Planetary Data System

PDS 2010

System Architecture Specification

February 28, 2010
Version 1.0

Jet Propulsion Laboratory
Pasadena, California

CHANGE LOG

Revision	Date	Description	Author
0.1	2009-11-10	Initial draft based on information collected by the System Architecture Working Group.	SAWG
0.2	2008-11-20	Added requirements tracing, descriptions for architectural elements and service provisioning.	S. Hardman
0.3	2009-01-08	Updated the service definitions.	S. Hardman
0.4	2009-10-26	Updated the service diagrams and corresponding definitions along with the scenarios. Also added some content to the Technology Architecture section.	S. Hardman
0.5	2010-02-04	Cleaned up some formatting and requirements tracing. Incorporated comments from the SDWG. Added content for the Data Architecture section.	S. Hardman, R. Joyner
0.6	2010-02-14	Added other component definitions and updated the EN-based distribution scenario.	S. Hardman
1.0	2010-02-28	Incorporated more comments and made an effort to strike the use of passive voice.	S. Hardman

Revisions 0.1 through 0.3 were under the purview of the System Architecture Working Group (SAWG). This working group consisted of the following members:

D. Crichton, S. Hardman, T. King, S. LaVoie, M. Martin, T. Stein

Revisions 0.4 and on are under the purview of the System Design Working Group (SDWG). This working group consists of the following members:

S. Hardman, T. King, M. Martin, P. Ramirez, A. Stanboli, T. Stein

TABLE OF CONTENTS

1.0 INTRODUCTION

The PDS 2010 project will overhaul the PDS data architecture (e.g., data model, data structures, data dictionary, etc.) and deploy a software system (online data services, distributed data catalog, etc.) that fully embraces the PDS federation as an integrated system while leveraging modern information technology. The data architecture portion of the project is an effort to develop PDS4, meaning version 4.0 of the PDS standards. This effort and the effort to design, implement and deploy software services, constitutes the PDS 2010 project.

1.1 Document Scope and Purpose

The purpose of this document is to convey the system architecture for the PDS 2010 system in a manner that is understandable to the broad spectrum of PDS stakeholders but is intended for the designers and developers of the PDS 2010 system. The scope of the document is limited to the architecture of the PDS system but will delve into the system design where appropriate.

1.2 Method

This document follows an architectural framework, detailed in section 3 of this document, to document the system architecture for PDS 2010.

1.3 Controlling Documents

[1] Planetary Data System Strategic Roadmap 2006 - 2016, PDS Management Council, February 2006.

[2] Planetary Data System (PDS) Level 1, 2 and 3 Requirements, August 2006.

[14] Planetary Data System (PDS) 2010 Project Plan, February 2010.

1.4 Applicable Documents

The list of applicable documents are in Appendix A.

1.5 Document Maintenance

The system architecture will evolve over time and this document should reflect that evolution. This document is under configuration control with any modifications submitted to the Management Council for approval.

2.0 BACKGROUND

The main goal of this effort is to define the over-arching System Architecture for PDS, which will encompass all PDS 2010 and future projects. This includes projects developed at the Engineering Nodes as well as the Discipline Nodes. The following “architecture” definitions are for reference purposes:

- Enterprise Architecture (applies to NASA)
  Simply stated, enterprise architectures are “blueprints” for systematically and completely defining an organization’s current (baseline) or desired (target) environment. [11]
- System Architecture (applies to PDS system as a whole)
  A formal description of a system, or a detailed plan of the system at component level to guide its implementation. [5]
  The structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time. [5]
- Software Architecture (applies to PDS software components)
  The two main aspects of software architecture are that it provides a design plan (a blueprint) of a system, and that it is an abstraction to help manage the complexity of a system. [12]
- Service Oriented Architecture (specific approach to software)
  A software architecture for building applications that implement business processes or services using a set of loosely coupled black-box components orchestrated to deliver a well-defined level of service. [7]

As defined above, this document focuses on system architecture.

3.0 FRAMEWORK

The decision early on was to follow an industry standard approach or framework to guide the system architecture development effort. The working group selected the Open Group Architecture Framework (TOGAF) [5] as the framework for developing the PDS 2010 system architecture. TOGAF is essentially a tool for assisting in the acceptance, production use and maintenance of architectures. It is very flexible with respect to the tailoring to an organization's unique needs and PDS has many of those.

TOGAF is also very adaptable with regard to incorporating other frameworks and standards. The following were utilized or are under consideration as the architecture progresses:

- Zachman Framework [6]
  Utilized for artifact identification and organization.
- IEEE 1471-2000 [9]
  Utilized for architectural description guidelines.
- Reference Model for Open Distributed Processing (RM-ODP) [10]
  Under consideration for the software architecture.

TOGAF's core consists of their Architecture Development Method (ADM), which is a systematic approach for developing and using an enterprise architecture. The ADM consists of the following phases:

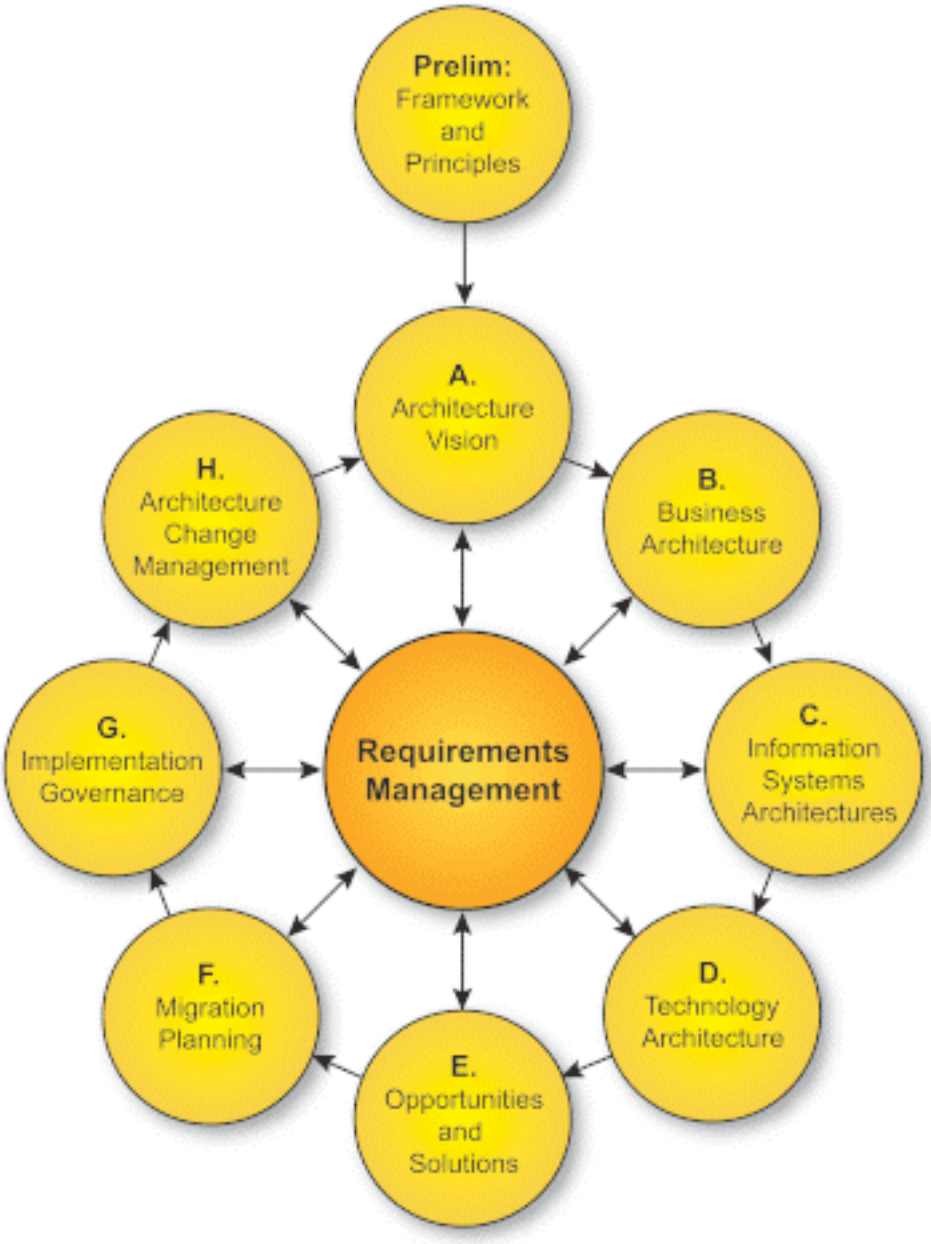

Figure 3.1: TOGAF's Architecture Development Method

Each of the above phases has a prescribed set of inputs and outputs (artifacts). Along with the ADM, TOGAF also consists of the Enterprise Continuum and the Resource Base. The Enterprise Continuum is a virtual repository of architecture assets. The Resource Base is a set of tools and techniques for applying TOGAF.

