Division 38
Instruments and Science Data System

Planetary Data System (PDS)
Version 4

Software Management Plan

VERSION 0.1
August 31, 2013

Planetary Data System (PDS)
Version 4

Software Management Plan

Document Owner:

Signature on file at PDS Library

_____
<Project Manager's name> Date

Approved By:

Signature on file at PDS Library

_____
<name> Date

Approved By:

Division 38 Process Engineer

_____
<Div 38 Process Engineer Name> Date

Concurred By:

Signature on file at PDS Library

_____
Andrew Bingham Date
Instrument Software and Science Data Systems
Section Manager (398)

PDS No.
Issue Date: < date >

Jet Propulsion Laboratory
California Institute of Technology

For PDS Configuration Item:

REVIEW SIGNATURE PAGE

Signature on file at PDS Library

<Name> Date
<Role>

<Name> Date
<Role>

PDS No.
Issue Date: < date >

Jet Propulsion Laboratory
California Institute of Technology

For PDS Configuration Item:

CHANGE LOG

Rev	Issue Date	Pages Affected	Change Description
0.1	8/31/2013	All	Initial draft.

TBD Items

Section	Planned Closure Date	Description	Completed

Table of Contents

Index of Tables

List of Figures

Tailoring

This Software Management Plan uses the pre-tailored Section 398 Software Management Plan adapted for this project and is 100% compliant with the Section plan. The Section 398 SMP tailoring has been reviewed and approved using the Tailoring Approval Process. The tailoring record resides in DocuShare at the Section 398 PAL As such, this document is SDR and SDSP compliant. No waivers are required beyond those granted for the Section 398 SMP. The tailoring record will be reviewed at major changes to the processes.

1 Overview

1.1 Project Summary

For over fifteen years, the Planetary Data System (PDS) has been NASA's official data system for archiving and distribution of data from planetary exploration missions. It has been a leader in defining data standards, working with missions and instrument teams, and developing data system technologies. The PDS has been instrumental in changing the scientific culture by working with the planetary science community to publicly release and peer review the data it captures. It has also been used as a model by other science data systems interested in establishing distributed scientific networks organized by independent discipline nodes at facilities that are doing leading-edge scientific research.

While PDS has been a leader in developing and exploiting new technologies and ideas, an increasing workload and substantial increases in the volume of delivered data are now threatening the system's ability to accomplish its primary missions of both archiving planetary science data and distributing it to working scientists. PDS identified these challenges in its Roadmap published in 2006. In addition to these challenges, the ten year Roadmap outlined several goals including improving the PDS data standards, increasing user services by leveraging newer technologies and technical standards, and re-architecting PDS to ensure efficient operations of the system while supporting the increasing demands on PDS by both the data providers and end users.

In response to these challenges and goals, PDS has developed a plan for the next generation called "PDS4". The vision for PDS4, as defined by the PDS Management Council at its April 2008 meeting, includes:

- Simplified, but rigorous, archiving standards that are consistent, easy to learn, and easy to use
- Adaptable tools for designing archives, preparing data, and delivering the results efficiently to PDS
- On-line services allowing users to access and transform data quickly from anywhere in the system
- A highly reliable, scalable computing infrastructure that protects the integrity of data, links the nodes into an integrated data system, and provides the best service to both data providers and users

1.2 Purpose, scope and objectives

This software management plan establishes and governs the lifecycle development approach for software being developed at JPL for the PDS. This plan provides a shared vision allowing key stakeholders to participate productively throughout the development process. It specifies responsibilities, plans, processes, policies, and guidelines. Individuals participating in software development for the PDS must comply with this plan or obtain approval for a waiver for specific portions of this plan.

- This SMP satisfies all NASA and JPL requirements for software development and management plans and complies with all project-level controlling documents (see Table 1.4-1).
- This SMP defines development activities and responsibilities at a level of detail that identifies the required resources and supports the monitoring of progress, the allocation of resources, the management of risk, and the attainment of the desired level of product quality.
- The scope of this software effort is further described in the PDS Work Agreements (WA).
- In order to maintain a single authoritative source, dynamically changing items such as budgets and schedules are not included in this document, but are referenced. In addition, where appropriate, the PDS creates individual plans for specific management planning functions. The PDS considers those plans an integral part of the SMP requiring similar review and approval controls.

Table 1.2:1 identifies the software systems with the respective software classification that shall comply with this plan.

Program Set name	**Identification Number**	**Software Classification**	**Safety Critical (Y/N)**	**Justification**
Infrastructure		*Class C*	*N*	*Failure will not cause loss of data nor compromise mission objectives*
Tools		*Class C*	*N*	*Failure will not cause loss of data nor compromise mission objectives*

Table 1.2:1 Software Classification & Identification

This SMP pertains to the entire PDS project.

1.3 Assumptions and Constraints

PDS follows the Division 38 Local Procedures. These procedures are in the Section 398 Local Procedures Library and are:

- Software Project Planning
- Software Cost Estimation
- Software Management and Development
- Software Requirements Development
- Software Requirements Management

- Software Risk Management
- Software System Architectural Design Definition
- Software Design and Implementation
- Software Inheritance and Reuse
- Software Delivery
- Software Project Monitoring and Control
- Software Configuration Management
- Software Build, Integration and Test
- Software Verification
- Software Validation
- Software PPQA
- Software Peer Review
- Software Measurement and Analysis
- Instrument Flight Software Design (where applicable)
- Software Development for Science Data Systems (where applicable)
- Software Maintenance and Adaptation for Science Data Systems (where applicable)
- Decision Analysis and Trade Studies

1.4 S/W Task Receivables & Deliverables

Receivables and deliverables are the handoff of a tangible asset or product from one team entity to another (e.g. S/w source code is a deliverable, but a signed software requirements document is a receivable. The following table lists the major deliverables for this task.

Deliverable Name	Receiver	Due Date	Responsibility (person|Role|subsystem)
PDS4 Standards	PDS Project Manager	Coordinated with Project Schedule	PDS EN System Engineering Lead
PDS4 Software	PDS Project Manager	Coordinated with Project Schedule	PDS EN Development Lead

Table 1.4:1 Deliverables

The following table lists the major receivables for this task.

Receivable Name	Deliverer	Due Date	Responsibility (person|Role|subsystem)
PDS Level 1, 2 and 3 Requirements	PDS Management Council	Coordinated with Project Schedule	PDS Management Council

Table 1.4:2 Receivables

1.5 Schedule and Budget Summary

The PDS project is an ongoing activity and does not follow the traditional project phased lifecycle. The PDS project budget is managed via the NASA Budget Process "Planning Programming Budget Execution (PPBE)" as documented in:

http://nodis3.gsfc.nasa.gov/iso_docs/pdf/H_OWI_7410_N_005_A_.pdf

The PPBE and budget actuals reports can be found in the PDS DocuShare library:

https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-198000

The PDS schedules and milestones are managed using FastTrack and can be found in the PDS DocuShare library:

https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-24529

1.6 Document and Records Management

The following table lists all documents and records that will be produced by PDS and includes approval authority, owner, retention and location for each.

Document	Due At	Number	Location	Update Frequency	Document/ Record owner	CM Repository	Approval Authority Role or Person	Retention
Software Management Plan (SMP) This Document	N/A	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-24205	Yearly	Project Manager	PDS DocuShare Library	Project Manager, 398 Line	Duration of project and archived with software
PDS4 Project Plan	PDR Draft	N/A	http://pds-engineering.jpl.nasa.gov/pds2010/pds4-proj-plan-07172013.pdf	As Needed	Project Manager	Subversion	PDS MC	Duration of project and archived with software
Budget Planned/Actuals	Begin Fiscal Year	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-198000	Yearly	WBS Leads	PDS DocuShare Library	NASA Program Exec	Duration of project and archived with software
Cost Estimates	Begin Fiscal Year	N/A	http://wa.jpl.nasa.gov	Yearly	WBS Lead	Work Agreement (WA) System	Project Manager, 398 Line	Duration of project and archived with software

Document	Due At	Number	Location	Update Frequency	Document/ Record owner	CM Repository	Approval Authority Role or Person	Retention
PDS Level 1, 2, and 3 Requirements	PDR Final	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-751880	As Needed	Project Manager	PDS DocuShare Library	PDS MC	Duration of project and archived with software
System Architecture Specification	PDR Final	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-497062	As Needed	WBS Leads	PDS DocuShare Library	Project Manager, SDWG	Duration of project and archived with software
General System Software Requirements Document (SRD)	PDR Draft CDR Final	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-1295207	As Needed	Development Lead	PDS DocuShare Library	Project Manager, SDWG	Duration of project and archived with software
Software Requirements and Design Document (SRD/SDD) (Per Component)	PDR Draft CDR Final	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-107575	As Needed	Development Lead	PDS DocuShare Library	Project Manager, SDWG	Duration of project and archived with software
Software Integration and Test Plan (SITP-1) – Planning	CDR Draft Begin Implementation Final	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-198102	Major Delivery	Operations Lead	PDS DocuShare Library	Project Manager	Duration of project and archived with software

