308-CD-001-008

EOSDIS Core System Project

SDPS Software Development Plan for the ECS Project

January 2001

Raytheon Company
Upper Marlboro, Maryland

SDPS Software Development Plan for the ECS Project

January 2001

Prepared Under Contract NAS5-60000
CDRL Item #049

RESPONSIBLE ENGINEER

Mary S. Armstrong /s/ 01/29/01

Mary Armstrong Date
EOSDIS Core System Project

SUBMITTED BY

William Knauss /s/ 01/29/01

Will Knauss, Director Development Date
EOSDIS Core System Project

Raytheon Company
Upper Marlboro, Maryland

308-CD-001-008

This page intentionally left blank.

Preface

This document is a formal contract deliverable with an approval code 2. As such, the Government reserves the right to request changes within 45 days of the initial submittal. The last revision of this document was completed in September 1999 and .contained information about the development organization that has since changed. This version documents the current software development methodology, planning, and management as it exists on the ECS SDPS program today. Any future changes in the software development process will result in an update to this document and resubmittal to the Government

Future changes to this document shall be made by document change notice (DCN) or by complete revision. Any future changes must be reviewed and approved by the Government.

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, Maryland 20774-5301

This page intentionally left blank.

Abstract

The ECS SDPS Software Development Plan (SDP), CDRL item 049, DID 308/DV2, defines the steps by which the development of ECS SDPS software will be accomplished and the management approach to software development. The SDP addresses software processes, methods, organizational responsibilities, tools, configuration management, software quality, and other activities relevant to accomplishment of the ECS SDPS statement of work. The SDP describes software development processes at a summary level and makes extensive reference to the collection of ECS SDPS Project Instructions (PIs). The PIs provide details for: 1) processes, such as metrics collection and inspections; and 2) project standards, such as the format and content for software development files (SDFs) and coding standards. The intent is for this document to provide the overall high-level process and the PIs and Work Instructions (WIs) to provide the detailed instructions on how ECS SDPS executes this process.

This plan addresses the processes used by the ECS Science and Data Processing Segment project. The ECS Mission Operations Segment (EMOS – formerly FOS) has a different software development life cycle and is not addressed as part of this plan.

Keywords: software, process, development, training, CASE, metrics, standards

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Submitted as Final
iii through xii	Submitted as Final
1-1 through 1-4	Submitted as Final
2-1 through 2-2	Submitted as Final
3-1 through 3-14	Submitted as Final
4-1 through 4-14	Submitted as Final
5-1 and 5-6	Submitted as Final
6-1 through 6-4	Submitted as Final
AB-1 through AB-2	Submitted as Final
GL-1 through GL-6	Submitted as Final

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
193-308-DV2-001	Pre-Approval	November 1993	
193-308-DV2-001	Original	May 1994	94QC-0021
308-CD-001-003	Original	December 1994	94-0186
308-CD-001-004	Final	July 1995	95-0459
308-CD-001-005	Final	November 1995	95-0837
308-CD-001-006	Submitted as Final	July 1996	96-0506
308-CD-001-007	Submitted as Final	September 1999	99-0864
308-CD-001-008	Submitted as Final	January 2001	01-1147

This page intentionally left blank.

Contents

4. Software Development Process

5. Software Quality

6. Software Configuration Management

List of Figures

List of Tables

Abbreviations and Acronyms

Glossary

1. Introduction

1.1 Identification of Document

This Software Development Plan (SDP), Contract Data Requirements List (CDRL) Item 049, whose requirements are specified in Data Item Description (DID) 308/DV2, is a required deliverable under the Earth Observing System Data and Information System (EOSDIS) Core System (ECS), Contract (NAS5-60000).

1.2 Scope of Document

The ECS SDPS SDP outlines the steps by which the development of ECS SDPS software will be accomplished and the management approach to software development. The SDP addresses software processes, products, methods, organizational responsibilities, tools, configuration management, software quality, and other activities relevant to accomplishment of the ECS statement of work. Overall, the plan for ECS SDPS software development consists several documents:

- The ECS SDPS SDP – discusses software development processes at a summary level
- ITS policies and directives (available on the Raytheon ITS web page) – prescribes practices that apply to the Raytheon ITS business unit
- ECS Project Instructions (PIs) and Work Instructions (WIs) (available on the ECS Internal Server) – provide details of how Development and other processes on the ECS project are executed
- Related Project Documentation (listed in Section 2 of this document) – provide additional information about the ECS Project, the software product, and related processes
- Baselined schedules and budgets maintained for the ECS project (available from the Program Office or Program Controls Department) – provide up to date status regarding the cost and schedule of software products under development.

These documents are applicable to all software development processes and standards on the ECS SDPS project unless a formal waiver identifying any deviation or exception is documented and approved. Note that ECS PIs and WIs take precedence over ITS policies and directives since the ECS PIs reflect the tailoring of IPDS and Raytheon SOIs to the ECS program, as well as specific contractual obligations.

- This document discusses software development processes at a summary level.

This plan addresses the processes used by the ECS Science and Data Processing Segment project. It covers the life-cycle and process for all the ECS SDPS releases. The EMOS has a different software development life cycle and is not addressed as part of this plan.

1.3 Purpose and Objectives of Document

The ECS SDPS SDP describes the processes the ECS SDPS project will use to develop and document the ECS SDPS software. It provides a systematic approach to software development, using NASA Software Documentation Standard, NASA-STD-2100-91, to tailor those software engineering practices to specifically meet ECS SDPS needs. The plan is used by the Government to monitor the procedure management, and contract work effort of the organizations performing software development.

The Raytheon Systems Company (RSC) processes, Integrated Product Development System (IPDS), . Software Operating Instructions (SOIs), and Raytheon ITS processes were used to generate the process documented in this plan. By using RSC standard tailoring procedures, ECS SDPS was able to generate this process quickly. The RSC tailoring process provided an easy way to identify where ECS SDPS processes fit in an overall system development process.

1.4 Document Status and Schedule

The final version of this ECS Software Development Plan was submitted to the Government in July of 1996 as an approval code 2 document. An updated version was submitted in 1999 to reflect significant changes in the ECS SDPS software development process and life cycle. This update provides clarification in specific areas to align our documented process with the Capability Maturity Model (CMM). This document will be reviewed with every major ECS Release, but may also be updated at other times if there is a need. This document does not require formal Government acceptance. The document is under configuration control, and the Raytheon approval authority is the Director of Engineering.

1.5 Documentation Organization

The contents of the document are as follows:

- Section 1: Introduction - Introduces the ECS SDPS SDP scope, purpose, objectives, status, schedule, and document organization.
- Section 2: Related Documentation - Provides a bibliography of reference documents for the ECS SDPS SDP organized by parent, applicable, and information subsections.
- Section 3: Software Development Management - Describes the planning associated with software development management activities. It includes discussions of roles and responsibilities, schedules and milestones, the development environment, and metrics.
- Section 4: Software Development Process – Describes the ECS SDPS development process including the software development life cycle and software reuse.
- Section 5: Software Quality Assurance - Summarizes the approach to ensure that all software meets the Performance Assurance Requirements (PAR) and the Performance Assurance Implementation Plan (PAIP) for ECS SDPS.

- Section 6: Software Configuration Management - Summarizes the approach to ensure that all software is developed and controlled by established configuration management practices and procedures.

This document avoids duplicating detailed information found in other documents. Even though this is a development plan, specific dates are generally absent. The ECS SDPS Master Schedule is a living schedule maintained in the Primavera Scheduler tool. It is the source of the most accurate and up to date schedule for the ECS SDPS Program. The baselined schedule is available from the Program Office.

This page intentionally left blank.

2. Related Documentation

2.1 Parent Documents

The following documents are the parents from which this document's scope and content derive:

101-CD-001	Project Management Plan for the EOSDIS Core System
423-41-01	Goddard Space Flight Center, EOSDIS Core System (ECS) Statement of Work
423-41-02	Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System (ECS)
423-41-03	Goddard Space Flight Center, EOSDIS Core System (ECS) Contract Data Requirements Document
NASA-STD-2100-91	NASA Software Documentation Standard

2.2 Applicable Documents

The following documents are referenced herein and are directly applicable to this plan. In the event of conflict between any of these documents and this plan, this plan shall take precedence.

102-CD-003	Release Configuration Management Plan for SDPS
214-CD-002	Security Plan for the ECS Project
305/DV2	Segment Design Specification for the ECS Project (Release 5Aand subsequent releases)
311/DV2	Subsystem Database Design and Database Schema (Release 5A and subsequent releases - all subsystems)
334/DV1	Science System Release Plan (Release 5A and subsequent releases)
335/DV2	ECS COTS Deployment Plan
409/VE1	ECS Overall System Acceptance Test Plan (Release 5A and subsequent releases)
609/OP1	Operations Tools Manual (Release 5A and subsequent releases)
905-TDA-001	ECS System Baseline Specification

2.3 Information Documents

The following documents, although not directly applicable, amplify or clarify the information presented in this document, but are not binding.

NA

3. Software Development Management

3.1 Overview

This section describes the planning associated with software development management activities. It outlines the organizational roles and responsibilities identified to accomplish software development tasks within the ECS SDPS project (3.2). It identifies the process for documenting and disseminating information about the software development schedule (3.3). The training plan for personnel who will be responsible for, or support, software development (3.4) is provided. A description of the software engineering environment (a collection of integrated hardware and software tools) used to automate and support software development processes (3.5) is described. The metrics program, which supports managing (by monitoring and assessing) the software process (3.6), is described. The security planning (3.7) is outlined.

3.2 Authority, Roles, and Responsibilities

The ECS SDPS Project is composed of one software development organization, the Development Department. However, the activities of software development span multiple organizations within the ECS SDPS project. The authority, roles, and responsibilities of the technical staff responsible for software releases within the ECS SDPS Project are described below. The organization charts are posted on a periodic basis to the ECS Internal Server. Although, the following sections provide a discussion of the roles and responsibilities of the organization, the organization charts provide the details of where individuals report and the underlying structure of the organizations discussed here.

3.2.1 Project Manager

The ECS SDPS Project Manager is responsible for ensuring that all the ECS SDPS organizations involved in producing the software product are synchronized in their planning and execution of the process. The ECS SDPS Project Manager is the authority for making cross-functional organization decisions.