Development of the sections and artifacts that follow were the result of progressing through the phases prescribed by TOGAF.

4.0 ARCHITECTURAL DRIVERS

Architectural Drivers for PDS 2010 come from several sources: the PDS Roadmap [1], the PDS Requirements [2], and input from the PDS Management Council in the form of questions and problems that PDS 2010 should address.

The PDS Roadmap [1] provides a number of drivers for the next ten years that directly affect the architectural choices that PDS will need to make in all four architectural areas (new and updated processes, changes to the data architecture, new functional capabilities for the application architecture and adoption of new technologies as part of the technology architecture).

The PDS Level 1, 2, 3 Requirements [2] were developed by the PDS Management Council. They provide a broad set of system level requirements for PDS and will guide all subsystem development. In the November 2007 Management Council meeting, a request was given to each Discipline Node to identify a set of questions for response by the PDS4 Working Groups. The PDS4 Architecture Working Group report [3] addressed each question.

Based on the available input from the roadmap, requirements and the Management Council, the working group extracted and created a categorized list of architectural drivers organized into thematic areas [4]. The summary of this list is as follows:

- **More Data**
  PDS storage requirements are projected to increase from 40 TB to over 500 TB in just three years. This will require more automation, scalable high capacity storage systems and advanced data movement techniques.

- **More Complexity**
  Missions, instruments, and data are all becoming more complex. This will require an improved information model for archiving diverse products (in situ, geographical, astronomical) as well as a modern online data dictionary with name space management and access control.

- **More Producer Interfaces**
  PDS is facing an increasing number of missions, a greater number and diversity of data providers, and smaller, focused missions. This will require a streamlined standards architecture that is easy to learn and use, with more reliance on delivering data in standard data formats. Cross-platform archiving tools must be provided which can be used to design, generate, validate, and deliver archival data sets.

- **Greater User Expectations**
  The World Wide Web has led users to expect well-documented data to be readily available via text-based or graphical search systems with data delivery in a variety of formats compatible with their data processing systems. This includes access to tools for displaying or analyzing discipline-specific data as well as special processing to produce higher order products.

- **Limited Funding**
  The emphasis on smaller, faster, cheaper missions that often include international partners may limit the ability to provide products suitable for analysis by the broader science community. This puts a burden on NASA Data Analysis programs or on the PDS to have to finish the job. As space exploration continues to become an international effort, PDS must expend increasing resources working with foreign agencies and international organizations to assure access to new mission data. The “internationalization” of space exploration will also necessitate additional standards that promote data sharing and interoperability and an international core data model for archiving and for querying remote archives.

- **Creating a "System" from the Federation**
  The current PDS nodes operate autonomously and independently with limited distributed access via PDS-D to node repositories. This means that each site must do its own planning, design, review, procurement, code development, testing and operations. There is little sharing of technical expertise in this heterogeneous environment. A better approach would be to provide technology specifications to allow distributed and shared services across the federation, and to ensure that tools can plug into local environments. Common infrastructure services would be provided where it makes sense (physical media production, security, backup, mirroring, web site maintenance).

5.0 ARCHITECTURAL PRINCIPLES

Architectural principles can form a general basis for decision making of architectural choices for a system. Large-scale software systems typically have explicit principles that guide the evolution of components and the systems for large enterprise teams [5]. The principles detailed below are based on the architectural drivers and the initial set of principles identified by the PDS4 Architecture Working Group [3], along with influences from the TOGAF example principles.

The principles will drive the design and development of PDS 2010. Once the system is developed and operational, the principles will guide future decisions pertaining to upgrades and new technology choices. The principles are organized according to their most pertinent aspect of the system architecture (e.g., process/core, data, application or technology). Descriptions for each of the principles include the following:

- **Statement** - Brief description of the principle.
- **Rationale** - Describes why the principle is important and any relationships to other principles.
- **Implications** - Lists any requirements or impacts this principle will have on the resulting architecture and system.

5.1 Process/Core Principles

The process/core principles are as follows:

Data Stewardship

Statement: PDS will manage NASA-related planetary data in order to maintain its usability, accessibility, integrity and quality.

Rationale: This principle embodies the PDS mission: "To facilitate achievement of NASA's planetary science goals by efficiently archiving and making accessible digital data produced by or relevant to NASA's planetary missions, research programs, and data analysis programs." [1] Although stewardship does promote accessibility, we have assigned data accessibility its own principle under the Data Principles section to cover this aspect of the mission.

Implications:
- PDS will collaborate with data providers as early as possible in the data creation process to ensure adoption and effective use of PDS Standards and tools.

- A key implication of data stewardship is the preservation of that data for the long-term. For the purposes of PDS, long-term is measured in terms of at least 10s of years.
- Preservation and integrity of the data must be maintained. In order to accomplish this, the PDS must adhere to a rigorous data integrity policy to ensure its data are reliable.
- Manage data in a way that preserves its meaning and promotes its understanding. This implies that software is available to read and transform the data for use in current day environments.
- Release data to the public in accordance with PDS policies.

Reliability

Statement: Maintain PDS operations in spite of system interruptions.

Rationale: As the storage and distribution of PDS data becomes more dependent on the PDS computing environment, we must consider the reliability of such systems throughout their design and use. Nominal operations should recover quickly in the event of hardware failures, natural disasters and data corruption.

Implications:
- Recoverability, redundancy and maintainability addressed at the time of system design.
- Specific applications assessed for criticality and impact on the PDS mission, in order to determine the level of continuity required and the necessary corresponding recovery plan.

Common Use Software

Statement: Development and integration of system-wide software (applications and APIs) is preferred over localized development of duplicative functionality.

Rationale: The reality of PDS, and the planetary science community as a whole, is that the data and the expertise to utilize that data are distributed across multiple disciplines. This distribution facilitates the separation of data and personnel along discipline boundaries. Development of common software and/or integration of off-the-shelf software will allow PDS to maximize the benefit of the software development resources as well as minimize the development of duplicative functionality across the nodes.

Implications:
- For PDS to function as a system, a number of common services must be developed and utilized by all nodes within the system. Common services might include security, search and data distribution services.

- The architecture must allow for changes in the distributed nature of the system including Discipline Node restructuring or Data Node dissolution resulting in data relocation and change of ownership.

5.2 Data Principles

The data principles are as follows:

Model Driven

Statement: The PDS will design and maintain a conceptual information model that is implementation independent.

Rationale: The need for an information model is directly supported by the "More Complexity" architectural driver as well as a need to simplify the current PDS Standards.

Implications:
- A formal data-modeling notation defines the information model.
- All data-related models derive from the conceptual information model.
- Software developed for the system evolves as the information model evolves.

Common Vocabulary and Data Definitions

Statement: Data are described consistently throughout the system, and the definitions are understandable and available to all users.

Rationale: A common vocabulary, based on a data model, is an essential component for generating and maintaining quality metadata. It also aids in the understanding of the data from the users perspective.

Implications:
- A data dictionary must be established and utilized uniformly throughout PDS.
- A change control board manages additions or modifications to the data dictionary.

Data are an Asset

Statement: PDS data are an asset that has value to NASA and the larger Planetary Science Community and are managed accordingly.

Rationale: Unlike a business where data are simply considered a resource, albeit a valuable resource, data are the cornerstone of the PDS mission. This principle relates to the Data Stewardship principle detailed above.

Implications:

- PDS will ensure that data ingested into PDS conforms to PDS Standards pertaining to metadata content, data format and data quality.
- PDS will ensure the safekeeping of its data holdings. In order to accomplish this, the PDS must maintain multiple copies of the data according to its policies and provide for disaster recovery.

Data are Accessible

Statement: Data are accessible for users to perform their functions, regardless of where the data are located.

Rationale: In the age of the Internet, everything is expected to be online and downloadable at the click of a mouse. The data holdings of PDS are no exception.