Document	Due At	Number	Location	Update Frequency	Document/ Record owner	CM Repository	Approval Authority Role or Person	Retention
Software Integration and Test Plan (SITP-2/3) – Procedures and Reports	CDR Draft Prior to first tests	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-198102	Major Delivery	Operations Lead	PDS DocuShare Library	Project Manager	Duration of project and archived with software
Software User's Guide (UG)	Implementation Draft Delivery Final	N/A	http://pds-engineering.jpl.nasa.gov/index.cfm?pid=145&cid=159	Major Delivery	Development Lead	PDS DocuShare Library	Project Manager	Duration of project and archived with software
Software Operator's Manual (SOM)	Implementation Draft Delivery Final	N/A	http://pds-engineering.jpl.nasa.gov/index.cfm?pid=145&cid=159	Major Delivery	Development Lead	PDS DocuShare Library	Project Manager	Duration of project and archived with software
Release Description Document (RDD)	Prior to Delivery	N/A	http://pds-engineering.jpl.nasa.gov/index.cfm?pid=145&cid=159	Major Delivery	Development Lead	Subversion	Project Manager	Duration of project and archived with software
Meeting Minutes	As Needed	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-21620	N/A	As Appropriate	PDS DocuShare Library	N/A	End of project

Document	Due At	Number	Location	Update Frequency	Document/ Record owner	CM Repository	Approval Authority Role or Person	Retention
MMR Presentations	Monthly	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-24505	N/A	WBS Leads	PDS DocuShare Library	PDS MC	End of project
Schedule	PDR Draft	N/A	https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-24529	Monthly	WBS Leads	PDS DocuShare Library	Project Manager	End of project
Issues (Standards)	N/A	N/A	http://pds-jira.jpl.nasa.gov	As Needed	Operations Lead	PDS CCB JIRA	N/A	End of project
Issues (Software)	N/A	N/A	https://oodt.jpl.nasa.gov/jira/browse/PDS	As Needed	Development Lead	PDS Software JIRA	N/A	End of project

Table 1.6:1 Project Documents and Records

1.7 Notation

No special notation is used in this document.

1.8 Controlling Documents

JPL and NASA Controlling Documents		
Document Number	**Document Description**	**Date/Rev.**
JPL Rules ID 57653	Software Development Requirements	7/30/2007 Rev
JPL Rules ID 78124, 78125, 78126, 78127, 78128, 78129, 78130, 78131, 78132, 78133, 78134, 78135, 78136, 78137, 78138, 78139, 78140, 78141, 78142, 78143, 78144, 78145	Software Development Standard Processes	5/21/2009 Rev
JPL Rules ID 35163	Project Reviews	8/01/2006 Rev
JPL Rules ID 43913	Design, Verification/Validation, and Operations Principles for Flight Systems	12/11/2006 Re
JPL Rules ID 58032	Flight Project Practices	3/06/2006 Rev

Table 1.8:1 JPL and NASA Controlling Documents

Project Controlling Documents		
Document Number	**Document Description**	**Date/Rev.**
N/A	PDS4 Project Plan	7/17/13 V1.0

Table 1.8:2 Project Controlling Documents

1.9 Applicable Documents

Applicable Documents		
Document Number	**Referenced Document Name**	**Date/Rev.**
N/A	Guide to Software Training at JPL	N/A
	398 SMP Template	9/1/2008 Rev 1
DocID 68612	Software Risk Management Handbook	11/12/2004/rev. 0
DocID 71712	Section 398 Software Development Procedures	Section 398 PAL
JPL D-25798, Rev. 0	Software Reviews Handbook on the Software Web Site	Rev 0. 2003
DocID 46512	Disseminating JPL-Developed Software	Rev. 2
DocID 56614	Release of Scientific or Technical Information	Rev. 7

Applicable Documents		
Document Number	**Referenced Document Name**	**Date/Rev.**
DocID 36472	Releasing Information for Unlimited External Distribution	Rev. 7

Table 1.9:1 Applicable Documents

1.10 Evolution of Plan

All work defined by this task is documented and approved in the Work Agreements. This SMP simply lays out how the task will be managed and, as such, no further updates to this document will be done with regards to the work implementation.

The following approvals for this document are required:

- PDS Project Manager, Dan Crichton
- Division 39 Process Engineer, Jordan Padams

1.11 Definitions

- The list of acronyms is in Appendix A: Acronyms.
- The Glossary is in Appendix B: Definitions

1.12 Security, Privacy, Safety

The following list identifies PDS safety and security requirements. These security measures shall be followed by all team members.

1. All team members are required to conform to computer security practices that satisfy the JPL requirements as specified in "JPL Information Technology Security Requirements" (JPL Rules!DocID 36852).
2. Flight Software provision not applicable.
3. Team members are not to disseminate or make available any work products, particularly source code, outside the PDS project without authorization from the PDS Project Manager. Any dissemination outside JPL must follow "New Technology, Laboratory Notebooks, and Software Dissemination", (JPL Rules! DocID 56592), the "Export Control Requirement", (JPL Rules! DocID 68033) and the "Release of Scientific or Technical Information Requirement", (JPL Rules! DocID 56614).
4. Team members are to conform to any laboratory safety and security requirements.
5. Team members are to follow all applicable safety requirements defined in the "JPL Standard for Systems Safety"http://rules.jpl.nasa.gov/cgi/doc-gw.pl?DocID=34880 (JPL Rules!DocID 34880) and "Project Software Quality Assurance Planning" (JPL Rules!DocID 44452).

2 Project Organization

2.1 Internal Structure

The organization chart for the PDS Engineering Node can be found here:
http://pds-engineering.jpl.nasa.gov/about_eng/node_org/pds_org_chart.pdf

The following members of the PDS Engineering Node are represented in the organization chart referenced above:

Position	Institution or Company	Contact	Responsibility	Phone and e-mail
PDS Project Manager	JPL	Dan Crichton	Project Management	818-354-9155 dan.crichton@jpl.nasa.gov
PDS System Engineering Lead	JPL	Steve Hughes	Standards Management	818-354-9338 john.s.hughes@jpl.nasa.gov
PDS Development Lead	JPL	Sean Hardman	Development Management	818-354-7188 sean.hardman@jpl.nasa.gov
PDS Operations Lead	JPL	Emily Law	I&T Management	818-354-6208 emily.law@jpl.nasa.gov

2.2 External Interfaces

Position	Institution or Company	Contact	Responsibility	Phone and e-mail
NASA Program Exec	NASA	Bill Knopf	Program Management	202-358-0742 WKNOPF@HQ.NASA.GOV
Program Scientist	NASA	Michael New	Program Science	202-358-1766 MICHAEL.H.NEW@NASA.GOV
PDS MC	Various	See http://pds.nasa.gov/contact/contact.shtml	Management Council	See http://pds.nasa.gov/contact/contact.shtml

Table 2.2:1 External Interfaces

2.3 New Technology Reporting (NTR)

PDS complies with JPL's new Technology, Laboratory Notebooks, and Software Dissemination,http://rules.jpl.nasa.gov/cgi/doc-gw.pl?DocID=56592 Rev. 3, DocID

56592 for reporting new technology, documenting technology development and disseminating new technology.

2.4 Roles and Responsibilities

The organizational roles and responsibilities are defined in the PDS4 Project Plan.

3 Management Processes

3.1 Start-up Plans

No specific start-up activities are required for this task. Existing resources will be used and the staff has been selected in advance of the work.

3.2 Estimation Plan

The cost estimate for the PDS4 effort was calculated as part of the total cost estimation for the PDS project software. To validate the cost, aspects of the PDS3 effort were used for analogy.

3.3 Staffing Plan

Details of the staffing profile (planned and actual) are captured in Institutional Budgeting Tool (IBT). The specific staffing profiles for each WBS element for a given fiscal year are captured in the Work Agreements (WA) for that fiscal year.

3.4 Resource Acquisition Plan

The procurements for the project are covered by PDS in-guide activities under existing PDS task plans. As a result, there is not a need for a formal procurement plan for the PDS4.