3.2.2 Development Organization

The ECS SDPS Development Department is responsible for executing the software development process to produce and integrate the custom software and deliver the product to the System Verification group. The ECS SDPS Development Director is considered the “software lead” on the project. As the software lead, the Development Director . is responsible for governing all software development-related processes, and for the development of software plans. For example, if Science Data Engineering (SDE) is developing code that will be delivered to the Distributed Active Archive Centers (DAACs), then SDE personnel are responsible for following those software development processes applicable to writing custom software. The Development Director or his designee is the chair for the program level SEPG, as the software lead of the

program. Also, the ECS SDPS Development Director is responsible for ensuring that software development plans and processes are synchronized with and supported by plans developed in other portions of the program organization. This includes, for example, QA plans, CM plans, and Systems Engineering plans such as the Science System Release Plan (SSRP).

3.2.2.1 Subsystem Groups

The Development Department is organized intoGroups, which generally equate to subsystems or collections of subsystems that correspond to particular WBS Elements. For instance, the Science Data Server traces to WBS 4.4.3, whereas the Planning Subsystem traces to WBS 4.4.5. In some cases, to balance the organization, WBS elements are shared across groups. Each group has a lead who is responsible for coordinating all development activities within the group.. This coordination includes technical as well as cost and schedule, and includes responsibility for L4 requirements definition, design, code/unit test, and production of deliverable documentation resulting from these activities. The groups provide support for system integration, which is managed by the Construction Office. The Group Leads interface with the Development Department Director, Systems Engineering, and other organizations.

3.2.2.2 Construction Office

The Construction Office controls and manages the system integration. This integration includes new functionality for each release in addition to any integration that must occur for patches to existing releases. The Construction Office controls the integration budget as well as the actual execution of the integration. Subsystems provide personnel to perform the integration under the supervision of the Construction Office personnel. The Construction Office is responsible for determining when a release or patch is ready to be turned over to the System Verification group for system testing. The Construction Office also manages cross-subsystem / group coordination within Development. The coordination that it supports includes Rough Order of Magnitude estimates (ROMs) in association with new scope (implemented through configuration change requests), participation in project CCBs, metrics collection across Development, documentation generation, and cross-subsystem detailed design issues. The Development Engineering organization facilitates problem solving at a finer grained level (i.e. , the coding level rather than an architectural level) than the Systems Engineering organization.

3.2.3 Systems Engineering

The Systems Engineering Department (SED) plays a role in a variety of software processes. The details of their roles and responsibilities are outlined in the following sections. These are not the only roles played by Systems Engineering. This is just the list of roles that System Engineering plays in the *software life cycle.*

3.2.3.1 Requirements Engineering and Architects Office

Systems Engineering and specifically, the Architect's Office (AO), is responsible for providing technical clarification of the system level requirements (L3s) through the use of operational concepts, high-level scenarios, technical directives, and trade-off studies. These techniques aid in isolating the applicable subsystem(s). The AO is responsible for performing the allocation of L3 requirements to the subsystems and providing draft detailed level requirements (L4). The Development Department finalizes the L4 requirements. Systems Engineering is then responsible for ensuring that the L3s are covered by the L4s through participation in peer reviews, as well as, traceability checks via the requirements database.

3.2.3.2 Configuration Management

ECS Configuration Management performs the daily management of the ECS system software, hardware, and documentation baselines necessary to develop, implement, test, and maintain the system. The ECS Configuration Management Office is responsible for maintaining and administering the ECS software and hardware baseline. This is the organization responsible for performing Software Configuration Management (SCM) on this project.

3.2.3.2 Systems Verification and Acceptance Test

Systems Engineering is responsible for execution and oversight of system level testing. This includes testing the system in the Verification and Acceptance Test Center (VATC) and the Performance Verification Center (PVC). The Development Department integrates the software and then turns it over for system level testing. Systems Engineering tests the system in . DAAC-like environments in the Landover facility, the VATC and the PVC. It is responsible for ensuring that requirements are tested, .that the system will work when fielded to the DAACs, and that the system meets performance specifications. This system level testing is also called acceptance testing since Systems Engineering is accepting the deliverable on behalf of the customer and verifying system requirements and reporting verification status at the Consent to Ship Review (CSR). After successful CSR, the system is fielded to the DAACs. At the DAACs SE is responsible to ensure that the system works according with the liens and limitations determined at the CSR.

3.2.3.3 COTS Upgrades

Systems Engineering leads the COTS Upgrade process with support from the Development Department to integrate the COTS upgrade in existing custom software baselines. The procedures covering the life cycle of upgrading a COTS product are provided in the DID 335, ECS COTS Deployment Plan. The process includes the requirements analysis that will initiate an upgrade activity, the reviews and sign off review boards utilized along the way as checkpoints/milestones to insure accuracy, adequate verification, and coordination with all ECS segments, customer activities, and DAACs that will be the recipient of the upgrades.

3.2.4 Maintenance and Operations

The ECS SDPS M&O Manager is responsible for the overall execution of all ECS SDPS activities related to ECS SDPS software delivered to the DAACs. Key M&O management responsibilities include coordination with ESDIS, DAAC management, and science working groups; establishing M&O policies and procedures; facilitating cross-DAAC information sharing and problem resolution; managing the sustaining engineering budget; coordinating patches and maintenance releases to already fielded baselines; and maintaining the ECS Operations Plan (DID 608). The main interaction with M&O for new software development is the communication that must occur between Development and M&O to provide the information required to update the M&O procedures.

3.2.5 Quality Assurance

The Quality Assurance organization is responsible for ensuring through an audit process that the software development process outlined in this plan as well as the PIs are followed. For more information on Quality Assurance, see section 5 “Quality Assurance”.

3.2.6 Software Engineering Process Group (SEPG)

The SEPG is a group established to define and refine the software development process for ECS SDPS. This includes all processes that influence the software development processes. For example, CM processes may not be directly software development related, but they may dictate how software is developed, thus they are directed by the SEPG. The SEPG is comprised of representatives from each organization, Systems Engineering, Quality Assurance, Science Data Engineering, Development, and Maintenance and Operations. The Development Department Director or their designee chairs the SEPG.

The SEPG is the central authority for software methodology, processes, and standards that span all ECS SDPS organizations and affect the development or deployment of software products. The SEPG regularly reviews software development processes and standards for the need to update, based on audit and metric reports. The SEPG will meet as required to address software engineering process issues that span the software development life cycle. The SEPG is a coordinating body, which ensures that each organization knows what processes they are responsible for and the scope of each process. The SEPG does not define the low level details of each process but commissions the organizations to define or refine the process and reviews the resulting PIs or WIs.

The SEPG is also responsible for the evaluation of process oriented lessons identified during a release and modification of appropriate PIs and the SDP, if required, to ensure that the lessons learned are incorporated in the defined ECS SDPS software development process and implemented in subsequent releases.

3.3 Software Planning

A planning cycle for each release occurs prior to the completion of the previous release. A draft Science System Release Plan (SSRP) is provided which describes the system requirements (L3s) and system capabilities. A system capability is a grouping of Level 4 requirements defined in order to be able to track a set of requirements throughout the development of the release. A capability ID is used as a code in the master schedule and estimates for level of effort are defined at the capability level.

After the draft SSRP is available, the capabilities are detail planned. An estimate of the source lines of code (SLOC) is generated. The SLOC estimate includes new, modified, and reused lines of code. An estimation model converts the SLOC estimates into level of effort estimates for requirements, design, code and unit test, and integration. These estimates are used to plan for staffing needs as well as to use as a basis for generating the detailed level schedule. The model is based on experience gained during prior releases. Finally, the maintenance of the release is also estimated based on the SLOC and past experience with the number of non-conformance reports (NCRs) per 1000 SLOC and the number of hours to fix an NCR. The estimates are retained for planning the next release.

3.4 Schedule and Milestones

Detailed schedules, including major milestones, are maintained in the ECS SDPS Master Schedule. These schedules are release oriented and contain software development activities. The ECS SDPS Master Schedule contains the planned software development schedule for each release. The Master Schedule was developed from the spacecraft launch dates and the EOSDIS Ground System Integration dates for the EOS Terra, Aqua, and Landsat 7 missions.

Each release begins with a planning period in which the requirements are analyzed and the schedules “baselined”. The ECS SDPS Intermediate Logic Network (ILN) was developed from the ECS SDPS Master Schedule. The ILN is available electronically through the ECS SDPS scheduling tool. It provides a logic network of the software development, hardware procurement, integration, and acceptance test activities for each ECS SDPS Release. The ILN provides schedules down to the software component level for the design, implementation, and integration phases of software development. It provides sufficient detail for critical path and float analyses for each release.

Detailed descriptions of the content of each release are provided in the Science System Release Plan (SSRP - DID 334) delivered prior to the requirements phase of each release. It provides a listing of the capabilities scheduled for each release in addition to the Level 3 (L3) requirements that will be satisfied or partially satisfied in the subject release.

Table 3-1 contains the major reviews for each release and their definitions. Some of these reviews are with the customer in attendance and others are internal only. The common thread among them is that they are focused on one release at a time. In other words, the IRR is not focused on all the requirements from now to the end of the project, but only for a particular release.

Table 3-1. Major Reviews and Their Definitions

Review	Review Definitions
Incremental Release Review (IRR)	At the conclusion of the Preliminary Design phase an IRR is conducted, to promote a common understanding between the ECS SDPS project and ESDIS of the capabilities that ECS SDPS must provide. This review includes an understanding of the requirements through use case scenario presentations.
Test Readiness Review (TRR)	Conducted after the Software Turnover Meeting, that is after the software has been turned over from integration testing into acceptance testing. This review signifies that the acceptance test procedures can be executed in the VATC. This is an internal review, not one with the customer.
Consent to Ship Review (CSR)	Review to determine the readiness of a release for transition to sites for acceptance testing.
Site Readiness Assessment (SRA)	At the completion of system test for each release, a technical exchange meeting is conducted with the DAACs and ESDIS to assess the readiness of the release for operations. This happens for all future releases, where the RRR only happens at the conclusion of the last release.
Release Readiness Review (RRR)	Conducted at the ECS SDPS system level for a GSFC project review team upon completion of release acceptance testing. The RRR is held to determine if the release is ready for transition to IV&V and Operations. This review is only held at the conclusion of the last release (currently 6B).

Table 3-2 contains a list of the reviews that have already been successfully accomplished on the program. This table is provided to specify the portion of the software development life cycle accomplished to this point in the project. Since these reviews have been completed successfully, they will not be repeated for each release. The architectural structure of the system was put into place at the System Design Review and has not changed over the many releases and patches of the system. The changes that occur with each release are minor and do not disrupt the overall architecture already reviewed. As new components are added or deleted, these are discussed in the IRR for the subject release. The preliminary design and detailed design of each of the releases is reviewed through peer reviews. The overall preliminary and detailed designs have already been reviewed, and thus, what occurs from this point on are additions or deletions to that already approved design. The design documentation available as a draft at the IRR and "as-built" at the conclusion of each release is updated to reflect the changes that occur during each release.