Implications:
- In order to enhance accessibility, the PDS must offer search interfaces for discovering data within the holdings.
- The PDS data holdings must be online and accessible via the Internet. This should include some accommodation for fail-over access to a secondary means for acquiring data if the primary means is unavailable.

Data are Usable

Statement: Data are usable for users to perform science analysis appropriate for their domain.

Rationale: In order to make PDS data usable, data are available to users in contemporary formats.

Implications:
- Data are available in or are easily converted to ready-to-use formats.
- Appropriate metadata is captured and made available to users in order to identify and analyze the usefulness of the data.

5.3 Application Principles

The application principles are as follows:

Modular Development

Statement: The system is decomposed into manageable elements and components in order to facilitate phased development and deployment.

Rationale: Given budgetary, mission and user constraints, it is critical that the PDS be able to evolve parts of the system over time. Separating the architecture

into elements and then components facilitates the evolution of the system while preserving other operational parts.

Implications:
- The system architecture captures the decomposition of the system.
- Components of the system will be designed to minimize inter-component dependencies.
- The development schedule prioritizes commonly used components.

Technology Independence

Statement: Software is independent of specific technology choices and therefore can operate on a variety of technology platforms.

Rationale: When software is independent of the underlying technology, it can be developed and maintained in a cost-effective manner. Certain dependencies are unavoidable, but obvious ones such as platform and operating system dependence are certainly achievable.

Implications:
- Design and development of software is in accordance with appropriate industry standards. For example, utilizing a standard such as eXtensible Markup Language (XML) offers a platform independent solution for constructing messages, logs, etc.
- Design and development of services utilizes middleware. A middleware layer can offer a generic software interface to one of those unavoidable technologies. Therefore, in the future, technology replacement only requires modification to the middleware software.
- Development of Application Programming Interfaces (APIs) will accommodate or provide a pathway for legacy applications to integrate with the new architecture.

Scalability

Statement: The system is scalable to accommodate increased data volume and complexity.

Rationale: Several of the architectural drivers point towards the need for an architecture and a system that is scalable (i.e., more missions producing increasingly more data that is more complex in nature).

Implications:
- Hardware solutions for data storage should allow for ease of expansion.
- Services for searching data sets and products should utilize query optimization methods for returning query results in a timely manner.

- Services for data distribution should offer transformation capabilities and utilize proven technologies for moving data across the Internet.

Ease-of-Use

Statement: Software is easy to use.

Rationale: Besides the obvious reasons for ease-of-use, one of the goals of PDS is to achieve early adoption of the PDS processes and tools with data providers. If the software is easy to use and well documented, it will require less of a resource impact on the data provider.

Implications:
- Develop software with intuitive interfaces.
- Applications and APIs are well documented not only for the PDS users but also for the PDS developers.

5.4 Technology Principles

The technology principles are as follows:

Requirements-Based Change

Statement: Perform modifications to the system (both software and technology) only in response to new or modified requirements approved by the Management Council.

Rationale: This principle will foster an atmosphere where the system changes in response to PDS needs and not the other way around. This is to ensure that the purpose of the system is to support the PDS mission.

Implications:
- Changes to the system will follow a software development process (i.e., Requirements, Design, Development and Deployment) with appropriate reviews.
- Manage system software under a change management process.

Interoperability

Statement: Software and hardware should conform to appropriate standards that promote interoperability for data, applications, and technology.

Rationale: Use of industry standards helps promote consistency and maintainability of services and their interfaces. It also opens up the possibility of adopting open source or vendor solutions. This type of standards conformance,

at least for the system's external interfaces, makes the system easier and more cost effective to interface with programmatically.

Implications:

- Industry standards should be adhered to where appropriate. These standards should pertain to capabilities and interfaces as opposed to specific products.
- Services and interfaces must be well defined to promote interoperability.

6.0 SCOPE

The following questions were presented at the Technical Session in September 2008 to help determine the scope of the PDS 2010 System Architecture:

- Will PDS 2010 impact how nodes currently conduct administrative tasks (e.g., budgeting, scheduling, calendars, status reporting)?
  Beyond providing functionality for capturing and reporting metrics, the architecture will not address functions like budgeting, scheduling calendars and status reporting.

- Will the PDS4 information model incorporate all of the PDS operational information (e.g., deliveries, reviews, orders, housekeeping, usage statistics)?
  The Data Architecture will focus on modeling observational data. Follow-on efforts can incorporate data related to deliveries, reporting, tracking, etc.

- Will PDS 2010 apply to sub-nodes or data nodes or might PDS 2010 be implemented just at the discipline nodes initially?
  Sub-nodes and data nodes are subject to the System Architecture. Provide service interfaces or wrappers to aide in interfacing or integration.

- Will PDS 2010 be enforced on existing but ongoing project interfaces or will existing projects be grandfathered?
  The consensus was that most existing projects did not need grandfathering, but consider each project on a case-by-case basis. The main issue here is whether a project is migrated or bridged to the new system.

With regard to the scope, the consensus at the Technical Session was that the PDS 2010 system should have a rigid core while allowing local extensions. This pertains to both application and data architecture.

7.0 VIEWPOINTS AND VIEWS

7.1 Viewpoints

The viewpoints, their associated stakeholders and concerns, for PDS 2010 are as follows:

Scope (Contextual)

Stakeholder: NASA Program
- This stakeholder represents the customer with respect to the fact that this is the source of project level requirements and funding.
- Ensures that the science community is getting the data and the support it needs from PDS.
- Obtain and provide sufficient funding to support the PDS mission.

Concerns:
- Does the architecture and design of the system satisfy project-level requirements?
- Is PDS ensuring the integrity and preservation of NASA-funded planetary data?
- Is the planetary science community able to access and utilize the data archived at PDS?

Project (Conceptual)

Stakeholder: Management Council
- This stakeholder represents the management level of the PDS.
- Responsible for prioritizing resources among discipline and support nodes.
- Works with NASA management to compete with other NASA disciplines for funding.
- Responsible for levying and approving requirements for the system.

Concerns:
- Are the requirements of the system, addressed by the architecture and design of the system?
- Does the system address the needs of the missions/data providers?
- How is migration to the new system accomplished?

System (Logical)

Stakeholder: System Engineer
- This stakeholder represents a portion of the PDS Technical group responsible for designing the PDS system.
- Responsible for developing lower-level requirements.

Concerns:
- Does the architecture and design of the system address the requirements levied on the system?
- Are the selected technologies sufficiently abstracted from the software to allow for future replacement?
- Are the selected standards appropriately applied according to their specifications?

Technology (Physical)

Stakeholder: Data Engineer / Software Engineer
- This stakeholder represents a portion of the PDS Technical group that interacts with the data and develops/maintains the software that comprises the PDS system.

Concerns:
- Are the services and their interfaces well defined?
- Is the software well documented and easy to use?

Detailed Representation (Definition)

Stakeholder: Data Provider
- This stakeholder represents the mission, instrument team and NASA-funded researcher who are involved with delivering data to the PDS.
- Needs to meet AO commitments with respect to archiving with minimum cost.
- May encounter serious budget pressure from overruns in other project phases.

Concerns:
- What is the PDS interface for submitting data to the archive?
- Does PDS offer documentation, software and standards for preparing data to archive?
- Can PDS handle the data volume and frequency for planned data submissions?

Operations (Instance)

Stakeholder: Data Consumer
- This stakeholder represents the Planetary Scientist, which includes those experienced with solar system exploration missions and those who are mission-naïve. They include graduate students.
- While the PDS is a NASA-funded program, many of its planetary missions are international partnerships and therefore the User Model must include non-US planetary scientists at some level.

Concerns:

- Can I find and retrieve the data I need to compliment my research?
- Does PDS offer a user-friendly interface for finding and retrieving this data?
- Can I transform data into a format that is useful for my research?
- Does PDS offer support and expertise for analyzing the data?