4 Work Plan

4.1 Work Activities

The project work activities are captured at a high-level in the PPBE and in the WA's for a given fiscal year.

4.2 Schedule Allocation

The schedule is managed in FastTrack and lists all activities necessary to complete this project.

4.3 Resource Allocation

The resource allocation is captured in IBT during the fiscal year planning process. The specific staffing profiles for each WBS element for a given fiscal year are captured in the WA's for that fiscal year.

4.4 Budget Allocation

The budget allocation is captured in the PPBE, entered into IBT and reflected in the WA's for a given fiscal year.

5 Control Plan

All control plans will follow the Section 398 Software Project Monitoring and Control in DocuShare.

5.1 Requirements Control Plan

PDS will follow the Section 398 Requirements Management local procedure and the Section 398 Software Requirements Development in DocuShare. Requirements management, development and control documents are under CM control and can be found as indicated in the following table:

Requirements Documents and Records	Location (Link)
PDS Level 1, 2 and 3 Requirements	*https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-751880*
General System Software Requirements Document (SRD)	*https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-1295207*
Requirements Mapping (Bi-Directional Trace Matrix between Level 3, 4, and 5 Requirements)	*https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-1295209*
Software Requirements and Design Documents (SRD/SDD) (Level 4 and 5 requirements for each component.)	*https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-107575*

Table 5.1:1 Requirements Documents and Records

Requirements changes will be indicated by creating new versions of the bi-directional trace matrix. All versions of the trace will be kept until the end of the task. Bi-directional traces will be reviewed for accuracy as time permits and any discrepancies will be corrected. To the extent possible based on available personnel, the requirements will be peer reviewed by a desk check method. This will primarily be done by the developer(s) who will have the responsibility to validate each requirement before it is accepted. Internal interfaces will be described in the code comments.

5.2 Schedule Control Plan

The schedule will be monitored on a periodic basis by the PDS Project Manager. Any deviation from the original schedule will be analyzed to determine if the commitments can still be met. If commitments cannot be met, the Project Manager and Project Leads will meet with relevant stakeholders to mitigate the schedule slip. Mitigations will depend on project priorities and may include re-prioritizing requirements, re-planning the schedule, bringing on more help or accepting the schedule and delivering late. Schedule is included in Monthly Report delivered to the PDS Program Manager.

5.3 Budget Control Plan

PDS will follow the Section 398 Software Project Monitoring and Control Local Procedure in the Section 398 Local Procedure Library in DocuShare to monitor the budget. Table 5.7-1: Metrics Collection and Reporting lists records and location related to budget control. The cost estimate location is in Table 1.6-1: Project Documents and Records.

The Budget Control plan provides updated budgets for the **current iteration**, current fiscal year, and/or the project overall. The Budget Control Plan is a mechanism to enable the Project Management to estimate and anticipate budget variances and longer term (>1 year) expected budgeting for the project.

The Budget Control Plan includes all relevant information about the financial aspects of the project. It includes metrics such as anticipated effort hours, updated financial and variance information, and anticipated resource requirements.

Prior to the commencement of iterations a budget will be compiled. It will include a listing of resources assigned for the iteration, an estimation of the effort necessary to complete the iteration, and the financial costs associated with the iteration.

Briefly stated, it is important that spending should not exceed the budgeted amount. The control aspect is a mechanism for monitoring when actual cost exceeds budgeted cost, as well as the actions to be taken should this happens.

5.4 Quality Control Plan

PDS will follow the Section 398 Local Procedure for Software PPQA in the Section 398 Local Procedure Library in DocuShare. Table 5.4-1: Quality control Processes and Records list the schedule for PPQA audits.

Item to audit	Frequency	Record Location
CM Process	*Bi-annually beginning in 2012*	*DocuShare*
Adherence to Coding Standards	*Yearly beginning in 2012*	*DocuShare*
Budget Control Process	*Annually beginning in 2012*	*DocuShare*
Delivery Process	*Annually beginning in 2012*	*DocuShare*

Table 5.4:1 Quality Control Processes and Records

Quality metrics will be collected and reported according to the table 5.7:1 in Section 5.7, Metrics Collection Plan.

Quality metrics will be analyzed and tracked for quality trends and actions will be taken as described in Section 5.8:1, Data Collection and Analysis. If the analysis shows the quality of the software could be improved, steps will be taken to correct the root cause of any quality deficiencies. The analysis will be reported to line management at monthly quiet hours and will include any root causes and corrective actions. Line management will offer assistance in improving quality if this becomes necessary.

5.5 Reporting Plan

Table 5.5:1 describes meetings, reviews and reporting that will be used as the PDS Reporting Plan.

Item to Report	Venue	Frequency	Review Type	Record Location
Metrics	*MMR*	*Monthly*	*Management Review*	DocuShare
Progress-Budget	*MMR*	*Monthly*	*Management Review*	DocuShare
Progress-Schedule	*MMR*	*Monthly*	*Management Review*	DocuShare
Anomaly Status	*Online*	*As Needed*	*Management Review*	JIRA
Test Results	*Delivery Confirmation*	*As Needed*	*Management Review*	DocuShare

Table 5.5:1 Reporting Plan

5.6 Stakeholder Involvement Plan

The involvement of stakeholders is monitored by the Project Manager to ensure good communication through monthly teleconferences and face-to-face meetings, demonstrations and presentations. Records are kept in the PDS Management Council repository at:
http://atmos.nmsu.edu:8080/myapp-0.1-dev/pwdhtml.jsp.

5.7 Metrics Collection Plan

PDS will follow the Section 398 Software Measurement and Analysis Local Procedure in the Section 398 PAL in DocuShare. This procedure describes the measures to be collected by all projects and tasks. The following table describes metrics to be reported, frequency of collection, the analysis methods, type of metric, source of the metric reporting method and storage location and alarm threshold.

Measurement		Description/Definition	Collection Frequency	Data Source	Collection Responsibility	Analysis Procedure	Reporting Mechanism, Venue, and Audience
Requirement Count		Typically the number of Level 4 or level 5 requirements. Include any derived requirements	Monthly	Track from requirements document	Development Lead	Trend the total number of requirements	Report trends to management
Effort		Effort—Planned and Actual Monthly FTE	Monthly	IBS	WBS Lead	Compare planned vs actuals	MMRs or periodic status reports
Unplanned Work (Progress)		Unplanned change requests	Monthly and by build/ Release	Defect tracking tool	Development Lead	Estimate the effort required to implement the change requests	Report the total amount of unplanned work to management at MMR or periodic status reports

Measurement		Description/Definition	Collection Frequency	Data Source	Collection Responsibility	Analysis Procedure	Reporting Mechanism, Venue, and Audience
Coding Progress (Progress)		Cumulative count of number of modules or packages planned to be completed vs number of modules actually completed per month	Monthly and by build/ Release	CM system or issue tracking system	Development Lead	Compare actuals to planned values	Report this development progress metrics to management
Estimate to Complete (Progress)		Effort – Planned Monthly FTE	Monthly and by build/ Release	Spreadsheet or effort loaded schedule	Development Lead	Regularly assess ability to complete	MMRs or periodic status reports
Defects (Quality)		The number of defects recorded to date	Build / Release	JIRA	Development Lead	Monitor the rate of modules completed vs modules requiring rework as a quality measure	MMRs or periodic status reports
SLOC (Size)		Cumulative Total Logical Lines in CM. Count lines of code using institutionally supported code counter, either SLIC or NCSL	Yearly	SLIC	Development Lead	Track CM'd code against expected lines of code	Section Audit

Table 5.7:1 Metrics Collection and Reporting

5.8 Data Collection and Analysis

Collection of data measures shall be performed by the WBS Leads using the data sources identified in Table 5.7:1 Metrics Collection and Reporting. Measures will also be retained in the PDS library for the duration of the work and submitted to the Section 398 repository at the completion of the task.

The Project Manager will review and summarize the measurement data and trends and report as indicated in Table 5.7:1 Metrics Collection and Reporting.

Based on collected development progress data, the WBS Leads shall, with concurrence of the Project manager, take action to keep the development on the planned schedule and budget, including

- Implementation of risk mitigation plans;
- Management and assignment of reserves and margins;
- Revision of development plan schedule, receivables and deliverables;
- Initiation of requirements change process
- Reallocation of workforce and other resources

Action items consistent with monitoring and control will be assigned, monitored and tracked to closure to ensure the task stays on schedule and in budget.

5.9 Risk Management Plan

PDS will follow the Section 398 Software Risk Management Procedure in the Process Asset Library in DocuShare. The PDS Risk Assessment and Management plan is captured in the PDS4 Project Plan.