Table 3-2. Reviews Already Successfully Accomplished

Review	Review Definitions
System Requirements Review (SRR)	The SRR encompassed a complete review of the ECS specification and the EOS/EOSDIS Requirements that drive the specification, it promoted a common understanding between the Project and the Contractor of the capabilities that ECS must provide.
System Design Review (SDR)	At the conclusion of the system design phase, a formal SDR was conducted to address the system architecture and the definition of the system level interfaces. This review signifies the completion of system design.
Preliminary Design Review (PDR)	At the conclusion of the preliminary design phase, a formal PDR was conducted to address the lower level preliminary design. It was conducted with the customer and user community.
Critical Design Review (CDR)	At the conclusion of the detailed design phase, a formal CDR was conducted to address the very detailed design issues. There were two of these conducted. One for Release A detailed design and one for Release B detailed design.
Interim Release Readiness Review (IRRR)	At the conclusion of the system test phase of the "at-launch" Landsat and AM-1 systems, an IRRR was conducted to determine that the release was ready for transition to Operations.

3.5 Training

This section addresses the requirements of NASA-STD-2100-91, NASA-DID-M200, Development Activities Plan, Section 7.0, Training for Development Personnel Planning. The requirements of Section 7.0 include:

- defining the personnel requiring training,
- identifying the types of training by categories of personnel, and
- identifying the plan for the conduct of training.

An RSC training committee exists to provide training to projects. ECS SDPS maintains representatives on this committee and submits its training needs based on the types of training beneficial to ECS SDPS and the needs of the individuals on the project. For example, software developers can sometimes benefit from training on use of software tools, C++ and/or Java, object-oriented design, etc. Training needs are assessed at the department level based on the proficiency of individuals in the department in their jobs, anticipated company business directions, and individual career goals.These assessments are provided to the training committee by each department's representative. The training committee then either brings in training or schedules the individuals for vendor training depending on the circumstances. The ECS SDPS representatives to the training committee are responsible for providing accurate and timely information about their training needs. They are also responsible for gathering feedback from the ECS SDPS departments that they represent.

This process was put into place for the following reasons:

- to ensure that training activities are planned,
- to provide training for developing skills and knowledge needed to perform management and technical roles,
- to ensure that individuals in the organization receive the training necessary to perform their roles.

Both process and tool oriented training requirements are identified and coordinated by the RSC training with the help of the ECS SDPS representatives. As additional training curriculum is identified, the training curriculum is updated.

A training administrator is responsible for the training program. The roles and responsibilities of the training administrator include coordination with instructors, scheduling resources for each course, notifying the representatives of the training committee of course details, ensuring that materials are available and distributed on time, collecting feedback from each course (course evaluations), and maintaining records of course conduct.

3.6 Software Tools and Environment

The ECS Development Facility (EDF) is a collection of hardware components and software tools that act in combination to support software development processes. The EDF is used to support software development and maintenance for all releases, thereby ensuring a consistent set of tools throughout the project. The environment supports all software processes from initial specification through integration and system turnover to test. In addition, the environment supports configuration management of all of the products of the software process.

This section focuses mainly on the hardware components and software tools that exist in the Development Infrastructure Environment of the EDF, which is separate from the Office Automation Environment of the EDF, as depicted in Figure 3-1.

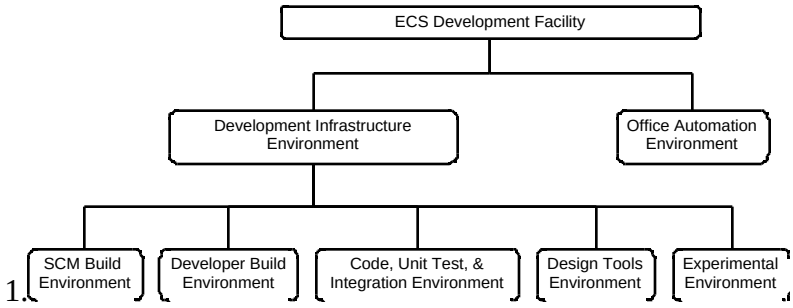

Figure 3-1. ECS Development Facility Environments

The Development Infrastructure Environment is logically separated into the multiple environments described below, which are used to group and define the primary uses for specific computers. This does not preclude the possibility of a machine being used to perform concurrent functions that are logically associated with different environments.

Table 3-3. EDF Environment Descriptions

Environment Name	Environment Description	Typical Users
Experimental Environment	Computers used for evaluations, prototyping, demonstrations, and COTS upgrade experiments.	Developers (authorized to do prototypes)
Developer Build Environment	Computers building the software during code and unit test. These platforms are specifically set up for compilations only. They are devoid of software that will allow the running of the ECS SDPS components. These were established to baseline the build environment and ensure that the developers are building the software with the same platforms as SCM.	Developers
SCM Build Environment	Computers for building the software on the ECS SDPS code baseline. The SCM group schedules the nightly builds of the ECS SDPS software on these hosts. They are configured the same as the Developer Build platforms to ensure consistency. They are isolated to ensure that the nightly builds will complete in a timely manner.	SCM
Code, Unit Test, and Integration Environment	Computers used for making code changes, unit testing these changes and then integrating the changes.	Developers
Design Tools Environment	Computers used for using CASE tools during the design phases of a release. These computers also house tools such as Discover, which allows the reverse engineering of existing code into models.	Developers
Office Automation Environment	Computers on the desks of personnel for the use of creating documentation, presentations, and other office automation tasks.	All ECS personnel

These environments all contribute to the “software engineering environment”, which is the environment that allows for the development of all the artifacts of the software process. Where it is important, references to the specific environments within the EDF will be made clear. Otherwise, for readability the term "EDF" is used and should be interpreted contextually.

3.6.1 Robust Development Environment

Table 3-4, EDF Development Environment Characteristics, defines the software engineering development support functions that are required for a robust development environment, the kind

of environment typical of a software project as large and complex as ECS SDPS. The ECS SDPS environment is an open environment, tailored to support development in a number of different programming languages using different design methods. This describes the general categories of tools. The last column, ECS Tool, provides specific examples of the tools used on ECS SDPS.

Table 3-4. EDF Development Environment Characteristics (1 of 2)

Tool Category	Category Description	ECS Tool
Analysis & Design Support		
Requirements analysis	Tools must be able to store requirements statements and relate them to other things (i.e. higher level requirements, design, etc.)	MS Access
Design	Tools must be able to support the methodologies specified in the software development process identified in this document, such as object-oriented design (UML) and entity-relationship diagrams for database design.	Rational Rose
Graphical User Interface (GUI) builders	Tools must be able to allow the rapid development of graphical user interfaces. These could be for Motif applications as well as Java applications.	Builder Xcessory
Automated code generation	The designs must be able to generate code whenever possible such as generating the header files from the object-oriented design tool and generating user interface code from the GUI builders.	Rational Rose, Builder Xcessory
Code Development Support		
Compilers	Tools to compile code on multiple platforms. Multiple languages are supported, so multiple compilers are required.	compilers from the hardware vendors Sun, SGI, HP
Linkers/loaders	Tools to link compiled code into executables.	same as above
Debuggers	Tools to debug executables during the code and unit test phase of development as well as during integration.	same as above
Code Coverage Tool	Tools to analyze the paths through the code during unit test runs. This provides information about the “coverage” of a unit test.	Pure Coverage
Memory Leak Detectors	Tools to analyze the allocation and deallocation of memory in C and C++ programs. These tools would report memory leaks as well as report memory usage violations.	Purify
Integration & Test Support		
Simulation/emulation tools	Tools to simulate user interfaces and their interaction with the system. This is useful for regression testing GUI applications.	XRunner

Table 3-4. EDF Development Environment Characteristics (2 of 2)

Tool Category	Category Description	ECS Tool
Test performance reliability	Tools to capture the performance and reliability metrics of programs. These tools would capture statistics such as the execution times of test runs and the percentage of successful execution of the tests.	LoadRunner
Other		
Defect tracking	Tools must be able to support the tracking of non-conformance reports (NCRs), which are liens against a software product. These tools should allow for the tracking and categorization of defects / errors in the product.	DDTS
Documentation	Tools must be able to support the generation of documentation, which includes multi-media information such as screen dumps and other figures and diagrams.	MS Office, FrameMaker
Lines of code counters	Tools must be able to read the source lines of code and calculate totals at a summary level. This is used to track the progress of development.	kdsi javancss Raytheon code counter

While Sun and SGI are the primary development platforms, there are also sufficient numbers of other Unix platforms (HP, DEC, and IBM) for developers to access if necessary. These are available to support porting efforts for those portions of the system that must be available on multiple software platforms (for example, Toolkit and EOSView). The development environment on each of the Unix platform is the standard development environment of the hardware vendor.

Software developers are typically provided with a computer with X-window capabilities. This is most often an X-terminal, which runs off an X-terminal server (a Sun server). Other desktop development options are a Sun workstation or a PC with X-terminal emulation capabilities. Regardless of the platform on the developer's desk, office automation environments are provided in some form in order for developers to provide necessary documentation. A separate PC typically provides the office automation if the developer is not using a PC for their development platform. Other times the office automation environment is provided by a PC emulator on the Sun workstation or X-terminal server.

3.6.2 Development Infrastructure Environment Evolution

The ECS SDPS Development Infrastructure Environment cannot be static if it is to remain useful. It must evolve as new versions of COTS software become available and necessary due to support issues with the vendor. Therefore, a mechanism must be in place in order to upgrade or maintain the development environment.

Requests for hardware or software changes in the EDF must be documented and controlled. Although configuration change requests (CCRs) may originate from any user in the EDF, the EDF Change Control Board (EDF CCB) must approve all CCRs. All EDF CCRs are dispositioned by the EDF CCB and tracked to closure. The EDF CCB and COTS upgrade processes are documented in project instructions.

3.7 Software Metrics

The process of software development can be effectively managed (monitored and improved upon) only if there is an objective means of measuring the quality of the development efforts. The ECS SDPS Project metrics program, organizational interfaces, responsibilities, and reporting vehicles are derived from the RSC Software Operating Instructions (SOIs) through the tailoring process. The SOIs provide a standard description of metrics and the methods for calculating them to ensure consistent implementation across projects and organizations.