7.2 Views

In order to provide a succinct representation of the views for the PDS 2010 System Architecture, the planned views have been mapped to the Zachman Framework for Enterprise Architecture [6]. The Zachman Framework details six described or modeled aspects of the enterprise. These aspects are What, How, Where, Who When and Why. Because the PDS 2010 effort is focusing on system architecture, we have chosen to only represent three of the aspects (What, How and Where) in our diagram. Although the Why aspect is certainly relevant to the definition of a System Architecture, that aspect is covered by the Architectural Drivers and Architectural Principles sections in this document. The PDS/Zachman diagram follows:

	System Architecture Aspects			
Stakeholders	**What (Data View)**	**How (Functional View)**	**Where (Deployment View)**	**Viewpoints**
NASA Program	Scoping Model (Where does PDS fit in Space Science Disciplines?)	Context (Where PDS Fits in Mission Pipeline?)	Distributed Nodes	Scope (Contextual)
Management Council	Conceptual Map	Service Identification	Service Provisioning	Project (Conceptual)
System Engineer	Logical Model (Class Diagrams, PDS Specification)	Service Definitions	Distributed Service Architecture	System (Logical)
Data Engineer / Software Engineer	Data Dictionary / Standards Reference	Service Interfaces	Deployed Service	Technology (Physical)
Data Provider	ODL Templates / XML Schema	Active Services	Service Usage	Detailed Representation (Definition)
Data Consumer	Data Instance	Active Services	Service Usage	Operations (Instance)

Figure 7.1: PDS/Zachman Mapping

Although chapter 39 of TOGAF (ADM and the Zachman Framework) goes into detail regarding how TOGAF maps to the Zachman Framework, we take a much more simplified view. The beginning of each architecture section in this document replicates this diagram with the views appropriate for that section highlighted.

8.0 ARCHITECTURAL ELEMENTS

There are four areas of architecture that are commonly accepted as subsets of an overall enterprise architecture, all of which TOGAF is designed to support:

- Business (Process/Core)
- Data
- Application
- Technology

The subsequent sections define these four areas. Based on the PDS Level 1,2,3 Requirements [2], the PDS has been decomposed into elements and organized based on these four architecture areas. The following diagram details this decomposition:

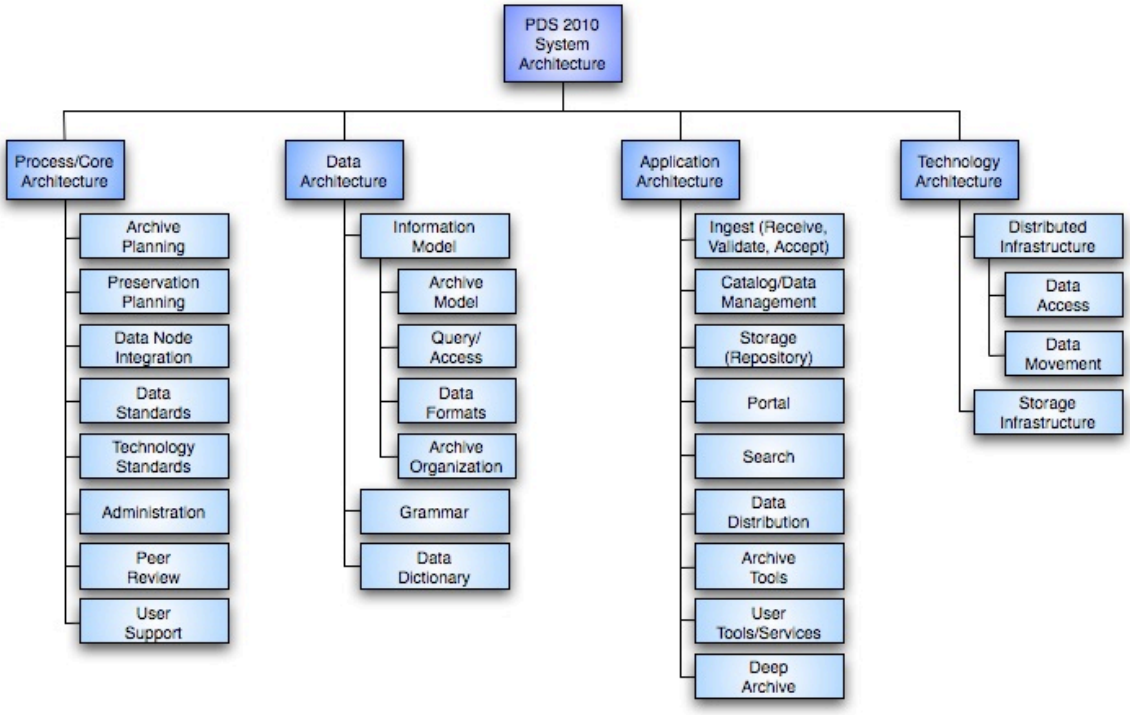

Figure 8.1: Architectural Decomposition

The following sections specifically address the elements in each of these architectural areas and specify the derivation of an element from PDS Requirements [2].

8.1 Process/Core Architecture

The following elements are associated with the Process/Core Architecture area:

Archive Planning

This element covers the necessary processes for archiving data within PDS. Derived from requirement 1.3.

Preservation Planning

This element covers the necessary processes for long-term preservation of PDS data. Derived from requirement 4.1.

Data Node Integration

This element covers the necessary processes for integrating data nodes into PDS. There is currently no PDS level 1,2,3 requirement for integrating data nodes into PDS.

Data Standards

This element covers the process for managing the PDS data standards. Derived from requirement 1.4.6.

Technology Standards

This element covers the process for managing and selecting technology standards, including interface standards, necessary for PDS to function as a federation. There is currently no PDS level 1,2,3 requirement for managing and selecting technology standards.

Administration

This element covers the policies and processes that are necessary to develop and operate the PDS. Derived from requirements 2.10.2, 4.1 and 4.2.

Peer Review

This element covers the processes for performing peer reviews of data submissions. Derived from requirement 2.4.

User Support

This element covers the policies and processes for supporting users. Derived from requirement 1.2.

8.2 Data Architecture

The following elements are associated with the Data Architecture area:

Information Model

This element identifies the overall information model for PDS including the objects, attributes and their relationships. Derived from requirements 1.4.2 and 1.4.4.

Archive Model

This sub-element of Information Model identifies the archive view of the information model. Derived from requirement 1.4.1.

Query/Access

This sub-element of Information Model identifies the query view(s) of the information model. These views are discipline-dependent. There is currently no PDS level 1,2,3 requirement for developing query views for the information model.

Data Formats

This sub-element of Information Model identifies the data formats needed for archiving of data. Derived from requirement 1.4.1.

Archive Organization

This sub-element of Information Model identifies the necessary structural layout of a repository. Derived from requirement 1.4.1.

Grammar

This element identifies the necessary grammar(s) for describing PDS data. Derived from requirement 1.4.3.

Data Dictionary

This element identifies the data dictionary as a component of the information model. Derived from requirement 1.4.2.

8.3 Application Architecture

The following elements are associated with the Application Architecture area:

Ingest (Receive, Validate, Accept)

This element describes the components that manage the receipt, validation and acceptance of PDS data deliveries. These ingest functions are described in the Open Archival Information System (OAIS) Reference Model. Derived from requirements 2.2, 2.3, and 2.5.

Catalog/Data Management

This element describes the components that manage the cataloging and tracking of PDS data including both PDS products and data sets. This function is also described in the OAIS Reference Model. Derived from requirements 2.6 and 2.2.2.

Storage (Repository)

This element describes the storage management components for managing the PDS archive. This function is also described in the OAIS Reference Model. Derived from requirement 2.7.

Portal

This element describes the access portal(s) to PDS data and related information. There is currently no PDS level 1,2,3 requirement for developing a portal.

Search

This element describes the components within the search infrastructure for searching PDS data sets and products. Derived from requirement 3.1.

Data Distribution

This element describes the distribution of PDS data to the user community including the ability to download data online. Derived from requirement 3.2.

Archive Tools

This element describes a set of archive tools for generating, validating and submitting data to PDS by data providers. Derived from requirement 1.5.

User Tools/Services

This element describes the set of tools and services for working with PDS data including visualization, transformation, display, etc. Derived from requirement 3.3.

Deep Archive

This element describes the deep archive functions and services used by the PDS. Derived from requirement 4.1.

8.4 Technology Architecture

The following elements are associated with the Technology Architecture area:

Distributed Infrastructure

This element covers the infrastructure technology for distributed access and exchange of data. Derived from requirement 2.8.

Data Access

This sub-element of Distributed Infrastructure covers the technologies and approach for facilitating access to PDS data. There is currently no PDS level 1,2,3 requirement for data access technology.

Data Movement

This sub-element of Distributed Infrastructure covers the technology necessary to move data between data providers, data nodes, and the NSSDC. There is currently no PDS level 1,2,3 requirement for data movement technology.

Storage Infrastructure

This element covers the technology for storing and managing the terabytes of data in the PDS archive. There is currently no PDS level 1,2,3 requirement for storage technology.