5.10 Change Control Plan

Software development is an ongoing process. Change control applies to the entire development lifecycle and to all types of changes from requirements changes to code changes. At each phase, changes can be handled slightly differently. This Section describes how PDS will implement a change control process.

As the system grows, developers and testers will detect problems with older designs and implementations. The process will lead to recommendations to fix problems or change the design approach. Problem Report (PRs) record recommended problem fixes. Change Requests (CRs) record suggestions to change the current design approach. A final category is changes that are needed as the result of normal development and these are not tracked.

The impact of potential problem fixes or design changes have highly variable impact. PDS change policies will differentiate between those changes that have a relatively low impact and those that have a significant impact on the system. If the characteristics of the change fit the following criteria, the change may be classified as one with minimal or small impact:

- All aspects of the change are under the purview of the same developer.
- The impact of the change occurs within a single executable. The outcome will not cascade to other segments of the software module.
- The change impacts relatively few lines of code.
- Any change to a component interface has no impact on the behavior of codes that utilize that interface.
- No significant impact to cost or schedule.

If a potential change meets the above criteria, the developer should alert the Development Lead of the problem and the proposed solution. If the developer and the Development Lead conclude that the proposed change is beneficial and has small or minimal impact, the developer will implement the change and the PDS team will not generate a formal record of the modification. The Development Lead has the final approval of these types of changes, but no Change Control Board is required.

If a potential code change meets any of the following criteria, the PDS will require a formal CR or PR to track the execution and effectiveness of the change:

- The scope covers the work area of more than one developer.
- The scope impacts more than one subsystem or executable in the software module.
- The change impacts a large number of modules or requires substantial change to any software module.
- The change in component interface will impact codes that utilize that interface.

- There is a change to the requirements and/or the design

The PDS will utilize JIRA to track formal CRs and PRs. This tool allows authorized users to submit new CRs and PRs, and transition items assigned to them through the system. In addition, it allows the user to attach a document to a CR or PR in order to provide supplemental information.

The Development Lead will review the request or report, and stipulate whether the change is warranted. These decisions will be in collaboration with the relevant stakeholders. Regardless of the formality of Problem Reports and Change Requests, developers must record all changes to software that fall under configuration control in Delivery Memoranda. Metrics will be collected as described in Section 5.7: Metrics Collection Plan.

6 Technical Process Plans

6.1 Process Model Life Cycle

PDS will use the evolutionary software life cycle model depicted in Figure 6.1:1 Evolutionary Life Cycle. The life cycle phases (builds) are delineated in the project schedule.

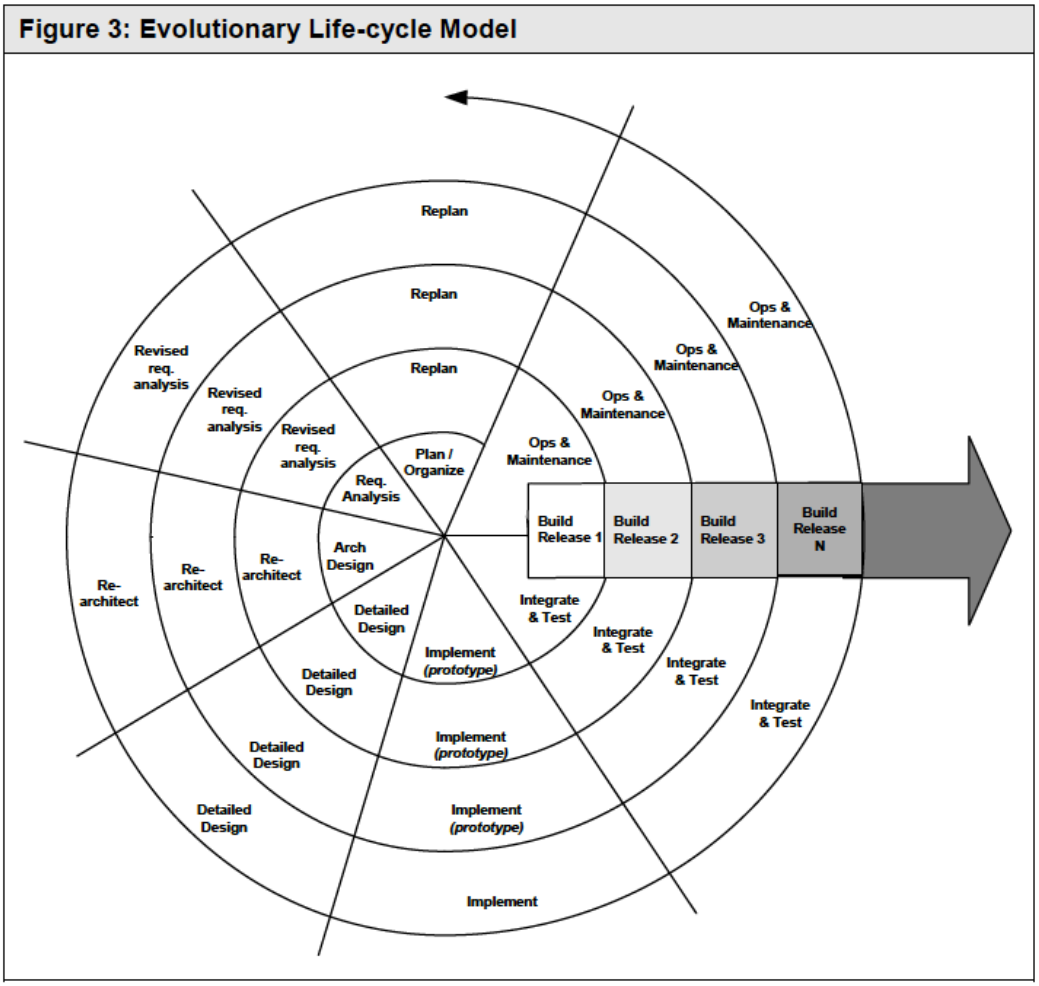

Figure 6.1-1 Evolutionary Life Cycle

6.2 Methods, Tools and Techniques

PDS uses no special methods, tools or techniques other than those described in the Section 398 Local Procedures.

6.3 Infrastructure Plan

NASA/JPL has a world-class computing infrastructure at JPL that is used to support the mission, science, engineering and administration functions of the Laboratory. This includes basic computing services up to server-class facilities and supercomputing. In addition, JPL is connected to several high-speed networks and is working on the upgrade to a 10 Gbit backbone. The PDS infrastructure is captured in: https://bravo-lib.jpl.nasa.gov/docushare/dsweb/View/Collection-197900/Document-1715395

6.4 Architectural Design Activities

As described in Section 6.1 of this document, after the requirements analysis phase, the software design begins. Architectural design specifies the overall system level design including component definition and functional allocation, external and functional area interfaces, protocols, environment, logical flow, development constraints, and design approach. The architecture is documented in the System Architecture Specification:
https://bravo-lib.jpl.nasa.gov/docushare/dsweb/Get/Document-497062

6.4.1 Inheritance

It is anticipated that Version 4 of the PDS system will inherit software from Version 3 of the PDS system. This form of inheritance will not be subject to the review process described below since the development team is already familiar with this software. Other software to be inherited will follow the review process in this section.

An inheritance review will be conducted per the Reviews Handbook on the JPL **Software** Website and Section 7.7, Reviews of this document. Legacy software will include reusable libraries, application software, design patterns, Off-The-Shelf software, etc. that require tailoring and adaptation.

The inheritance software review will be held prior to finalizing the architectural design. The inheritance review purpose will be to establish feasibility and risk, and to estimate the additional development effort required.

Inherited software will be reviewed and evaluated based upon benefits in comparison with the operational risks and life-cycle costs prior to use, using the following criteria:

- Completeness of any support documentation; absence of needed support documents;
- Prior history (includes how it has been used; what problems were encountered);
- Compatibility with project requirements (e.g., does it meet project specifications?);
- Support (e.g., will the inherited code by supported by a knowledgeable person?);
- Test results (results of tests of the inherited code);
- Risks (based upon past history, past issues, test results etc.);

- Cost;
- Recreation of previous test results using as-delivered system wherever possible

The approach to inheritance shall also be reviewed at project design reviews, where appropriate. The outcome of an inheritance review will be approval from project management to proceed or not proceed with the adoption and modification of inherited software.