3.7.1 Responsibilities

Each ECS Department is responsible for establishing and reporting its metrics for the ECS SDPS Project. The ECS Development Director is responsible for establishing software metrics for the ECS Project, and ensuring that they are implemented within the Development Department or other departments as needed (e.g., M&O collects metrics that indicate the status of software non-conformance reports for software in operation at the DAACS).

3.7.2 Examples of Software Metrics

Table 3-5 shows the list of metrics that are collected and analyzed at this time. The list may changed in the future as described above, but this table provides an example of some of the metrics that have been found to be useful management tools.

Table 3-5. Examples of Software Metrics (1 of 2)

Metric	Description
Software Size	Shows the total software size typically by subsystem. The rate at which the total software size increases or decreases in conjunction with hours expended provides useful information about productivity. It is also used to estimate the number of defects that will be incurred later in the life-cycle.
Non-conformance Reports	Shows the number of severity 1, 2, and 3 NCRs and what state they are in. The number of defects provides an estimate of the quality of the code and the process as well as an estimate of the remaining rework to be done. This metric currently is calculated for NCRs written prior to shipment of the product. The NCRs from the field will also be measured.
Staffing	Shows the full-time equivalent engineers of the software development staff. This provides the resource base from which the work can be accomplished.

Table 3-5. Examples of Software Metrics (2 of 2)

Metric	Description
Cost Performance Index	Shows the budget of work performed (BCWP) / actual cost of work performed (ACWP). This provides an indicator of the efficiency of the progress being made towards the estimated costs.
Schedule Performance Index	Shows the budget of work performed (BCWP) / budget of work scheduled. This provides an indicator of the efficiency of the progress being made towards the scheduled work.

3.8 Security

The ECS SDPS project will employ security practices and procedures to ensure the security, integrity, and continued operation of the EOSDIS Core System and the information it stores and processes. These practices and procedures are defined in the Security Plan for the ECS Project, CDRL 214-CD-001-001, and implemented by the ECS SDPS Systems Engineering Department designated Project Security Manager. There also exists a Technical Security Interoffice Working Group, which is used to decide on security issues and disseminate decisions. The security practices and procedures are summarized in the following paragraphs. For details, refer to the Security Plan.

The ECS Security Plan addresses sensitive, unclassified information and national resource protection requirements associated with ECS SDPS design, development, implementation, operation, and maintenance. The objectives of the ECS security plan and program are:

a. to establish a baseline for updating, improving, developing, maintaining, and managing Automated Information Systems (AIS) security requirements for ECS SDPS,

b. to ensure that the ECS SDPS design and implementation incorporate federal and NASA AIS security policies and guidelines,

c. to promulgate an ECS SDPS AIS security policy and to guide AIS security procedures for ECS SDPS,

d. to document the current AIS security environment, establish program objectives, outline a plan of action with milestones for implementing the ECS SDPS AIS security program, and implement the plan.

AIS security management requires specific activities throughout the ECS SDPS lifecycle. These activities include administrative, physical, personnel, and technical (some software related) measures. The preliminary design ensures re-evaluation of system requirements from a security perspective prior to each release, incorporation of security requirements in the ECS SDPS system and segment design, and planning for security testing and evaluation and for configuration management.

This page intentionally left blank.

4. Software Development Process

4.1 Software Development Process Overview

The ECS SDPS project is delivered in several releases or drops. An evolutionary development process minimizes integration and test risk, and facilitates availability of system functionality when it is required. The contents of each release are controlled by the Science System Release Plan for that release. Any changes in functionality of the release require a CCR to document the change. Figure 4.1-1 shows the entire development process. This document, however, will only address the "software" tasks of this process. This entire cycle is executed for each release.

Figure 4.1-1 was derived from the IPDS through a tailoring process. The IPDS provided the set of tasks and these were tailored for the ECS SDPS project. Each task is decomposed into subtasks. Project instructions are mapped back to the tasks that they support. The tasks describe what needs to be done and the project instructions describe how the tasks are to be performed. The tailored IPDS process is available on the EDHS Internal Server along with the project instructions.

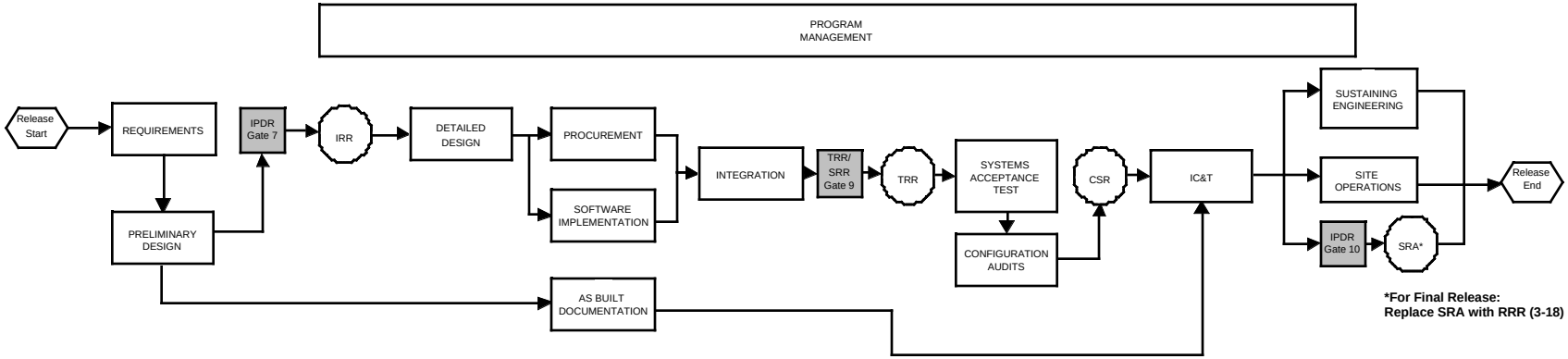

Figure 4.1-1. Software Development Process

4.1.1 System Requirements Definition

The development process for any release within the ECS SDPS project begins with a Requirements Definition phase. This phase is performed at the outset of each release to define the environment and functionality for the entire system. The box in Figure 4.1-1 entitled "1-2 REQUIREMENTS" actually has two parts, a System Requirements Definition and a Software Requirements Analysis. The software development organization supports this entire phase; however, the SED has lead responsibility for many of the activities that occur during this time. The software requirements analysis is addressed in Section 4.1.2.1 Software Requirements Analysis.

The Functional and Performance Requirements Specification (F&PRS) and the Interface Requirements Documents (IRDs) provides the system level requirements for ECS (Level 3 (L3) and interface requirements). SED is responsible for ensuring the proper allocation of these L3s to releases as well as to system configuration items; jointly with the Development Department. The requirements may be allocated to computer software configuration items (CSCIs) or hardware configuration items (HWCI). SED also provides an operations concept or architectural scenario for how these L3s interact with the CSCIs. This provides the basis for breaking the L3s down to a lower level of detail.

In this phase, the system requirements and system tools are analyzed. The architecture model (as defined in DID313 and DID305) is revised to include new interfaces as required due to the L3 requirements allocation. Typically most of the CSCIs are already identified in prior releases, and later releases add capabilities to existing CSCIs which are appropriate. This does not preclude the definition of a new CSCI in a later release if the corresponding functionality does not correspond well with existing CSCIs.

Additionally, the following tasks are performed.

- The system capabilities are identified for each release and documented in the Science System Release Plan. A system capability is a high-level description of a group of one or more requirements. The system capability name is used for tracking purposes during scheduling, integration, and delivery to test. Requirements (L3) are mapped to the system capabilities. During the Software Requirements Analysis phase, the L4 requirements are also mapped to the system capabilities.

- New CSCI interfaces and system level scenarios are defined. The interfaces at this point are named and a protocol supplied. The details of the interfaces, for example the data structures passed, are not defined until the design phase.

- Software sizing in terms of Source Lines of Code (SLOCs) is re-estimated by the Development Department. Estimates for system capabilities (groups of requirements) already exist, but the estimates may need to be revised due to the operations concept produced during this phase.

At the conclusion of the system requirements definition phase, SED produces the Science System Release Plan. This plan provides the customer and the other ECS SDPS organizations with a system description for a release. The other organizations such as Development, Science Data Engineering, and M&O use this document to finalize their detailed plans for the release.

4.1.2 Basic Software Development Phases

Figure 4.1-2 shows the software development activities of a typical capability and the artifacts produced from each stage. The activities are at the top of the bars and the artifacts below the bars. Each phase contains a peer review (the milestone in the figure) of the outputs and a workoff period. The workoff period is where the defects from the peer review are resolved, resulting in improved artifacts. Each system capability is planned in the master schedule in this manner.

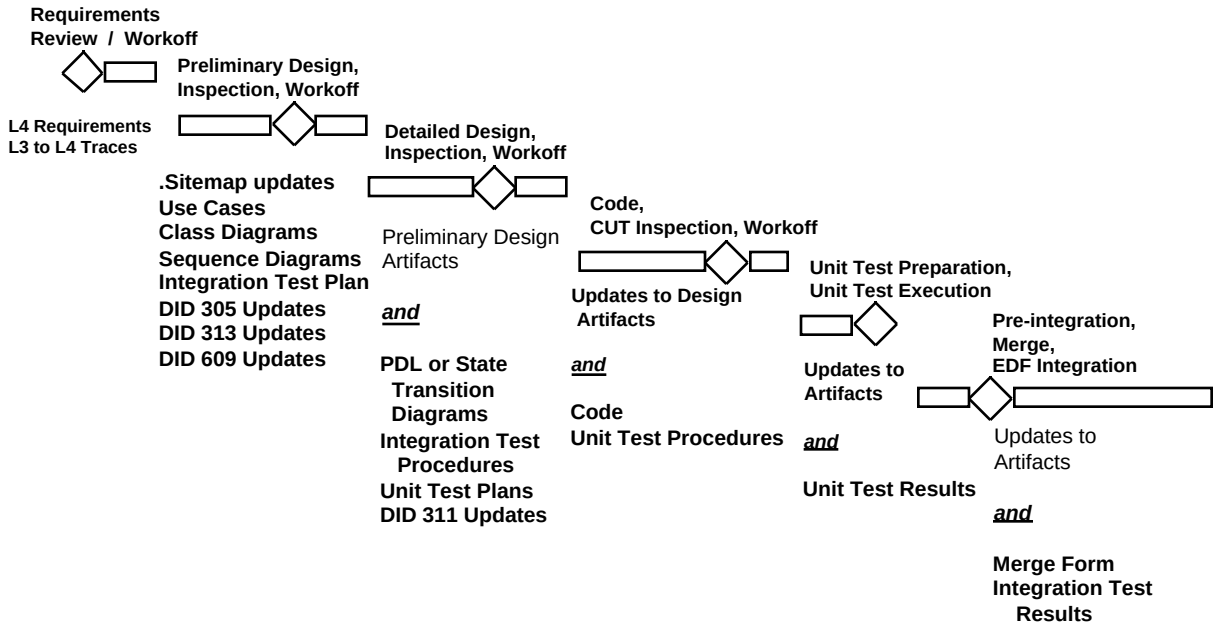

Figure 4.1-2. Development Tasks and Artifacts

4.1.2.1 Software Requirements Analysis

In this phase a set of Level 4 requirements are generated based on the Level 3 requirements allocated to each of the CSCIs by the SED during System Requirements Definition (section 4.1.1). Potential candidates for software reuse (heritage software, ECS SDPS common software, and commercial-off-the-shelf (COTS) software) may be explored during the analysis.