9.0 PROCESS/CORE ARCHITECTURE

The Process/Core Architecture represents the business strategy, governance, organization, and key business processes of PDS. Now, PDS does not function as a business but we do have key processes and procedures. The highlighted portion of the architecture decomposition diagram below indicates the elements associated with this portion of the system architecture:

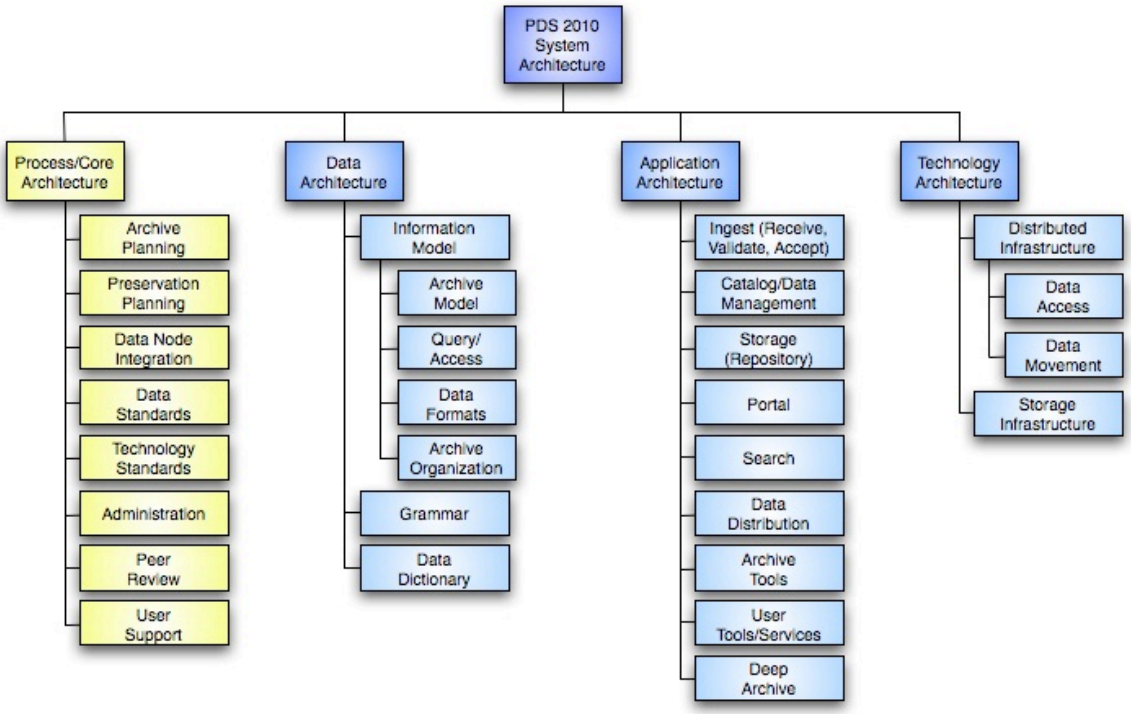

Figure 9.1: Architectural Decomposition

The majority of the process/core elements highlighted above relate to policy, process and procedure and will be addressed as needed to complement the other portions of the architecture. The policies, processes and procedures that currently exist for PDS will be carried forward and optimized for the PDS 2010 system. They are available from the password protected PDS Engineering Node web site at:

http://pds-engineering.jpl.nasa.gov/index.cfm?pid=128

The will not be addressed further in this document.

10.0 DATA ARCHITECTURE

The Data Architecture represents the structure of an organization's logical and physical data assets and data management resources. The highlighted portion of the architecture decomposition diagram below indicates the elements associated with this portion of the system architecture:

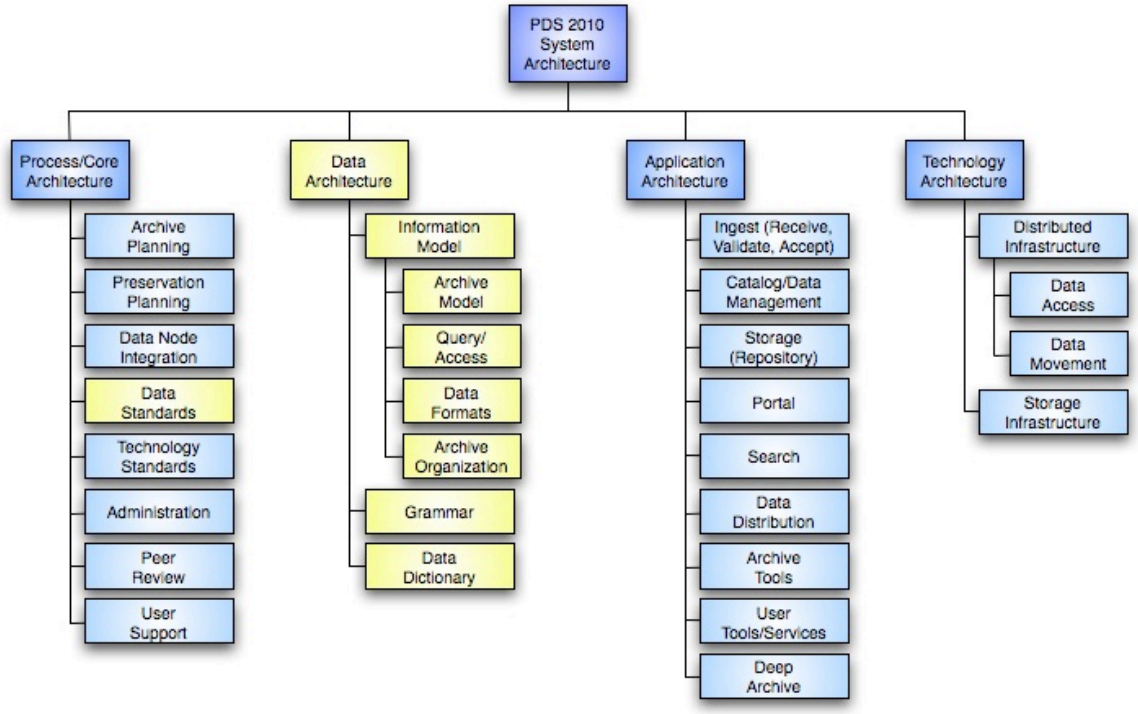

Figure 10.1: Architectural Decomposition

The principle objectives of this work have been to address the vision for PDS 2010, as defined by the PDS Management Council at its April 2008 meeting:

- 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

This section of the document describes that part of the PDS 2010 operational concepts that are either directly or indirectly affected by the data architecture (i.e., how the PDS4 Information Model and the other data architecture elements will be used (either directly or indirectly) within the archive lifecycle). This section

details use scenarios for each of the data architecture elements. The PDS4 Information Model Specification document [13] addresses the data architecture design.

As defined in the Viewpoints and Views section of this document, there are a number of views utilized to represent the data architecture. The following figure depicts three aspects of a data driven architecture:

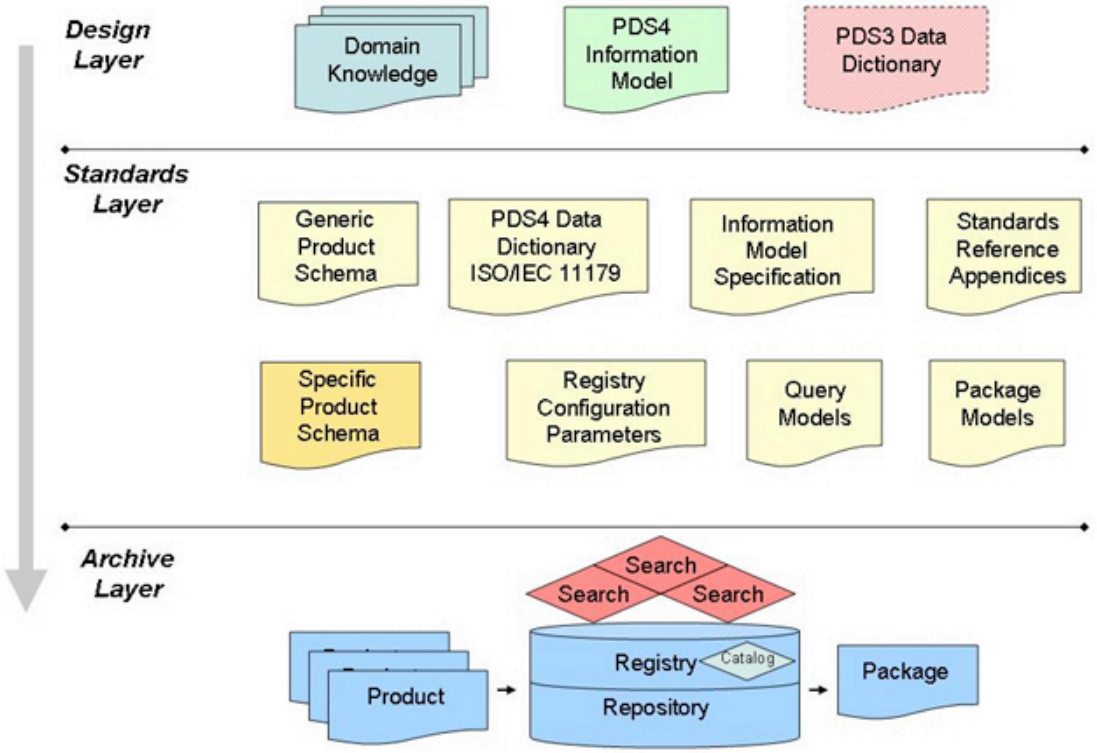

Figure 10.2: Data Driven Architecture

The basic premise of Data Driven Design is that the information known about a system, application, or organization, operating within the PDS, provides the primary means to define the processes that will manage the information. In essence, the processes exist to process/manipulate the underlying data (e.g., data centric applications such as models, registries, databases, archives, digital libraries, etc.).

For information systems, an information model is a conceptual representation of the entities along with their attributes and operations, which comprise the body of knowledge (corpus), which is stored and operated on within the system. An information model serves as a guide for the design and development of the system. Formally specifying an information model, as PDS 2010 is, will make management of the model simpler since revisions can be tracked and consistency within the model can be checked (ensured/enforced) with existing modeling tools.

10.1 Principles

The development of the PDS4 data architecture follows a set of principles, derived from the architectural principles, so that individual efforts such as those of the PDS4 Data Model Working Group can be coherent with the overall vision of PDS 2010. Some principles related to the PDS4 information model are:

Interoperability
The PDS works to ensure interoperability among planetary science archives by seeking community consensus on a core set of common objects and data elements.

Partitioning
The data model is logically separated into partitions in order to allow for management and evolution of components of the data model independently. For example, the imaging community manages the image model.

Formal Specification
The data model is explicitly and unambiguously defined using a formal data engineering notation and/or language.

Standards
PDS applies commonly accepted and documented standards that address its requirements.

Evolve-ability and Flexibility
The data model should be extensible and flexible enough to meet new requirements.

Model Expressions
The data model is implementation neutral and can have different expressions to support subsystem functions:

- Database Development
  A database developer needs an Entity-Relationship, UML model or schema.
- User Interface
  A web developer requires a taxonomy for navigation and keyword/values for facet-based navigation system.
- Persistent Storage
  Express an instance of all or part of the data model in a form that can be stored indefinitely (XML, ODL, etc.).
- Model-to-Model Mapping

Where possible and necessary, translations from the PDS data model to other data model expressions in order to support the interchange of information.

- Notation Conversion
  Express the model using a number of notations or languages (e.g., Ontology, UML, ER, Taxonomy, etc.).

10.2 Data Architecture Representations

The following diagram depicts the four representations of the Data Architecture design:

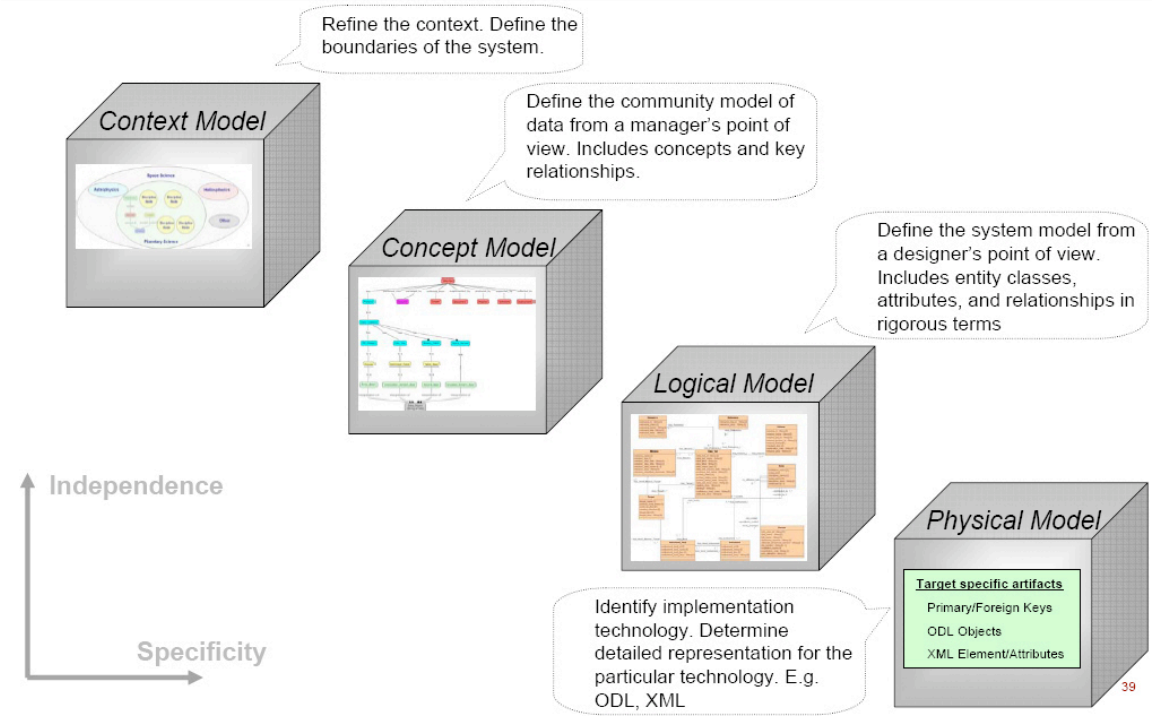

Figure 10.3: Data Architecture Representations

The above representations of the data architecture illustrate the traditional four technical views and the tradeoff of model independence versus model specificity.

10.3 Data Architecture Terms and Definitions

The following are some definitions of essential terms used throughout the remaining part of the data architecture section of this document.

An *attribute* is a property or characteristic that provides a unit of information about a *class*.

A *class* is the set of attributes, which identifies a family. A *class* is generic – a template from which individual members of each family may be constructed.

A *data object* is constructed from a *class*. It is a specific instance of a *class*. A *data object* can be one of three types, digital, conceptual, or physical.

A *digital object* is an object consisting of digital information. For example, an image is a *digital object*.

A *physical object* is tangible. For example, a spacecraft instrument is a *physical object*.

A *conceptual object* is an object that is not tangible. For example, a mission is a *conceptual object*.

A *descriptive class* provides information that is useful for the interpretation of a *data object*.

A *structural class* provides information that defines the components and organization of a *digital object*.

In addition, there are several specific terms used in the definition of a data element.

A *data element* is a unit of data for which the definition, identification, representation and *permissible values* are specified by means of a set of attributes. For example, the concept of a *calibration_lamp_state_flag* is used in the PDS archive to indicate whether the lamp used for onboard camera calibration was turned on or off during the capture of an image. The *data element* aspect of this concept is the named attribute (or data element) *calibration_lamp_state_flag*.

A *data element concept* can be represented in the form of a *data element*, described independently of any particular representation. The *data element concept* for *calibration_lamp_state_flag* is the generic notion of a *calibration lamp state*.

A *value domain* is specified by a list of all its *permissible values*. These can be enumerated or non-enumerated. The *value domain* for *calibration_lamp_state_flag* is an enumerated list containing "YES" and "NO".

A *permissible value* is an expression of a *value meaning* allowed in a specific *value domain*. A *permissible value* for *calibration_lamp_state_flag* is "YES".

A *value meaning* is the meaning or semantic content of a *permissible value*. A *value meaning* for *calibration_lamp_state_flag* would be a definition of "YES" as a binary state.

A *conceptual domain* is a set of valid *value meanings*. The *conceptual domain* for *calibration_lamp_state_flag* is a binary state.

10.4 Information Model Partitions

The PDS4 Information Model is divided into the following top-level partitions:

Participants
Individuals, organizations or objects, which contribute to or define the context for other defined entities.