6.4.2 Interfaces

All interfaces external to the software program set being developed will be identified and characterized in the architectural design. Key internal interfaces between component units will be identified and characterized in the architectural design. Hardware-software interfaces will be documented and controlled with changes systematically communicated to all affected parties. The software architectural design will identify component dependencies. Interface specifications for each component will be created and documented in the applicable Software Requirements and Design Document. External interfaces in legacy code will be physically demonstrated and validated in the "as delivered" system.

6.4.3 Data Definitions

The software architectural design will define data types, valid ranges, and appropriate exception handling for components. The information architecture for PDS4 is captured in the PDS4 Information Model.

6.4.4 Dependencies

The software architectural design will identify and document component dependencies in the applicable Software Requirement and Design Document. Component dependencies will be an input to the build selection process.

6.4.5 Architectural Design Review

Preliminary architectural review will be conducted per Section 7.7, Reviews of this document in accordance with the Reviews Handbook on the JPL Software Website.

If new technologies, tools, or architectural approaches are proposed, a separate formal technology readiness review will be held to assess potential risks prior to the final architectural review.

The architectural design will be peer reviewed by team members and relevant stakeholders prior to the System Design Review. Action Items and RFAs from the review will be collected, tracked to closure and reported at the System Design Review. Operational scenarios will be used to verify that the architectural design includes the required functionality, operating modes, and states.

6.4.6 Architectural Design Maintenance

Upon approval of the System Architecture Specification Document, it will be placed under configuration management. Changes to the architectural design will be subject to

change control processes described in Section 5.8, and all changes will also be documented and placed under configuration management.

6.5 Detailed Design Activities

PDS will follow the Section 398 Design Rules in the SW System Architecture Definition Local Procedure and the Design and Implementation Local Procedure in the Section 398 Software PAL. Detailed design follows architectural design. Detailed design is the component level response to the architectural design including sufficient detail to allow for unambiguous implementation.

Alternative design will be considered where appropriate and where the design options vary due to risk and/or cost. Alternative designs will be peer reviewed and selection will be made by the Development Lead with support from the System Engineering Lead and Project Manager.

The detailed design is initially completed at the System Design Review and documented in the applicable Software Requirement and Design Document. The detailed design is revised and updated for each major delivery build prior to implementation.

The detailed design document will include the following information and be consistent with the software architecture:

- Methods
- Operational concepts
- Definition of operational environments
- Scenarios
- Identification of safety-critical units where applicable
- External Interfaces
- Internal Interfaces
- Adaptability and Reusability when applicable
- Failure Modes
    - Software Analysis (SFTA) for Safety Critical, **Class B** and above
    - Failure Modes, Effects and Criticality Analysis (SFMECA) for Safety Critical, **Class B** and above
- Data Flow diagrams, state transitions or equivalent graphics to facilitate subsequent design and testing where appropriate
- Software defined to the unit level with identification of any critical units for special review and handling

Once the design is complete and approved, a build plan will be developed based on the design, the project and task needs and dependencies. The build plan will be placed under configuration management and can be updated after each software release or as necessary.

6.6 Software Implementation

6.6.1 Implementation Approach

The PDS will generate highly modular software architecture. Modular software is more flexible, maintainable and reusable. The PDS will generate code libraries that may be used in all of the PDS software and executables. Libraries with reusable code will include:

- All of the code modules that handle errors and exceptions,
- All of the code modules that handle datasets, attributes and groups,
- All of the code modules that handle standard file I/O, and
- All of the code modules that interface between the components.

In addition, the code structure will clearly separate those modules that perform algorithmic function from those modules that handle standard I/O functions. By doing so, members of the code will be more easily shared among various stakeholders. Infusion of other software into the development process reduces the workload of the developers who need to generate test conditions, and provides another means for knowledgeable individuals to ensure that the PDS software is operating appropriately.

All code will be written in JAVA on Linux systems using Object Oriented Technology where appropriate. In addition, it is anticipated that development will also consist of Python and UNIX-based scripts. To the extent possible and consistent with Safety & Security issues, the software will employ open source and non-proprietary tools and platforms.

6.6.2 Coding Standards

PDS will adhere to the JPL Institutional Java Coding Standard for Java code development. The Development Lead is responsible for ensuring adherence to this coding standard. For all other programming, the Development Lead will determine whether the software is programmed in a standard manner and whether a peer review is necessary to ensure adherence.

6.6.3 Development Standards

This task follows the JPL Software Development Standard Processes tailored for this task.

6.6.4 Implementation Tools

Table 6.6:1 lists the tools utilized during implementation of the PDS software:

Tool	Description
Eclipse IDE	Tool for aiding in software development.
Apache Maven	Tool for building and packaging software.
Subversion	Tool for software configuration management.

Table 6.6:1 Implementation Tools

6.6.5 Unit Testing

Unit tests will be developed, updated as necessary and documented and performed on all software before delivery to Integration and Test. Wherever possible, unit tests will be automated so they can easily and frequently be run to ensure good quality of software.

Purpose: Unit tests ensure that the smallest unit of testable software, a module or subroutine, meets its functional or performance requirements. Requirements at this level are often inseparable from the component design. The developer notes the requirements as comments in the code itself, rather than in separate requirements documents.
Test subject: Every public module or subroutine
Responsibility: The developer of the module or subroutine
When: As part of the implementation process for the module or subroutine
Where: Development environment
Results: The development of a unit is not complete until it passes all unit tests. Unit tests must verify data types and data ranges as well as functionality and adherence to requirements. The developer reports the unit test status as part of the test readiness review for the component being developed. For Safety Critical Software Units and Mission Critical Software units, the unit tests will be peer reviewed and the unit test results will be analyzed and peer reviewed.

6.7 Software Integration and Test

PDS takes an incremental approach to integration and testing. During each subsystem testing phase, the Tester must test all of the new and updated components delivered for that build plus all unchanged components previously delivered. Creating a test plan for the first build that would also include all of the tests performed for the last build is not feasible. For the first build, the Tester creates a test plan that serves for that build alone. For subsequent builds, the Tester adds new tests to the plan that check the components delivered in that build. This incremental approach allows the Tester to create complete test plans for each build with the minimum of effort.

The “test plan” created for each build is actually a collection of three types of records. After each build, these records are stored in DocuShare as controlled records, Section 1.2. The test records are defined as follows:

Test Process – Describes the processes used for each subsystem test. This process may change from build to build, but the changes should be minor improvements to an established process.

Integration and Test Plan—Identifies and describes the order of integration of the software units and each test run during subsystem test. The tester completes this portion of the plan before beginning the testing. After the test cycle has been completed, the Tester adds the results of each test. The tester adds new tests for each build as new capabilities and components are added to the system. See Table 7.2-1 in Section 7.11 for verification and validation plans.

Test Results—the tester creates a chart for each test (Appendix TBS). The chart includes all of the details needed to run that particular test including:

- Test name
- Components tested
- Capabilities and/or Requirements tested
- Interfaces tested
- Input file description
- Output file description
- Expected value file identification
- Exceptions allowed
- Test procedure

The Tester completes this portion of the chart before beginning the test. After he completes the test, he adds the test results. The Tester reuses each chart for each build completely, except for the results Section.

The records for one build serve as the starting point for the records for the next build. Integration and Test will be approached at several levels. Definitions of verification levels and methods are found in Appendix D.

Table 6.7:1 describes the testing approach to be used for PDS. Every test case used to verify safety critical requirements must be annotated as such.

Purpose	Test Subject	Responsibility	When	Where	Results
Integration Tests	Every Program	System Tester	System test phase of build cycle	Validation Environment	Reviewed during system delivery review
End-to-end tests	Every complete program chain	System Tester	After I & T of program chains	Validation Environment	Reviewed during delivery review of entire build
Regression Tests			After each new build		
• Unit	Subroutines/ modules	Developer		Dev Environment	Reported to I&T before requesting integration
• Acceptance	Every program	Developer	Dev at test phase	Development Environment	Not reported
• Interoperability	Pairs (series of programs	One or a pair of developers	Dev at test phase	Development Environment	Not reported
• Integration	Every program	System Tester	System test phase	Validation Environment	Reported at delivery review
• Stress Tests	Key input module I/O	System Tester	After I&T	Ops environment	Prior to delivery to ops

Purpose	Test Subject	Responsibility	When	Where	Results
• End-to-end	Every program chain	System Tester	System test phase	Validation Environment	Reported at delivery review

Table 6.7:1 Testing Approach

Table 6.7:2 indicates the type of integration and test, responsible parties and frequency.