The L4 requirements are traced to the L3 requirements and system capabilities during software requirements analysis. The conclusion of the software requirements analysis is marked by a peer review in which the L4s and their mapping to L3s and system capabilities are reviewed. SED is a participant in this review to ensure that the intention of the L3 allocation and operations concept

development has been carried forward in the L4 requirements. After all the defects from the peer review have been corrected, the L4 requirements with their mappings to L3s are provided to SED for incorporation into a CCR to be presented to the Science Development (SD) CCB. Upon approval of the CCR, the Requirements Verification Traceability Matrix (RVTM) is updated to reflect the requirement changes.

4.1.2.2 Preliminary Design

During the Preliminary Design phase, a high-level design is generated for each system capability. The preliminary design includes an almost identical set of design artifacts as detailed design, but the artifacts describe the design at a higher level.

The Rational Rose analysis and design tool is used to document the object-oriented artifacts of the design (object diagrams, sequence diagrams, use case diagrams, etc.). The Unified Modeling Language (UML) is the methodology used on ECS.

High-level descriptions of the artifacts generated during preliminary design are provided below. These examples are provided only to illustrate the types of artifacts produced during this phase. The most up-to-date list of preliminary design artifacts is documented in the PI for peer reviewing design. It is important to note that these artifacts apply to the new capabilities only. For example, use case diagrams do not exist for all the previous releases of ECS since UML was not the original methodology used on the program. Use case diagrams will not be produced for all the previous capabilities, only the new capabilities for a release.

- L4 requirements and their mapping to components of the design.
- HW/SW mapping showing what H/W components the S/W executes on.
- Object-oriented design artifacts such as: use case diagrams, class diagrams, sequence diagrams.
- Design deliverable documents: the Segment/Design Specification (305), Internal ICD (313), and Operators Tools Manual (609) - Release 5A and subsequent releases.
- Integration Test Plans
- SLOC and resource estimates. This is revised from the original estimates in order to review the feasibility of the implementation schedule.

Peer reviews are conducted to validate allocation of level 4 requirements to the design components and to validate the overall high-level design itself. Potential candidates for software reuse are explored during this phase. The successful completion of the preliminary design peer review and the correction of all defects signify the completion of the preliminary design for that capability. The detailed design phase for that capability then begins.

4.1.2.3 Detailed Design

During the Detailed Design phase, a detailed "code-to" design is performed based on the preliminary design approved during the preliminary design peer review. In addition to the list of preliminary design artifacts, the following artifacts are generated. The most up-to-date list of detailed design artifacts is located in the PI for the design peer reviews.

- Program Design Language (PDL) to describe the complex algorithms of methods. Guidelines for which methods require PDL are included in the PIs.
- State Transition diagrams can be developed as an alternative to the PDL.
- Fully populated object-oriented design artifacts. At preliminary design the object-oriented artifacts are at a high-level. At this stage, they are fully defined. For example, the class diagrams will include all classes with all attributes and their data types and all methods with full signatures.
- Updates to the Release x Database Design and Database Schema Specifications (311)
- Integration Procedures are produced providing detailed steps on how to integrate the final software capability. These Integration Procedures are incorporated in the Acceptance Test Procedures by the System Verification group within SED.

Peer reviews are conducted to validate the detailed design of the capability. After completion of all the defects identified during the peer review, the detailed design phase is complete and implementation can begin.

4.1.2.4 Implementation

This phase can be broken down into the following activities.

1. Software Coding
   - During software coding, classes are coded and a clean compilation produced.
   - Coding standards and naming conventions PIs are followed during this process. A different coding standard is provided for each language used on ECS SDPS.
   - A set of step-by-step unit test procedures is developed to verify that the requirements are satisfied.
2. Peer Review
   - Peer reviews are conducted to ensure that the code implements the allocated requirements and complies with project standards.
   - In addition to the source code and design artifacts, the unit test cases and procedures are reviewed. For the official code and unit test artifact list, refer to the code and unit test peer review PI.

3. Unit Testing

   - After the code is peer reviewed and the defects from the peer review resolved, the component must be tested to ensure that its allocated requirements (i.e., the requirements allocated up through the next release) are satisfied.

   - The software developer conducts a unit test walkthrough with an appropriate subsystem or technical lead to verify the functionality of the unit. Quality Assurance is also invited to the unit test walkthrough, although they are not required to attend. The results of the unit test will also be placed into the respective SDF.

   - Any discrepancies in the code are documented. Only code with non-critical non-conformance reports (NCRs) in the newly introduced functionality are allowed to pass the unit test.

4. Merging to the baseline

   - The merge process is documented in a project instruction, but is summarized here at a high-level.

   - A merge request is submitted to the Software Turnover Tracking System (STTS) upon completion of the unit test.

   - The merge request is discussed at a meeting with all the subsystems represented

   - The Construction Office leads the merge meeting. If the merge request is complete and the integration lab is ready to integrate the functionality, the merge is approved by the Construction Office.

   - Upon approval, the software developer responsible for the code, uses the configuration management tool ClearCase to “merge” the code to the appropriate software release baseline.

The “merge” of the code to the appropriate release baseline signifies the end of the implementation phase and the beginning of the integration phase. The software is built by SCM and staged to an area within ClearCase where the integration lab can receive it.

4.1.3 Software Integration and Test

In the EDF, the Development Department is responsible for integrating the software into a working software system, through the execution of integration procedures. The Construction Office oversees the integration. Problems with the software are resolved through an iterative approach of writing NCRs, fixing them, merging them to the baseline, and updating verifying the NCR fixes. The NCRs that are written are considered “informal” NCRs since they are found internally. The software development project instruction for NCRs defines the process for these NCRs from the time of submittal through resolution including the definition of the severity of the NCRs.

When all the severity 1 and 2 NCRs have been resolved for a particular integration procedure, a formal run is executed. The Construction Office oversees the procedure execution. Quality Assurance and the System Verification group are invited to attend. The results are documented on a test execution form. Only non-critical NCRs can exist in order for the integration procedure to be considered a successful execution. These non-critical NCRs are recorded in the NCR database and on the test execution form. The test execution form is saved in a test execution folder to be turned over to System Verification at the Software Turnover Meeting.

When all of the required integration procedures for a release have been successfully executed, a Software Turnover Meeting (STM) is conducted. The STM will ensure that the software tested meets the corresponding Level 4 requirements, and the integration test documentation is complete. The Construction Office provides direction to SCM to generate a code baseline and tar file(s) of executables prior to the meeting in preparation for the turnover to test. The Systems Verification group determines the readiness of the product for turnover based on the items supplied by the Construction Office. The SED chairman of the Science Development CCB has signature authority to accept the delivery of the software. The items provided for the STM are documented in PIs.

4.1.4 System Installation in the VATC

Following the STM, the System Verification group installs the system in the VATC for system verification and acceptance testing. SCM provides the tar files generated prior to the STM to the System Verification group for this activity. A critical function of the installation is to validate the system will work outside of the EDF and that the installation instructions are complete. Formal checkout procedures are executed, in full or in part, to characterize the success of the installation. NCRs associated with configuration problems as a result of the installation will be forwarded to the EDF. This allows time to fix the installation issues before delivering the release to the Distributed Active Archive Centers (DAACs). Fixes are provided to the System Verification group as patches and are regression tested in the VATC, prior to the CSR.

4.1.5 Acceptance Test

At the completion of the installation and checkout of the release, a Test Readiness Review (TRR) is held. The purpose of the TRR is to assess the readiness for the start of acceptance tests. The problems found in installation and checkout are assessed as well as the readiness of the acceptance test procedures. The pre-installation serves as a pathfinder for the installation of the formal delivery occurring after CSR.

Acceptance testing consists of executing operational scenarios on the system. Acceptance test procedures are developed during design and implementation, debugged during integration, and approved by SED and the customer. The System Verification group, to the maximum extent possible, establishes representative site configurations within the VATC to verify the site-unique testing to be performed in the field.

After TRR, the System Verification group begins to execute the acceptance test procedures in the VATC. Each procedure is dry run first. After successful dry runs of the procedure, a formal run is scheduled with IV&V and the customer. Quality Assurance also witnesses formal runs of the

tests on a sampling basis. Upon completion of the full set of acceptance tests, a Consent to Ship (CSR) review is held and the software is shipped and installed in the field (the DAACs).

Problems found during acceptance testing are documented in NCRs. The Development department resolves the NCRs and supplies patches to the release as requested by the System Verification group. Refer to section 4.1.8 “Non-conformance Reports” for more discussion on how NCRs are resolved.

The System Verification group then has the responsibility for conducting a subset of the acceptance testing on the ECS SDPS system at each site. Complete details concerning the acceptance test phase are provided in the Overall System Acceptance Test Plan (409-CD; Release 5A and subsequent releases).

At the end of the acceptance test phase, a Site Readiness Assessment (SRA) is conducted. The results of site release testing are presented at the SRA, and the review board determines whether the release is ready to be incorporated into the operational system. Acceptance is based on the results of the system acceptance tests, documentation of those tests, open NCRs, other system documentation (e.g., operations, maintenance, training, and logistics documentation), FCA and PCA audits, and the operability and maintainability of the new release (based on recommendations of the DAAC representatives). At the last release of the system, a Release Readiness Review (RRR) is conducted in place of the SRA. The content of the RRR is the same as the SRA, but the RRR marks the completion of the system development.