Products
Entities that result from an observation or analysis, provide additional supportive information, or describe the operation or (analysis) process.

Resources
Entities that are components of the system.

Collections
Groupings of entities.

Query
Additional information for locating entities in the system.

10.5 Information Model Entities and Associations

The PDS4 Information Model defines a set of entities for each of the above partitions. The following list illustrates a representative set of entities for each of the above partitions:

Participants
- Mission
- Observatory
- Instrument
    - Detector
- Person
- Reference
- Target

Products
- Sample (Physical)

- Data Structure (Digital)
    - Catalog (record collection)
    - Table (row, column)
    - Image (x,y,z)
    - Movie (x,y,z,t)
    - N-Array
- Documents
- Software

Resources

- Repositories
- Registries
- Web sites
- Services

Collections

- Volumes
- Datasets
- Events
- Campaigns

The following diagram provides a representation of associations between the above entities:

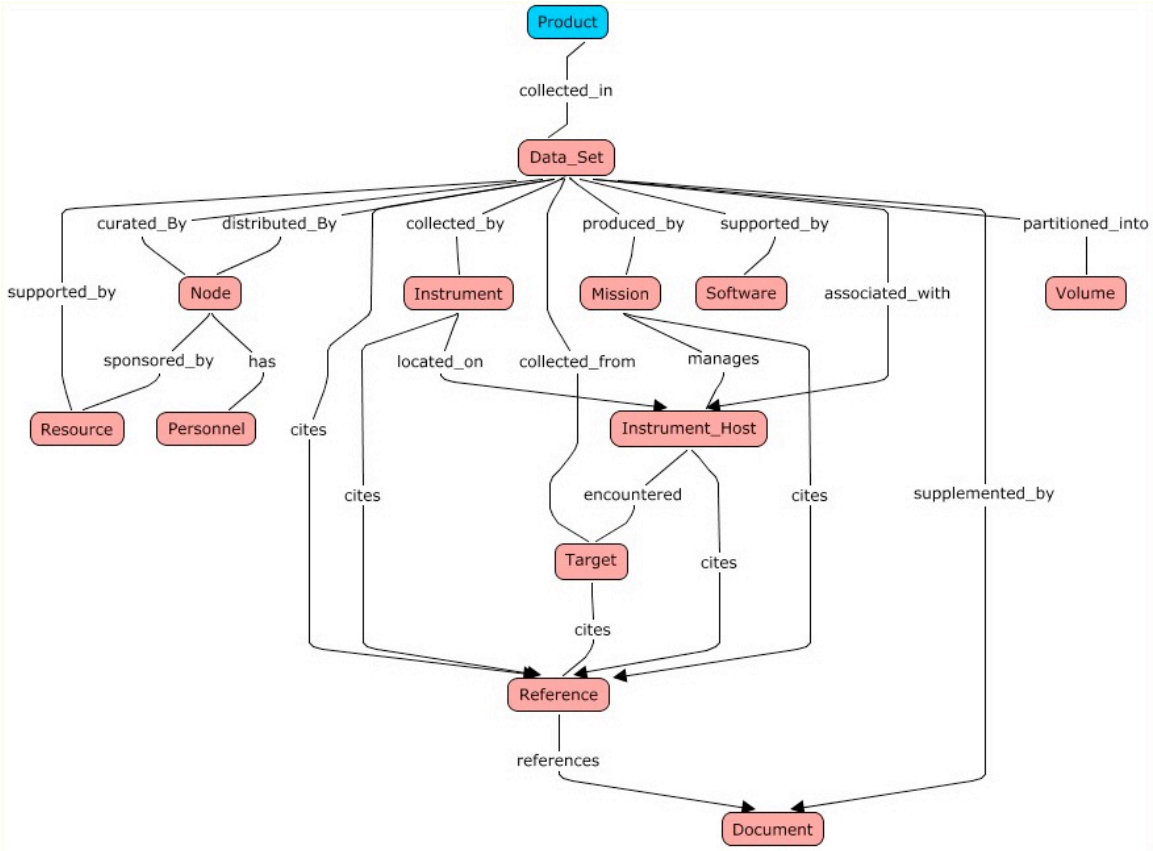

Figure 10.4: Product Entity Relationships

11.0 APPLICATION ARCHITECTURE

The Application Architecture represents a blueprint for deployment of the individual application systems, their interactions, and their relationships to the core business processes of the organization. The highlighted portion of the architecture decomposition diagram below indicates the elements associated with this portion of the system architecture:

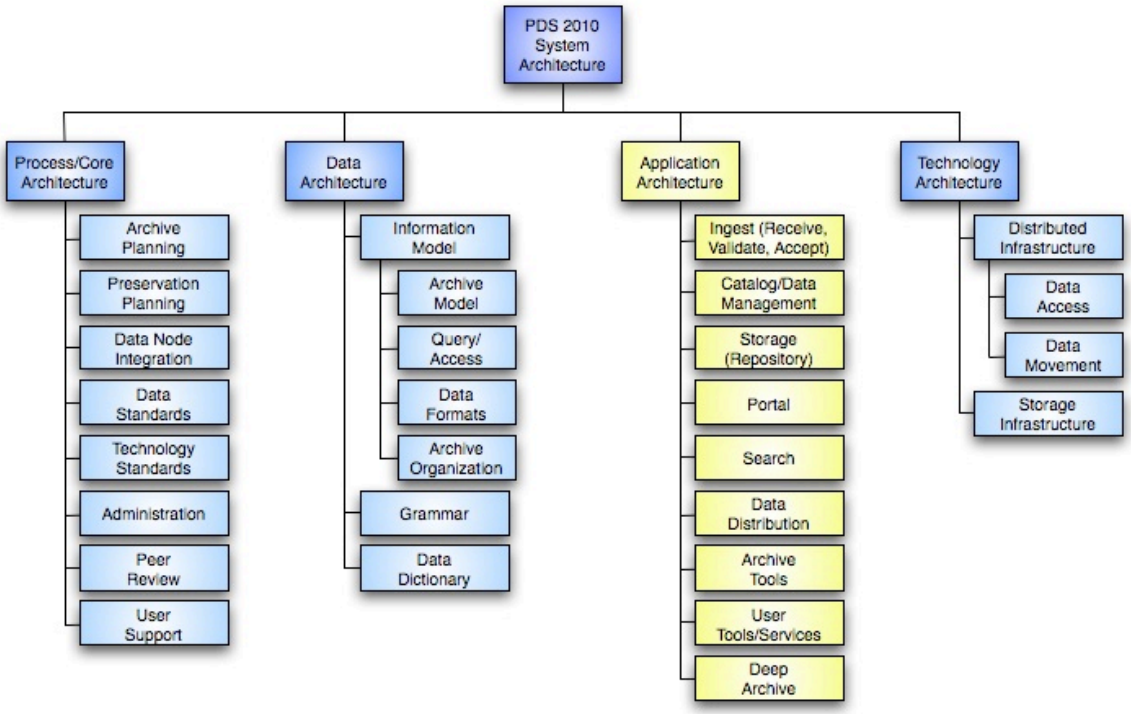

Figure 11.1: Architectural Decomposition

Although it was not entirely predetermined, a Service-Oriented Architecture (SOA) was determined to be the appropriate software architecture for the PDS 2010 system as derived from requirement 2.8 [2]. The definition of SOA is:

> A software architecture for building applications that implement business processes or services using a set of loosely coupled black-box components orchestrated to deliver a well-defined level of service. [7]

There are several advantages to adopting SOA, including:

- Captures many of the best practices of previous architectures.
- Well suited for a distributed system.
- Promotes “loose coupling”, “software reuse”, and “encapsulation”.
- A service-based architecture provides currency and timeliness for the system.

Although we have decided on a software architecture, the work is not complete because there are still several options to consider for deployment. The SOA architecture can operate under several protocols (e.g., SOAP, REST, Web Services, etc.). In addition, the SOA solution will be tailored for PDS meaning that the service-based functionality will focus on search and retrieval of data and a tool-based approach is still appropriate for certain functions within the system. These decisions and others like it will be made during the design phase.

As defined in the Viewpoints and Views section, there are a number of views that represent the application architecture. The content that follows falls mostly in the Service Identification and Service Definition areas of the PDS/Zachman mapping (see section 7.2).