Type of I&T	Responsibility	Frequency
Local-components of the task	Developer	Minor Milestones to verify development is progressing as expected
Intermediate	Development Lead	Major Milestones
Project	System Tester	Project Builds

Table 6.7:2 Applicable Integration and Test Levels

The Development Lead will be responsible for integrating the development components and doing the local testing. Test results will be informally communicated to the team members. If any major problems are found, Anomaly Reports will be generated and will follow Section 7.7.

6.8 Software Maintenance Plan

PDS will follow the Section 398 Software Maintenance Procedure for all maintenance work.

6.9 Software Retirement Plan

Prior to decommissioning all or part of the Project software set, stakeholders will be consulted and a retirement plan shall be developed and documented.

The software retirement plan will include:

- Analysis of the retirement requirements;
- Results from interaction with all relevant stakeholders;
- Determination of the impact of retiring the software;
- Identification of the replacement software and associated training, if applicable;
- A schedule and definition of responsibility for retiring the software;
- Identification of the responsibility for future support, if applicable;
- An archival process that includes retention of the software and associated documentation for a specified period. This includes all code, Operating System information needed to resurrect the program at a later date if necessary.

A review of the retirement plan will be conducted with relevant stakeholders.

The retirement plan will be approved by the Project Manager, with concurrence by all relevant stakeholders.

6.10 Product Acceptance Plan

The final step in testing is the product acceptance test. System Tester will develop the acceptance test plan that will include the following:

- Requirements to be tested
- Test Cases mapped to the requirements
- Acceptance Criteria
- Test Result
- Written concurrence/acceptance by the customer

The Acceptance Plan is stored in DocuShare. The developers and primary stakeholders will participate in the product acceptance testing. The stakeholders will assess how well the products meet the needs of the stakeholders as expressed in the acceptance criteria. Results of the Acceptance tests will be presented at the delivery to operations review or equivalent.

6.11 Software Delivery Plan

Each major formal delivery, whether internal or external will include the following items:

- Source code, Executable or compiled code, Test code, libraries, and scripts, from which all debugging and testing statements, instrumentation, and seeded defects have been removed or disabled;
- Configuration Files (startup files, tables);
- Database(s) or data files;
- Resource files (e.g. icons, graphics, sound, video);
- Requirement and design information; including descriptions of new or changed functionality;
- COTS software, government furnished software, freeware, and open source tools needed to construct and operate the software
- Make files, scripts, build and load instructions, and other tools supporting building, test, installation, and operation;
- Description of both development and operational environments and settings (e.g.,environment variables, switch and jumper settings, command line arguments, directory structures, access control parameters, system generation parameters, etc.);
- User's guides, operator manuals and help files;
- Test reports and other verification results
- Documentation of known problems, liens, or defects with workarounds;
- Points of contact, problem reporting protocols, and other end-user support;
- Licenses and copyright documents;
- Release Description Document

The delivered software will be installed in the targeted operational environments. Procedures for installing the PDS software in the Engineering Node environment as well

as the Discipline Node environment are documented in the Release Description Document (RDD) for each release.

6.12 Training

6.12.1 Role Based Training

PDS follows the suggested training as documented in the JPL Software Web Site. This is followed through by the PDS member's group supervisor and is reviewed in the annual performance review process. Table 6.12:1 summarizes the suggested training for the different PDS members based on their roles.

6.12.2 Technical Training

In addition to the role-based training, members of different areas may need to be trained in different technical areas. Table 6.12:2 summarizes the expected training for each functional area if applicable. Required training will be budgeted into the schedule. Project management will periodically verify that these courses have been taken. Table 6.12:2 shows actual technical training for this project.

Suggested Training \ Roles	Project Manager	SE Lead	Dev Lead	Developer	Ops/CM Lead	Tester	CM Engineer	System Administrator
S/W Mgmt & Planning	*X*	*X*	*X*		*X*			
Software Cost Estimation	*X*	*X*	*X*		*X*			
Managing S/W with Metrics	*X*	*X*	*X*		*X*			
Risk Management	*X*	*X*	*X*		*X*			
Requirements Management	*X*	*X*	*X*					
Applying the SDR	*X*	*X*	*X*		*X*			
Software Product Engineering		*X*	*X*	*X*	*X*	*X*	*X*	
Software Peer Reviews		*X*	*X*	*X*	*X*	*X*	*X*	
Software Testing			*X*		*X*	*X*		
Software Design		*X*	*X*	*X*	*X*	*X*		
Software Configuration Management			*X*		*X*		*X*	
Product Integration			*X*		*X*	*X*	*X*	
CM Tool		*X*	*X*	*X*	*X*	*X*	*X*	
Requirements Management Tool	*X*	*X*	*X*		*X*			
IT Security								*X*

Table 6.12:1 Role Based JPL Required Training

Suggested Training \ Functional Area	***System/Tool Development***	***Web Development***	***I&T***	***CM***	***SE***	***SA***
System Tools and Utilities	*X*	*X*	*X*	*X*	*X*	*X*
C	*X*					
Python					*X*	
Java	*X*	*X*			*X*	
Java Servlet Pages	*X*	*X*				
HTML		*X*				
Cold Fusion		*X*				*X*
Relational Database Knowledge	*X*				*X*	*X*
Protegé					*X*	
LDAP Authentication	*X*					*X*
Networks and Protocols						*X*
Data Storage Systems						*X*

Table 6.12:2 Project Specific Technical Training

6.12.3 End-User Training

The development team (Development Lead and developers) provide training on new capabilities to the end-users through demonstrations, user-guides and one-on-one training.

6.12.4 Training Records

All training records are kept through the Learning Management System (LMS). On the job training is an acceptable form of training. This training does not need to be tracked in the training records.

7 Supporting Process Plans

7.1 Configuration Management Plan

Configuration management will be applied with differing levels of formality depending upon the software development phase. There are three levels of SCM that will be applied with the general criteria defined in Table 7.1:1.

Software Configuration Management (SCM) Levels		
Informal -- Personal Level SCM	**More Formal -- Group Level SCM**	**Formal -- Project Level SCM**
• Development environment; • Maximum development flexibility; • Prior to first successful module test; • Document informally (e.g., in file header); • Retain all assembled / compiled versions; • Development code in CM tool (Subversion) to allow for check-in check-out with checked-in code residing on a separate computer • Periodically back up personal development environment. • Anomalies tracked by individual.	• Test Environment; • Allow concurrent development; • Reduce the number of users finding identical problems; • Facilitate reuse; • Units available for peer review, I&T, V&V; • Local problem reporting system in place, users alerted, solutions coordinated.	• Operations environment; • Provided for external SW distribution and use; • Internal and external users; available for partners, contractors, suppliers, verification / validation team, end users; • Formal change and impact process, prioritize maintenance effort; • Object image validated against test version; • Institutional problem reporting system used.

Table 7.1:1 Software Configuration Management Levels

The items to be stored in the CM system are identified in Table 7.1:2.

Type	CM Level	Item	Repository
Environment	Personal Group Project	Development Environment Test Environment Operations Environment	Wiki DocuShare DocuShare
Source Code	Project	System and Tool Software Web Sites	Subversion
Software Documentation	Project	System and Tool Software	Subversion
Project Documentation	Project	Requirements Design	DocuShare DocuShare
Build Procedures	Project	Delivery/Deployment	Subversion
Test Cases	Project	Unit Tests Plan, Procedures, Reports	Subversion DocuShare
Change Requests Problem Reports	Project	Issues	JIRA

Table 7.1:2 Configuration Items

The Configuration Items Table will be updated to reflect new configuration items as they are added to the task. The high-level PDS CM Plan is as follows:

1. Select and setup CM tools for requirements, code, documentation, and test environment
2. Control items during development (see Section 7.1) using the CM tools and procedures.
3. Create baselines for official releases
4. Track and control changes to code and other products (requirements, designs, release information, etc.)

The procedures related to the SCM levels described above are as follows:

Personal and Group Level SCM Procedure

1. Development Lead specifies requirements for software modifications or additions. This information is either captured in the Project Requirements document or as a change request or problem report in the JIRA issue tracking system. New versions of project-level documents are submitted to the PDS DocuShare repository.
2. Developer identifies the affected source code and retrieves it from the Subversion repository.
3. Developer makes the requested modifications or additions according to the specified requirements.
4. Developer performs unit and module testing. Developer testing normally occurs in the developer's environment. Depending on the scope of the modification or addition, testing may occur in the test environment in order to provide sufficient test coverage.
5. Developer submits source code modifications and additions to the Subversion repository. A JIRA issue identifier should be included in the submission comment, if appropriate.
6. Developer updates the associated JIRA issue, if appropriate.
7. Development Lead reviews submissions to the Subversion repository and comments on JIRA issues where appropriate.
8. Development Lead deploys the latest software modules from the Subversion repository to the test environment, following deployment procedures that are also captured in the Subversion repository, and then performs integration and test activities in the this environment.
9. Development Lead verifies new or modified functionality and updates associated JIRA issues where appropriate. Non-resolved or newly discovered issues take the process back to step 1.