4.1.6 Performance Verification

The Performance Verification Center (PVC) was established to test the system under a load equivalent to the load that will be present during operations. Special test procedures are defined to test the performance and stability of the system under operations loads. Parallel to the system being tested against the functional requirements in the VATC, the system will be tested against performance criteria in the PVC. NCRs are generated for performance issues just as they are written for functionality issues. Patches are applied and the performance regression tested until the release performs satisfactorily in order to deploy. This must be accomplished prior to the RRR of a release.

4.1.7 Documentation

During the preliminary and detailed design phase, the software deliverables referenced in Table 4.1-1 will be generated. These deliverables will consist of redlines or change pages of the existing documents. After implementation and integration, these redlines are incorporated into a full document set of “as-built” documentation.

Table 4.1-1. Software Design Documentation

Document #	Document Name	Description
305/DV2	Release x Segment/Design Specification	Provides details on the context and design of CSCIs, at the Unix process level. In addition, information about libraries and classes are provided for pointers into the code.
313/DV2	Release x Internal ICDs	Provides details of interfaces between CSCIs including protocol information. Scenarios are used to illustrate interfaces. Tables provide high-level information about the interfaces such as whether they are remote procedure call interfaces or low-level socket calls.
609/DV2	Release x Operations Tools Manuals	Provides details of operator tools (the graphical user interfaces). Each operator tool is explained without regard to the procedures being operated. Other documents discuss the procedures used to perform functions of an operator.
311/DV2	Release x Database Design Specification	Provides information about the database tables, columns, relationships, indexes, etc. Includes everything associated with the physical implementation of databases in the system.

4.1.8 Peer Reviews

Peer reviews are internally conducted reviews focused on identifying defects in software development artifacts. On the ECS SDPS project, there are three methods possible for peer reviews: inspection, routing, and walkthrough.

- Inspection: A peer review where the material is distributed in advance of a meeting, which is held to discuss the defects found by the participants.
- Routing: A peer review where the material is distributed and individuals provide written comments back to the author and review lead. There is no meeting to discuss the issues.
- Walkthrough: A peer review where the material is presented and discussed in a meeting. The material is not distributed in advance.

The project and work instructions for peer reviews discuss these methods in detail and define criteria of when to use each. The criteria are used for both new capabilities and NCR fixes. When a new capability or system requirement is defined, the developer or technical lead, must estimate the amount of change required. The table in the project instruction is used to determine whether the software life-cycle should begin at preliminary design, detailed design, or implementation. For example, a new capability requiring of 5 lines of new code might not require a peer review

of any kind, where a new capability of 200 SLOC might require the developer to present a detailed design and then move on to implementation. The same methodology is used for NCR fixes.

The participants in the review are internal to the project and are determined based on the type of work product being reviewed. The work instructions for the specific work product type designate the different departments that must be invited to the peer review. In addition, peers within the developing organization are selected. The review team should be as small as possible in order to maximize the amount of defects found and minimize the cost of the review.

Peer reviews are scheduled in the master schedule. The rework required for fixing defects are also scheduled. This ensures that a phase does not complete without a peer review and that the work-off period also must be designated as complete with a distinct activity. This also allows visibility by other departments such as Quality Assurance into the peer review schedule.

Quality Assurance participates in peer reviews in an auditing role. They receive notification and distribution of each of the packages. They schedule the peer reviews that will be audited and produce an audit report for each review audited.

4.1.9 Non-conformance Reports and Patches

There are two types of NCRs on the ECS SDPS project. The “informal” NCRs are those NCRs found for a release prior to the CSR of that release. The “operational” NCRs are those NCRs found against a release in the field (installed at a DAAC). The process for each of these types is not the same, although they are similar. There are project instructions that define the NCR states (such as new, assigned, resolved, verified or closed), NCR severity (1-5), and responsibilities by organization for moving the NCRs.

During the acceptance test phase, the System Verification group of SED may identify problems (documented as “informal” NCRs) that must be fixed before the release can be shipped to the DAACs. The Development department must fix the required NCRs using the standard development process outlined in sections 4.1.2 through 4.1.4. The amount of change required determines where in the life-cycle the NCR fix must begin. Refer to section 4.1.7 “Peer Reviews” for more information on how this is determined. When the NCR fix or set of NCR fixes is turned over to the System Verification group, a patch is installed in the VATC and the acceptance tests that previously failed are repeated.

At the CSR for a release, any outstanding “informal” NCRs are transferred to the “operations” NCR list since the release is being shipped to the site with known defects. After the CSR of a release, problems may be identified at the DAACs. Maintenance and Operations project instructions govern the definition and prioritization of these problems. Initially the problems are documented as trouble tickets and when the trouble tickets are confirmed as a change to a baseline, they become “operational” NCRs. The Development department resolves the NCRs using the standard development process outlined in sections 4.1.2 through 4.1.4. The amount of change required determines where in the life-cycle the NCR fix must begin. Refer to section 4.1.7 “Peer Reviews” for more information on how this is determined.

The Deployment IPT (Integrated Product Team) schedules the release of patches and maintenance releases. Patches are turned over to the System Verification group for testing in the VATC and then a Pre-Ship Review (PSR) is held. The patch or maintenance release contents are based on the priorities of the "operations" NCRs. Maintenance and Operations project instructions provide details on how the Deployment IPT works and the prioritization of trouble tickets and "operations" NCRs.

The important aspect of this discussion is that the software development process is the same regardless of whether the software being developed is for an "informal" NCR, an "operations" NCR, or new functionality.

4.1.10 Process Variances

In order to diverge from the software development process, a process variance request must be approved. A project instruction describes the process for this and the signature approval authorities. This provides a mechanism to document and heighten awareness within the project when a process does not make sense for a particular instance. Process variances also provide a means to determine process improvements. If process variances are frequently being approved for the same part of the process, then it may be time to change the process.

For example, an NCR fix that is stopping all storage of a particular kind of data at one DAAC has a 125 SLOC change. The peer review process requires a three day inspection notification to peer review the code change. A process variance request could be approved to reduce the three day inspection notice to one day. An alternative process variance request could request to use the walkthrough method instead of the inspection method, which only requires a one day notice.

4.2 Prototyping

Prototyping at this stage in the ECS SDPS Project consists of direction as approved by ESDIS in Engineering Support Directives (ESDs). These ESDs have high-level schedules and information on process embedded in them. Therefore there is no special reporting or management oversight required for prototypes in general. Each ESD is managed according to the direction documented in it.

4.3 Software Reuse

For the ECS SDPS project, software reuse consists of:

1) the use of COTS software,

2) the use of heritage software (software obtained from a non-commercial source external to the ECS SDPS project, e.g. the Delphi class library or public domain class libraries), and

4.3.1 COTS Software

A significant portion of the ECS SDPS system consists of commercial off-the-shelf (COTS) software. Integration of COTS software with ECS SDPS-developed applications is a major task of the program. An ECS SDPS PI exists to define the process by which COTS software products are identified, selected, and incorporated in the ECS SDPS system.

A COTS baseline, which defines all the COTS products used on the ECS SDPS project, is maintained by SED. The COTS baseline must distinguish deliverable COTS software (COTS software that is to be delivered as part of ECS SDPS) and development support COTS software (COTS software that will be used at the EDF to support ECS SDPS development). Note that in some cases a COTS software product may be both deliverable COTS and development support COTS.

A significant amount of the effort with COTS is focused on maintenance and upgrades. COTS vendors update their products and the ECS SDPS project must determine when it is appropriate to integrate a new COTS version into the existing ECS baseline. A project instruction defines the COTS upgrade process.

4.3.2 Heritage Software

During the development of the ECS SDPS Project, heritage software (i.e., software developed on previous projects or publicly available) will be used whenever feasible to reduce life-cycle costs. However, potential cost savings must be balanced against overall ECS SDPS goals (e.g., evolvability) and the rigor of the development process.

According to the ECS Performance Assurance Requirements (PAR), heritage software does not have to be developed to ECS SDPS development guidelines. However, the application of PAR software assurance requirements to any modifications to heritage software is required. In addition, the PAR, Section 1.4.b, focuses on "establishment of suitability for use on ECS SDPS" through:

1. requirements comparison
2. review of all Verification & Validation records
3. identification of all waivers and deviations
4. review of mission experience, problems, or anomalies
5. additional testing planned

An ECS PI describes the development life-cycle for heritage software. In general, heritage software is not made to follow the ECS coding and design standards, since this would add extra effort and reduce the benefits of using the heritage software.

This page intentionally left blank.

5. Software Quality

5.1 Software Quality Assurance Overview

The Quality Assurance organization ensures that software products developed, modified or procured during the ECS SDPS contract (except prototype software used only to help in requirements definition) comply with contractual requirements and standards, thereby promoting the highest quality standards. Quality will be the joint responsibility of all ECS SDPS employees during the ECS SDPS development and maintenance effort.

5.2 Quality Assurance Organization

The ECS SDPS project approach to ensuring that all software meets the performance assurance requirements for ECS SDPS is managed in two ways. First, the process and procedures for software development are documented herein and in the Project Instructions (PIs) referenced throughout this document. Second, the Quality Assurance team provides an independent monitoring, auditing, and corrective action function, which ensures that the approved software development process and procedures have been followed or a variance request has been approved. The ECS SDPS project will employ a quality program throughout the system development life cycle.

The Software Quality Assurance (SQA) program is described in the ECS QA Plan, which has been mapped to the Performance Assurance Implementation Plan (PAIP) DID 501/PA1. Specific work instructions and quality assurance procedures are maintained in the Information Technology (IT) homepage. The key SQA activities are summarized in this document.

The Quality Assurance organization will ensure the implementation of the approved software development processes and standards identified in this document. To accomplish this, the Quality Assurance Manager maintains a dual reporting role to the ITS Vice President and General Manager, and the IGS Product Assurance Manager, thereby assuring attention and authority through all levels of Raytheon. Additionally, the Quality Assurance organization will be the focal point for system quality coordination with the Goddard Space Flight Center (GSFC) Quality Assurance manager.

Members of the Quality Assurance team will be assigned to monitor, audit and ensure corrective action of specific functional areas of the ECS SDPS project while reporting to the ECS Quality Assurance Manager. The Quality Assurance personnel perform independent assessments of the engineering development and test phases of the product. In this way, there is continuity of coverage for verifying the process and product compliance of ECS SDPS project activities.

