saving the snapshot to a file which can then be displayed or printed via a paint program.
Standards Checking	The process of checking whether or not source code and shell scripts follow prescribed coding standards.
Subscription	A request for an action to be taken whenever a particular event occurs. Usually, a request to be notified whenever certain specified data product is produced or received under certain specified conditions.
Universal Reference	An identifier within ECS uniquely referencing an accessible element.
WWW Browser	Software (local or remote) which allows a user to access the WWW either textually or graphically. WWW (World Wide Web) is a mechanism for connecting Internet via a set of hypertext documents.