Project Level SCM Procedure

1. Development Lead tags the latest release in the Subversion repository and builds the software modules following the delivery procedures that are also captured in the Subversion repository.

2. Tester deploys the delivered software modules in the test environment, following the deployment procedures.
3. Tester verifies new or modified functionality following the test plan and procedure. New issues are captured in JIRA issue tracking system. Non-resolved or newly discovered issues take the process back to step 1.
4. Operator/SA deploys the delivered software modules in the operations environment, following the deployment procedures.

Configuration Audits will be provided as described in Section 7.6, Reviews and Audits. The audit information will be automatically generated by the CM tool and will reflect current code base and changes to code baseline. Results will be stored in the CM system and will be available for inspection if needed. Anomalies found with the CM process/plan will be captured in the JIRA issue tracking system to be resolved by the System Engineer.

7.2 Verification and Validation Plan

To ensure quality and customer satisfaction, PDS shall perform Product Verification and Validation as indicated in Table 7.2:1. All verification and validation activities include:

- Procedures
- Reported results
- Issues in the appropriate form of
  - Action items
  - Change requests or
  - Anomaly reports
  - Analysis of the findings or results

Informal verification and validation activities occur automatically through Integration and Test procedures (see Section 6.7). Formal verification and validation activities also appear on the PDS Schedule. Table 7.2:1 is a summary of the verification and validation activities. Where more in depth information exists, this table refers to the appropriate Section in this document or the appropriate external document. This table also indicates the types of artifacts associated with each activity. Because this is a relatively mature system, all environments have been developed, tested and used so no further environments need to be established. In most cases, acceptance criteria will be set by the group performing the validation and this will be documented in the meeting minutes for the validation. For formal Integration and Test, successful execution of the test suite constitutes the acceptance criteria. The results are documented in the "I&T Report" (See Section 7.7).

Product	Ver	Val	Ref	V&V Level	Responsible Party	Method	Environment	Success Criteria	Frequency	Artifacts
Requirements	X	X	*Sec 3.1*	*Formal*	*Project Mgr* *System Engineer*	*Inspection* *Peer Review*	*None*	*See Section 5.1*	*Annually at beginning of PPBE cycle*	*Reqs in DocuShare; Inspection minutes; Action items; Analysis*
Design		X	*Sec 3.2*	*Informal*	*Dev Lead* *System Engineer*	*Inspection* *Peer Review*	*None*	*See Section 6.4*	*Upon design change*	*Design; Peer Review notes; Action items;*
Application Interface	X	X	*Sec 3.3*	*Informal*	*Dev Lead*	*Peer Review* *Demonstration*	*Test system;* *Ops system*	*Interface meets requirement; interface functions as intended in production system*	*Upon design of new interface; upon change to existing interface*	*Code in CM; Peer Review notes; Action items; Test plans; Test results; Test analysis*
Code	X			*Informal*	*Dev Lead* *Developer*	*Peer Review*	*None*	*Code meets quality standards and coding standards*	*When segment of code reaches maturity as determined by the Dev CDE*	*Code in development; Peer Review notes; Action items; Analysis*
User Guide		X		*Informal*	*Dev Lead*	*Actual Use*	*None*	*UG meets requirements and enables operations to perform successfully*	*Prior to operational delivery*	*UG in CM; Review feedback from operations; Action items*
Test Environment		X		*Formal*	*Dev Lead* *Ops Lead*	*Actual Use*	*Test System*	*All regression tests complete successfully*	*Prior to Testing*	*Test Environment validation report*
Operations Environment	X	X		*Informal*	*Tester* *Sys Admin*	*Regression Tests*	*Ops System*	*All regression tests complete successfully*	*Upon production equipment change*	*n/a*
CM Audit	X			*Formal*	*SQA Engineer*	*Objective Evaluation Checklist*	*Ops System*	*Sec 2.6.4*	*Annually*	*Audit Trail*

Table 7.2:1 Verification and Validation Plans

7.3 Trade Studies and Technical Decisions

All trade studies will be conducted using the Section 398 Decision and Analysis Local procedure located in the Section 398 PAL. The template for documenting the trade stud is in **Error! Reference source not found.**. A trade study will be implemented when the decision results in a major change in schedule, budget or priorities. A major change is quantified at 50K or more in cost or a schedule slip that will impact the ability to deliver on time for the project or a decision that will affect multiple developers with regards to design or development schedule.

7.4 Documentation Plan

See Section 1.6, Documents and Records for the Documentation Plan and related information.

7.5 Quality Assurance Plan

The Quality Assurance Plan for PDS4 is described in the PDS4 Project Plan section 10.

7.6 Reviews and Audits

PDS4 will provide or participate in the reviews scheduled in the PDS4 Project Plan section 14.2.2. PDS will provide or participate in the reviews in the following table. Reviews will follow the guidance in the Technical Status Reviews DocID 35493 and the Reviews Handbook on the Software Website. There are several helpful checklists in the JPL software site that can be used to prepare for milestone reviews.

Review	Frequency	Chair	Stakeholders	Appr
Work Implementation Plan Review or SMP	*During annual PPBE cycle*	*Project Manager*	• *WBS Leads* • *Line Management*	• *Lin* *Ma*
Cost Estimate Review	*As Costs Change*	*Project Manager*	• *Program Management* • *Line Management*	• *Pro* *Ma* • *Lin* *Ma*
Functional Requirements Document Review	*During annual PPBE cycle, or as needed*	*Project Manager*	• *Dev Lead* • *SE Lead* • *Ops Lead*	• *Pro* *Ma* • *Ma* *Co*
Software Requirements Review	*As needed*	*Dev Lead*	• *Dev Lead* • *SE Lead* • *Ops Lead*	• *Pro* *Ma*
Design Review	*As planned in the Project Schedule*	*Program Manager*	• *Data Providers* • *Data Consumers*	• *Bo* *Ma* *Co*

Review	Frequency	Chair	Stakeholders	Approvers
Operational Readiness Review	*Prior to major milestone delivery*	*Program Manager*	• *Data Providers* • *Data Consumers*	• *Board or Management Council*
Peer Reviews	*As needed*	*Dev Lead*	• *Dev Lead* • *SE Lead* • *Ops Lead* • *Developers*	• *N/A*
Configuration Audits	*Annual*	*Ops Lead*	• *Dev Lead* • *SE Lead* • *Ops Lead*	• *N/A*
Inheritance Reviews	*As needed*	*Dev Lead*	• *Dev Lead* • *SE Lead* • *Ops Lead*	• *Project Manager*
PPQA Audits	*Annual as described in the PPQA Plan*	*Line Management*	• *Line Management*	• *N/A*

Table 7.6:1 Applicable Reviews for this Task

7.7 Peer Reviews

All peer reviews indicated in this document will follow the Section 398 Local Procedure for Peer Reviews in the Section 398 PAL. The software manager will determine the formality of the peer reviews based on risk. Results from formal Peer Reviews will be captured in the Peer Review Results Memo in **Error! Reference source not found.**. Items identified will be tracked to closure as specified in Section 7.8. Table 7.7:1 lists the products that will be peer reviewed for this task:

Products Required to Be Peer Reviewed				
Product	**Method**	**Critical Units Only?**	**Peer Review Leader**	**Comment**
Software Management Plan (SMP)	Walkthrough		Line Management	Involve stakeholders to establish buy-in
Cost Estimate	Walkthrough		Project Manager	
Requirements	Walkthrough		SE Lead	
Software Inheritance	Presentation		Dev Lead	Prior to PMSR

Products Required to Be Peer Reviewed				
Product	**Method**	**Critical Units Only?**	**Peer Review Leader**	**Comment**
Software Requirements (SRD)	Presentation		Dev Lead	Prior to CDR
Architectural design (SDD-I)	Presentation		Dev Lead	Prior to CDR
Detailed design (SDD-II)	Presentation	**X**	Dev Lead	Prior to CDR
Code	Walkthrough	**X**	Developer	Prior to Delivery
Unit Tests and Test Results	Inspection	**X**	Dev Team	Prior to Delivery
Integration and Test Plan (SITP-I)	Presentation		Ops Lead	Prior to CDR
Integration and Test Procedures (SITP-II)	Presentation		Ops Lead	Prior to CDR and ORR
User's Guide	Walkthrough		Dev Lead	Prior to DR
Maintenance and Retirement Plan	Presentation		Dev lead	Prior to start of Maintenance Phase

Table 7.7:1 Products to be Peer Reviewed

7.7.1 Mandatory Review Topics

Table 7.7:2 lists the topics that will be covered, at a minimum for each Milestone Review at various phases of the PDS.