5.2.1 Roles and Responsibilities

All members of the ECS Project will take an active role in SQA activities. To ensure the appropriate completion of Quality Assurance activities the following roles and responsibilities are defined:

- Quality Assurance Manager
  - Ensure the ECS QA team is staffed appropriately
  - Manage the ECS Quality Assurance budget
  - Conduct performance evaluations of Quality Assurance staff
  - Provide office space and appropriate tools to conduct quality assurance activities to staff
  - Ensure Quality Assurance Engineers receive the appropriate training
  - Coordinate ISO Audits
- Quality Assurance Engineer Lead
  - Allocate resources to functional areas to prepare and conduct quality assurance evaluations
  - Prepare monthly ECS Quality Assurance status reports
  - Coordinate Quality Assurance activities with GSFC Performance Assurance Office, ECS Project Management
  - Conduct Quality Assurance presentations on request at ECS Project Management Reviews
  - Ensure correct Quality Assurance representation to the SEPG meeting in order to learn of pending and upcoming process changes.
- Quality Assurance Engineers
  - Ensure the appropriate standards, processes, and procedures are selected, implemented, and adhered to by performing evaluations against consistent criteria.
  - Initiate problem avoidance by disseminating evaluation results and supporting corrective action to resolve identified discrepancy issues.

- ECS Functional Area Manager
  - Approve and support the SQA activities
  - Receive and act on discrepancy reports and escalated items
  - Participate in SQA evaluations if required
  - Provide the Quality Assurance Engineer access to the same tools as project personnel
- ECS Project Team Members
  - Identify and report any inability to perform a procedure to the project manager, functional area manager, and Quality Assurance Engineer through the process variance request.
  - Provide information needed to conduct a quality evaluation
  - Identify and implement the solution for discrepancy issues

5.3 Software Quality Activities

The Quality Assurance Engineer will perform quality assurance evaluations to provide the ECS Software Development Management insight into the use and adherence to the processes documented in the software development plan. There are two types of evaluations performed, audits and product evaluations. An audit is an objective examination of documented processes to verify that company and or contractual requirements are being met. A product evaluation is an objective examination of deliverable, non-deliverable, or non-developmental products to verify they are in agreement with company and /or contractual requirement. In addition to conducting quality assurance audits and product reviews, the Quality Assurance Engineer is responsible for creating/updating audit and product evaluation criteria, preparing and maintaining the results of audits, product evaluations, and cited deficiencies. The Quality Assurance Engineer will also provide periodic quality assurance status reports.

5.3.1 QA Audits

The following list the software development processes, as well as system engineering processes in which software development is an active participant, that will be evaluated by the Quality Assurance Engineers:

- Preliminary Design
- Detailed Design
- Implementation
- System Installation in the VATC

- Acceptance Test
- Software Integration and Test
- Peer Review

5.3.2 QA Product Evaluations

The Quality Assurance Engineers perform product evaluations in accordance with ITS Quality Assurance Procedures. The following list the work products that will be evaluated by the Quality Assurance Engineer:

- 305/DV2 Release x Segment/Design Specification
- 313/DV2 Release x Internal ICDs
- 609/DV2 Release x Operations Tools Manuals
- 311/DV2 Release x Database Design Specification

5.3.3 Quality Assurance Audit Criteria

The Quality Assurance Engineer will create audit and product review criteria based on the tailored ECS /IPDS tailored task descriptors, as well as existing plans, corresponding project instructions, and work instructions. Quality Assurance Engineers will follow the procedures maintained on the Information Technology (IT) home page to develop consistent and objective criteria for quality evaluations.

5.3.4 Quality Assurance Deficiency Reporting

The Quality Assurance Engineer will prepare and maintain the results of audits, product evaluations, and cited deficiencies according to the Quality Assurance Deficiency Reporting Procedure, 19-0-14. This procedure defines the criteria for the Quality Assurance Engineer to report deficiencies as either Corrective and Preventive Action Reports (C/PAR) or Deficiency Reports. Additionally, this procedure defines the steps taken in reporting deficiencies to higher levels of management until resolution is achieved. This procedure also describes the use of a Quality Assurance Tracking Database that is used to record quality assurance evaluations. The Quality Assurance Engineer will also be responsible for filing hard copy of the evaluation and corresponding discrepancy reports.

5.3.5 Quality Assurance Status Reporting

The Quality Assurance Engineer will provide a monthly status report to the Development Director. The report will address activities that are completed or in process during the last reporting period, as well as contain a list of activities to be started or completed during the next reporting period. The report will also identify all open issues and all issues closed during the previous reporting period.

5.4 QA Resources and Schedule

ITS Quality Assurance Program provides office space and equipment to support the Quality Assurance Engineers. ITS Quality Assurance Program will provide software that is unique to supporting the Quality Assurance records tracking system. The appropriate SQA training is handled by the ITS Quality Assurance Training Program.

The following represents the standard tools used by the Quality Assurance Engineers:

- MS Project
- MS EXCEL
- MS WORD
- MS Power Point
- Other tools provided by the project

The Quality Assurance Engineer develops the SQA schedule in conjunction with the project schedule and updates as needed. The schedule includes all SQA activities (that is, plan development, criteria development, evaluations, SQA training, and SQA participation in other related project activities). The schedule is maintained in the master schedule with the rest of the project's activities.

This page intentionally left blank.

6. Software Configuration Management

6.1 Configuration Management

This section covers plans and processes for configuration management (CM) of ECS SDPS software. Under ESDIS direction, the ECS SDPS has the sole responsibility for all changes to ECS SDPS products. Software configuration management is the responsibility of the Configuration Management group, which resides in the ECS SDPS Project's System Engineering Department. The Release Configuration Management Plan for SDPS, CDRL 102-CD-003, contains a description of the CM process, responsibilities and tools.

To manage requirements during the ECS SDPS development, a requirements database exists. All ECS SDPS requirements are based on the ECS Functional and Performance Requirements Specifications (the Level 3 requirements). The ECS SDPS system requirements management process consists of those functions traditionally performed to thoroughly understand requirements at the inception of a development program, plus several functions designed to meet the unique requirements of ECS SDPS. These unique ECS SDPS functions include:

- Expanded requirements traceability.
- Allocation to configuration items (software/hardware).
- Allocation to releases / drops.
- Allocation to test procedures (both integration and acceptance).

Configuration management PIs provide lower-level details on each of the CM activities discussed in the following sections.

6.1.1 Configuration Identification

All of the configuration items controlled under CM are documented in ECS System Baseline Specification (905-TDA-001). This document contains a listing of the items from the configuration management database. The processes for updating the database and documentation are described in CDRL 102-CD-003, Release Configuration Management Plan for SDPS and CM PIs.

The ECS custom code components are not listed in the ECS System Baseline Specification. The ClearCase tool contains all the information about all the custom code files and versions. The Configuration Management Plan and CM PIs describe the use of ClearCase to control code. All other configuration items are listed in the ECS System Baseline Specification and the configuration management database.

6.1.2 Configuration Control

Configuration control is maintained through Configuration Control Requests (CCRs) and Configuration Control Boards (CCBs). CCRs must be submitted to the appropriate CCB in order to change items under configuration control. For each configuration item, the configuration management database contains the name of the CCB, which controls the item.

6.1.2.1 Reporting Documentation

Many forms are used in the ECS SDPS configuration control process. These forms are available on the ECS Internal Server. Each form contains instructions or is described in a PI/WI. CM retains the completed forms as a record of the changes to configuration items.

6.1.2.2 Review Procedures

Configuration control of baseline documentation defining ECS SDPS requirements, design, and as-built software is implemented by CCBs. CCB PIs discuss the process for review and approval of changes on the ECS SDPS Project and provide details on the responsibilities and membership of each CCB. Additional details on the ECS CCBs are included in the Release Configuration Management Plan for SDPS (CDRL 102-CD-003).

6.1.2.3 Storage, Handling and Delivery of Project Media

CM is responsible for establishing and controlling the Software Development Library (SDL). The SDL is the repository for source code and test materials, including test scripts, input data, output data, and test results. CM ensures that CCB-authorized material is archived and stored in the library, and that no unauthorized changes are made to established software baselines. The Data Management Office (DMO) controls hardcopy material after approval by a CCB.

CM is responsible for maintaining accountability for materials in the SDL, and for making and releasing copies to internal and external users.

Archival of ClearCase is described in CM procedures for the backup and storage of Version Object Base (VOB) data.

6.1.3 Software Development Library

The Configuration Management Plan for the ECS Project describes the process and tools for maintaining the SDL. The ClearCase tool provides for the management of the code. The CM organization provides scripts on top of ClearCase to provide better controls over the baseline and to help the developers.

Developers use ClearCase during the development of the software from design through unit test. After the unit test phase, a merge request is submitted and approved (refer to section 4.1.2 for more details). Upon merging the code to the appropriate baseline, the CM group builds the software for use in the integration lab. At the point when the release or patch is ready for delivery outside of the EDF, the CM group performs a final build of the software and creates the

appropriate tar files for delivery. Code or executables are only delivered to the sites through the CM organization.

6.2 Software Migration

ECS SDPS software follows the processes and flow described in CDRL 102-CD-003, Release Configuration Management Plan for SDPS, as it migrates from the individual programmer levels, to the segment level, and then to the ECS SDPS system-level. In addition, software migrates from each ECS SDPS release to its following release in a controlled manner. The Version Description Document (VDD) is an integral part of all release deliveries by documenting the contents of the release. Functional Configuration Audits (FCA) and Physical Configuration Audits (PCA) verify all formal deliveries. Detailed information on VDDs, FCAs and PCAs can be found in the Release Configuration Management Plan for SDPS CDRL 102-CD-003.

6.3 Configuration Management at Operational Sites

The emphasis changes from development to maintenance and operations when ECS SDPS products are delivered to operational sites. Details of these activities are also included in CDRL 102-CD-003, Release Configuration Management Plan for SDPS. Maintenance and Operations PIs and the Maintenance and Operations Management Plan (601-CD-001) provide details for operational site configuration management, including the specific roles and responsibilities of the operational CCBs.

6.3.1 Configuration Status Accounting

Configuration status accounting consists of recording and reporting information about the configuration status of the ECS SDPS Project's documentation, hardware, and software products, throughout the Project life cycle. Periodic and ad hoc reports keep ECS SDPS Project management and ESDIS informed of configuration status as the Project evolves. Reports to support reviews and audits will be extracted as needed. CM maintains CM Web pages. Configuration Status Accounting is described in CDRL 102-CD-003 Release Configuration Management Plan for SDPS. Project instructions provide additional details on configuration status accounting.