Topic	**PMSR**	**IR**	**CDR**	**ORR**	**DR**
a. Definition and adequacy of customer and user requirements.	*X*		*X*		
b. Commitment to a proposal or work package.	*X*	*X*	*X*	*X*	
c. Definition and adequacy of software plans.	*X*	*X*	*X*	*X*	
d. Inheritance of legacy code, reusable components, and Off-The-Shelf products including test environment - emphasis on risk and effort.		*X*	*X*		
e. Technology readiness.	*X*				
f. Architectural design, addressing interfaces and interactions among modules.			*X*		
g. Definition and adequacy of software requirements and design.			*X*	*X*	
h. Test approach and test plan, including design of testbeds, simulators, and models.			*X*	*X*	
i. Test readiness.				*X*	
j. Test results.					*X*
k. Functional verification/validation or pre-acceptance test.			*X*	*X*	*X*

Topic	PMSR	IR	CDR	ORR	DR
l. Requirements and assumptions for software delivery and maintenance.				*X*	*X*
m. Implications for Retirement			*X*		*X*

Table 7.7:2 Milestone Review Topics by Project Phase

7.8 Problem Resolution Plan

PDS will follow the Change Control process in Section 5.10. All problems and issues are maintained in the appropriate repository and are tracked to closure and reported at the Monthly Management Reviews (MMR) and summarized at milestone reviews. Appropriate level of Configuration Management is achieved by submitting the artifacts to the repository specified in Section 2.2. The owner of the corrective actions, as specified in the Corrective Actions Table 7.8:1, analyzes the issues and corrective actions to determine the root cause by informal methods. He or she then makes appropriate changes to reduce the number of corrective actions over time and tracks the improvement.

Issue Type	Name	Tool	Responsible Party	Reporting Mechanism
Process Issues	*RFA List*	*Spreadsheet*	*Dev Lead* *SE Lead* *Ops Lead*	*Informal email feedback*
Requirements	*RFA List*	*Spreadsheet*	*Dev Lead* *SE Lead*	*Informal email feedback and peer review*
Design	*RFA List*	*Spreadsheet*	*Dev Lead* *SE Lead*	*Informal email feedback and peer review*
Software Anomalies during I&T	*Anomaly Reports*	*JIRA*	*Ops Team*	*Feedback captured in JIRA and test reports*
Software Anomalies during Operations	*Anomaly Reports*	*JIRA*	*Ops Team*	*Feedback captured in JIRA and RDD*
Peer Review Findings	*Issues List*	*JIRA*	*Dev Lead*	*Informal email feedback*
Infrastructure	*Issues List*	*JIRA*	*Ops Lead*	*Informal email feedback*

Table 7.8:1 Corrective Actions and Reporting

7.9 Subcontractor Management Plan

There is no subcontractor management required for this task.

7.10 Process Improvement Plan

The Section 398 Process Improvement Plan is in the Process Asset Library and is found at link. This task will follow the process improvement plan and contribute data and

assets as required. The Process and Product Quality Assurance process will be used as one way of collecting process improvement information.

PDS will implement a two-tiered process improvement plan. As processes are executed, their effectiveness will be evaluated for accuracy and efficacy. When problems are found, the problem will be

a. Opened as a Change Request if the problem affects a majority of the members of the team. The CR will be entered into the CR system and tracked to closure.
b. Documented in the TBS. It will be the responsibility of the TBS to ensure the problem is resolved and documented.

When processes apparently work, but are not efficient, the affected group will analyze the process and make appropriate changes to improve efficiency. PDS will provide process assets and information to the Section PAL as required by the Section Process Improvement Plan.

8 Appendices

8.1 Appendix A: Acronyms

IBT	Institutional Budgeting Tool
PDS	Planetary Data System
PDS3	PDS Version 3
PDS4	PDS Version 4
PDS MC	PDS Management Council
PPBE	Planning Programming Budget Execution
SDWG	System Design Working Group
WA	Work Agreement

8.2 Appendix B: Definitions

N/A

8.3 Appendix C: Coding Standards

Coding standards are referenced in Section 6.6.2.

8.4 Appendix D: Verification Levels and Methods

Verification levels

Requirements verification is performed at three levels:

1. **Component** - Verification of requirements and capabilities of individual applications and libraries, including interfaces between individual components. Verification of component requirements may imply verification of parent subsystem requirements.
2. **Subsystem** - Verification of requirements and capabilities of the PDS as a whole, including end-to-end processing capabilities. Subsystem-level verification includes verification of PDS internal interfaces and interfaces with non-PDS entities. It does not include interfaces with other portions of the PDS ground segment.
3. **Ground segment** - Verification of interfaces with other portions of the PDS ground segment.

Verification Methods

Five methods for verifying requirements are used.

1. **Similarity** - Similarity is employed where the design of hardware or software is identical to or sufficiently similar to proven hardware or software such that further verification is unnecessary.

2. **Inspection** - Inspection is the visual examination or non-destructive measurement of product characteristics or the review of design, production, or test documentation to determine compliance with the specification requirements.

   During each build cycle for a component, the primary developer of that component calls two reviews: an objectives review and a delivery review. In addition, each developer occasionally calls a code review. These reviews provide insight into the development process. At these reviews, peers and stakeholders help the developer verify that the software will meet the needs of its stakeholders.

   - **Objectives review:** An objectives review ensures that all stakeholders agree on the list of features included in the development cycle. The developer calls this review once he understands that list. He presents these requirements to the GDS Systems Engineer, the GDS Manager, and any Science Team member with a stake in the component.

   - **Delivery review:** The developer summarizes the tests she performed on the software she plans to deliver. The GDS Systems Engineer, the

GDS Manager, the GDS Configuration Manager, and any other stakeholder determine whether those tests validate the software for use in the current build.

The System Tester must also create a delivery document after he completes the system testing. He presents the test results to the GDS Manager and the GDS Configuration Manager. The attendees examine the report and ask any questions that need to be answered before the final build is made.

- **Code review:** A code review helps a developer reduce the number of defects in his code. During this review, members of the GDS Development Team analyze a module, subroutine, or object coded by one of the team members. They check the code both that the developer follows the team's coding standards and that the logic of the code is correct.

  This type of review is not tied to the build cycle directly, as are the other types of reviews. Each developer holds at least one code review per build. He chooses which code to review. Code selected for review is sufficiently complex to benefit from being double-checked.

3. **Analysis** - Analysis is an engineering assessment or mathematical verification method that uses techniques and tools such as math models, computer models or codes, prior test data, statistical data, or analytical assessments to confirm compliance with specification requirements with appropriate design margin. Analysis is mainly used where test methods are not practical or feasible.

   Analysis also verifies a requirement that serves as a parent to one or more lower-level requirements. Verification of all of the child requirements implies verification of the parent requirement. In some cases, this analysis partially verifies a requirement but additional verification methods are needed to verify it fully. In that case, multiple verification methods are specified for the requirement.

4. **Test** - Test is a quantitative verification method used to verify conformance of functional or performance characteristics with specific requirements. Tests normally require the use of special test equipment to measure and verify performance. Analysis of the test results is included as part of the test.

   Types of tests include:

   - **Functional** - Verifies the functionality and compatibility of the test article.

- **Stress Test** - System continues to operate, even at reduced performance levels, when subjected to high boundary input and throughput loads.

- **Performance** - Verifies that the test article operates according to specification by measurement of performance levels.

- **Acceptance** - Verifies that the product meets predetermined criteria established by the primary customer of the product.

- **End-to-end Information System (EEIS)** - Verifies interfaces across the mission. These tests are joint Flight System (FS) and Mission Operations System (MOS) tests.

- **Mission Operations System** - Verifies functions and interfaces among MOS elements prior to integration with the flight system.

- **Mission Scenario** - Verifies the compatibility and functionality of the Flight Segment and MOS hardware and software in flight-like conditions.

- **Operations Readiness Test (ORT)** - Verifies the readiness of the MOS (h/w, s/w, people, and procedures) to support the mission timeline; primarily a MOS test

- **Integrated Operations Test (IOT)** - Verifies MOS hardware, software, people, and procedures without necessarily following the mission timeline.

5. Demonstration