6.3.2 Configuration Audits

Configuration auditing is the means by which management ensures that both the technical and administrative integrity of the product are being met throughout the Project development life cycle. The audit process consists of CM self-audits, ECS SDPS Project internal audits, and formal audits conducted by ESDIS. Formal audits are a prerequisite to formal approval of the "as-shipped" configuration. They provide verification that each CI in the baseline being shipped is logically related to the corresponding CI in preceding baselines. Configuration audits (including FCAs and PCAs) are described in CDRL 102-CD-003 Release Configuration Management Plan for SDPS.

This page intentionally left blank.

Abbreviations and Acronyms

CASE	Computer Aided Software Engineering
CCB	Configuration Control Board
CCR	Configuration Change Request
CDR	Critical Design Review
CDRL	Contract Data Requirements List
CM	Configuration Management
CMM	Capability Maturity Model
CM	Configuration Management
COTS	Commercial Off-the-shelf
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSMS	Communications and System Management Segment
CSR	Consent to Ship Review
CSU	Computer Software Unit
DAAC	Distributed Active Archive Center
DCN	Document Change Notice
DID	Data Item Description
DMO	Data Management Office
ECS	Earth Observing System Data and Information System (EOSDIS) Core System
EDF	ECS Development Facility
EOS	Earth Observing System
EOSDIS	Earth Observing System Data and Information System
FCA	Functional Configuration Audit
GSFC	Goddard Space Flight Center
ILN	Integrated Logic Network
IV&V	Independent Verification and Validation

LAN	Local Area Network
M&O	Maintenance and Operation
NASA	National Aeronatics and Space Administration
NCR	Non-conformance Report
PAR	Performance Assurance Requirements
PCA	Physical Configuration Audit
PDL	Program Design Language
PDR	Preliminary Design Review
PI	Project Instruction
QA	Quality Assurance
RRR	Release Readiness Review
SDF	Software Development Folder
SDL	Software Development Library
SDP	Software Development Plan
SDPS	Science Data Processing Segment
SDR	System Design Review
SEPG	Software Engineering Process Group
SLOC	Source Lines of Code
TRR	Test Readiness Review

Glossary

Acceptance Testing	Verification that is conducted to determine whether a release satisfies its acceptance criteria and that provides the Government with information for determining whether the release should be accepted.
Baseline	Identification and control of the configuration of software (i.e., selected software work products and their descriptions) at given points in time.
Build	An assemblage of threads that produces a gradual buildup of system capabilities. Builds are combined with other builds and threads to produce higher-level builds.
ClearCase	A COTS automated tracking and control tool by Atria in use on the ECS project.
Commercial Off the Shelf (COTS)	COTS is a product, such as an item, material, software, component, subsystem, or system, sold or traded to the general public in the course of normal business operations at prices based on established catalog or market prices.
Computer software configuration item (CSCI)	A configuration item comprised of computer software components (CSCs) and computer software units (CSUs).
Computer Software Component (CSC)	A distinct part of a computer software configuration item. CSCs may be further decomposed into other CSCs and computer software units.
Computer Software Unit (CSU)	An element specified in the design of a Computer Software Component (CSC) that is separately testable.
Configuration	The functional and physical characteristics of hardware, firmware, software or a combination thereof, as set forth in technical documentation and achieved in a product.
Configuration change control	The systematic coordination, evaluation, and release of approved changes to an established baseline.
Configuration Change Request (CCR)	A document that request and justifies a change to a configuration.
Configuration Item (CI)	An aggregation of hardware, firmware, software or any of its discrete portions, which satisfies an end use function and is designated for configuration management.

Configuration Management Tool	Software tool for doing automated configuration management of source code, scripts, documentation, and other computer files.
Consent to Ship Review (CSR)	Review to determine the readiness of a release for transition to sites for acceptance testing.
Critical Design Review (CDR)	A detailed review of the "code-to" design is performed, including details such as COTS selection and heritage software, and their interfaces with the rest of the release software. Critical Design Review signifies that the CSCI is ready to begin development (for the corresponding release).
DDTS	Distributed Defect Tracking System. A COTS problem reporting system used by the ECS project.
ECS	EOSDIS Core System
Hardware	That combination of subcontracted, COTS, and government furnished equipment (e.g., cables and computing machines) that are the platforms for software.
Hardware Configuration Item (HWCI)	A configuration item comprised of hardware components.
Incremental Release Review (IRR)	At the conclusion of the Preliminary Design phase an IRR is conducted, to promote a common understanding between the ECS SDPS project and ESDIS of the capabilities that ECS SDPS must provide. This review includes an understanding of the requirements through use case scenario presentations.
Independent Verification and Validation (IV&V)	Verification and validation performed by a contractor or government agency that is not responsible for developing the product or performing the activity being evaluated. IV&V is an activity that is conducted separately from the software development activities governed by the ECS contract.
Informal NCR	A Non-Conformance Report tracked in DDTS during the development phase, whose longevity ends at TRR. It is a non-reportable item.
Inspection	The visual, manual examination of a software work product and comparison to the applicable requirement or other compliance documentation, such as engineering drawings. A specific type of peer review.
Integration	The orderly progression of combining lower level software and/or hardware items to form higher level items with broader capability.

Maintainability	The measure of the ability of an item to be retained in or restored to a specified condition when maintenance is performed by personnel having specified skill levels, using prescribed procedures and resources, at each prescribed level of maintenance and repair.
Non-conformance	The failure of a unit or product to conform to specified requirements.
Non-Conformance Report (NCR)	A report of non-conformance found in a test phase. NCRs are tracked in DDTS and become reportable items after TRR.
Preliminary Design Review (PDR)	PDR is held for all CSCIs. The PDR addresses the initial design of the system down to the CSU level and will be held in the case of Release A.
Problem Tracking Tool	Software tool for doing automated tracking of problems found in the software, as well as the status of the resolution of those problems. DDTS selected for ECS.
Process	A logical sequence of tasks by which a job is accomplished.
Product baseline	The baseline which established the “as-built” configuration for system-level integration and testing (I&T) and independent acceptance testing. This baseline is validated by functional and physical configuration audits, and reviewed and approved by the Goddard Space Flight Center (GSFC) as part of release readiness review.
Prototype	Prototypes are focused developments of some aspect of the system which may advance evolutionary change. Prototypes may be developed without anticipation of the resulting software being directly included in a formal release. Prototypes are developed on a faster time scale than the incremental and formal development track.
Prototyping	The construction of a solution of a design or implementation problem, the feasibility of which needs to be determined as early as possible in order to arrive at a critical decision.
Quality Assurance	A subset of the total performance assurance activities generally focused on conformance to standards and plans.
Release	A delivery containing a specific set of requirements introduced in the SSRP and agreed upon at the IRR.
Release Organization	One of the functional subdivisions of the ECS, i.e., A/TRMM, B/EOS AM-1 Landsat-7, C, D.

Release Readiness Review (RRR)	Conducted at the ECS system level for a GSFC project review team upon completion of release acceptance testing. The IATO leads the RRR to determine, with the GATT and the COTR, if the release is ready for transition to IV&V and Operations.
Requirement	A statement to which the developed system must comply. Varieties of requirements: levels 2, 3, 4; performance, functional, design, interface.
Reusable Software	Software developed in response to the requirements for one application that can be used, in whole or in part, to satisfy the requirements of another application.
Risk	An event, action, or thing with a: 1) potential loss associated with it, 2) uncertainty or chance involved, 3) some choice involved.
Risk Analysis	Risk analysis is the application of a standardized methodology to determine threats, risk factors, vulnerability exposures, and potential losses. Risk analysis satisfies an organization's need to protect the assets in which it has invested. It also identifies an organization's potential performance problems and the adverse affects these problems might present to the organization's ability to meet its obligations. Finally, risk analysis is a mechanism by which management can address these problems according to relative importance based on financial analysis, and can develop reasonable and cost-effective safeguards.
Risk Management	The process of identifying, measuring, and controlling risk factors associated with a program development and/or support activity.
Scenario	A description of the operation of the system in user's terminology, including a description of the output response for a given set of input stimuli. Scenarios are used to define operations concepts.
Site Operational Readiness Review (SORR)	SORRs shall be conducted to review the readiness of site operations to receive ECS software for a release. SORR may be held coincident with CSR.
Segment	One of the two functional subdivisions of the ECS, i.e., EMOS, and SDPS.
Software	A combination of associated computer instructions and computer data definitions required to enable the computer hardware to perform computational, data manipulation, and control functions (to include parameters and procedures associated with software products).

Software Development Folder	A repository for a collection of material pertinent to the development or support of software. Contents typically include (either direct or by reference) design considerations and constraints, design documentation and data, schedule and status information, test requirements, test cases, test procedures, and test results.
Software Development Library (SDL)	A generic term which describes a controlled collection of software, documentation, and associated tools and procedures used to simplify the development and subsequent support of software. An SDL provides storage of and controlled access to software in both human readable and machine readable form. Also, it may contain management data pertinent to the software development project.
Standards Checking	The process of checking whether or not source code and shell scripts follow prescribed coding standards.
Subsystem	A combination of sets, groups, etc., which performs an operational function within a system and is a major division of a system
System	A composite of equipment, skills and techniques capable of performing and/or supporting an operational role. A system includes all equipment, related facilities, material, software, services, and personnel required for its operation and support to the degree that it can be considered a self-sufficient item in its intended operational environment.
System Design Review (SDR)	At the conclusion of the system design phase, a formal SDR is conducted to address the system architecture and the definition of the system level interfaces. This review signifies the completion of system design.
System Requirements Review (SRR)	The SRR encompasses a complete review of the ECS specification and the EOS/EOSDIS Requirements that drive the specification, it promotes a common understanding between the Project and the Contractor of the capabilities that ECS must provide. This review creates the initial ECS System baseline. It signifies the end of Conceptual Phase and the start of Definition Phase.
Test	A procedure or action taken to determine under real or simulated conditions the capabilities, limitations, characteristics, effectiveness, reliability or suitability of a material, device, system or method.

Test Readiness Review (TRR)	Conducted at the end of integration tests to assess the readiness of the software to enter the acceptance test process.
Thread	A set of components (software, hardware, and data) and operational procedures that implement a scenario, portion of a scenario, or multiple scenarios.
Unit	An assembly of any combination of parts (classes / methods) mounted together, which are normally capable of independent operation in a variety of situations.
Validation	The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.
Verification	The process of evaluating the products of a given development activity to determine correctness and consistency with respect to the products and standards provided as input to that activity.
Version	1) the culmination of a series of ECS releases, in conjunction with incorporation of SCF-developed science data processing software and unique site capabilities 2) a software file revision indicator