11.1 Service Identification

A definition of service with respect to SOA is as follows [8]:

- Is a logical representation of a repeatable business activity that has a specified outcome (e.g., check customer credit; provide weather data, consolidate drilling reports)
- Is self-contained
- May be composed of other services
- Is a “black box” to consumers of the service

This section identifies the software components (services, tools and applications) for the PDS 2010 system along with definitions for each. The following diagram identifies the components:

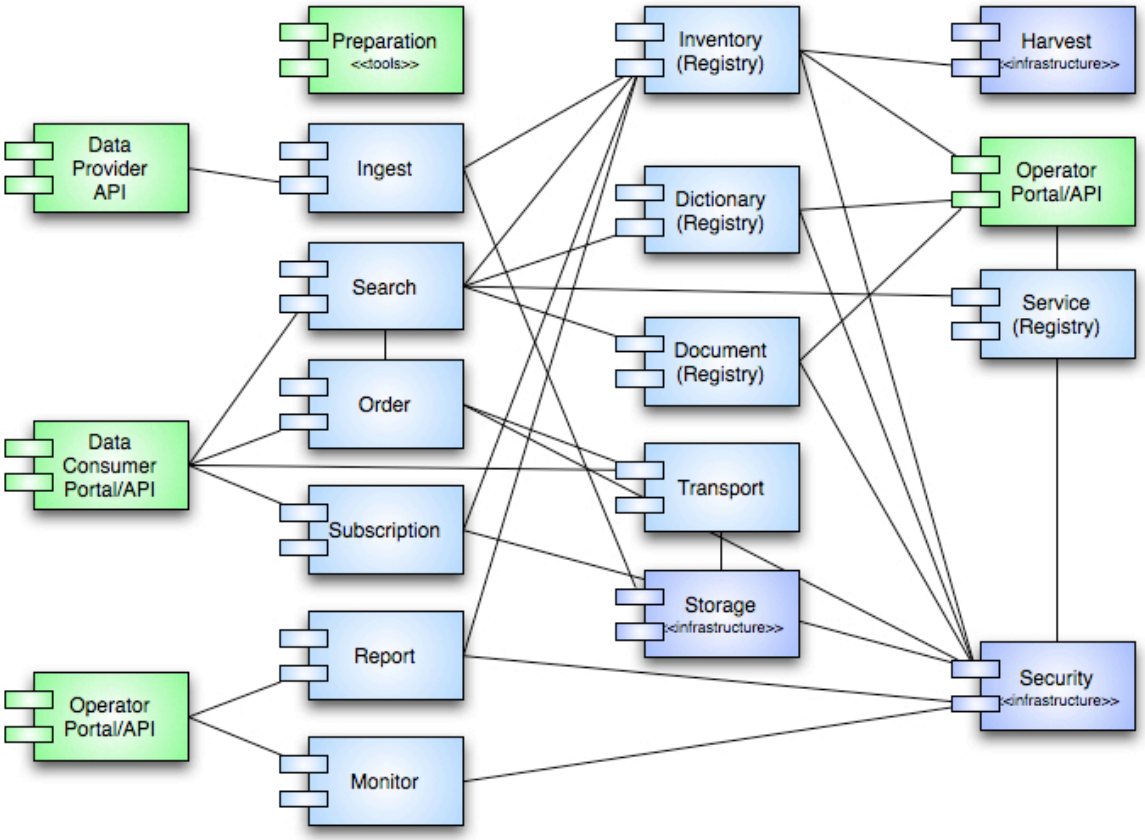

Figure 11.2: System Context

The green components represent tools, APIs or portals that utilize or access the functionality of the other system components. The darker blue components represent infrastructure functionality and typically do not have external interfaces. The lighter blue components represent functionality in the system packaged as a service and typically offer external interfaces.

11.1.1 Service Definitions

The service definitions that follow will provide starting points for specifying level 4 and 5 requirements for each service. The list of services is in alphabetical order:

Dictionary Service

This service is an instance of the Registry service and provides functionality for capturing the data dictionary, which consists of object/element definitions and their associations. The Data Model manages the definitions and periodically exports them to the Dictionary service. A commonly used standard for such registries is ISO/IEC 11179.

This service does not derive directly from an Application Architecture architectural element but does relate to the Data Dictionary element under the Data Architecture.

Document Service

This service is an instance of the Registry service and provides functionality for managing project documents (e.g., Archive Preparation Guide, Proposers Archiving Guide, etc.), product label schemas, etc. The scope of this service includes Node-specific documents as well as PDS-wide documents including documents not meant for public consumption.

This service does not derive directly from an Application Architecture architectural element but would fulfill the need to capture and manage documents across PDS.

Ingest Service

This service provides functionality for receiving data and metadata from data providers for ingestion into PDS. The process of ingestion involves validation (i.e., syntactic and semantic) of the metadata and verification (e.g., checksum) of the data. This service acts as the interface to the Inventory and Storage services for the data provider.

This service derives from the Ingest (Receive, Validate, Accept) architectural element.

Inventory Service

This service is an instance of the Registry service and provides functionality for managing the metadata registry. This includes population of metadata from the Ingest and Harvest services as well as responding to queries from the Search Service. This service would support registration of catalog and product metadata in a number of distributed instances located at the Discipline Nodes and the Engineering Node.

This service derives from the Catalog/Data Management architectural element and relates to the Archive Model element under Data Architecture.

Monitor Service

This service provides functionality for monitoring service status across the system and facilitates notification of appropriate Node staff if a service were inoperable. A Commercial Off-The-Shelf (COTS) product would most likely satisfy this service.

This service is not derived directly from an Application Architecture architectural element but can be derived from requirement 2.10.1 and is necessary to manage and operate a system of distributed services.

Order Service

This service provides functionality for requesting special PDS services (i.e., science-related services) from the Discipline Nodes (e.g., offline data delivery, value added products, subsetting, etc.). This Node-specific service would provide a common interface for placing orders and will be tailor-able on the backend to support the types of orders supported at the Node.

This service derives from the Data Distribution architectural element.

Registry Service

This service provides functionality for tracking, auditing, locating, and maintaining artifacts within the system. These artifacts can range from products consisting of data files and label files, schemas, dictionary definitions for objects and elements, etc. This service will provide a common implementation for supporting the requirements of the Dictionary, Document, Inventory and Service (Registry) services. An implementation based on the ebXML standard is under consideration.

This service derives from the Catalog/Data Management architectural element.

Report Service

This service provides functionality for capturing and reporting metrics. Although each new service will have functional requirements to track metrics, those metrics should be submitted to this service via a common interface or captured in a common format so that they can be harvested by this service. The service is not limited to metrics generated by PDS 2010 services, but should also include metrics from the FTP and web logs from each of the nodes as well as any other commonly generated metric. A Commercial Off-The-Shelf (COTS) product would most likely satisfy this service.

This service does not derive directly from an Application Architecture architectural element but is a definite need within PDS.

Search Service

This service provides functionality for accepting queries from data consumers for registered artifacts. It also includes functionality for retrieving search results. This service acts as the interface to the Inventory Service for the data consumer. The service itself will consist of indexed metadata utilizing common facets to facilitate

search across the Nodes along with Node-specific facets utilized by Node-specific search applications.

This service will represent the next-generation of the PDS Data Distribution system (PDS-D). Existing applications (e.g., Planetary Image ATLAS, Geosciences Orbital Data Explorer, etc.) that rely on the PDS-D infrastructure will require a transition path or the service will need to be backward compatible.

This service derives from the Search architectural element and relates to the Query/Access element under Data Architecture.

Security Service

This service provides the authentication and authorization functions for the system. In addition to security, this service could encompass directory service functionality by utilizing a standard such as the Lightweight Directory Access Protocol (LDAP). The intent of this service is to control access to interfaces and services that require authentication and authorization (e.g., Monitor, Report, Registry instances, etc.).

This service does not derive directly from an Application Architecture architectural element but is necessary to limit access to internal interfaces and to facilitate tracking and curation.

Service (Registry) Service

This service is an instance of the Registry service and provides a registry for other services that are available in the system. A service like this is often associated with Service Oriented Architectures (SOAs), and specifically with architectures based on Web Services.

This service does not derive directly from an Application Architecture architectural element but does derive from requirement 2.9.2 and is a necessary service in a SOA system.

Storage Service

This service provides functionality for managing the data repository. This includes movement of data files in and out of the repository as well as periodic verification of those files. This includes packaging and movement of the data files via the Transport service to the deep archive (e.g., NSSDC, SDSC) or other external repositories.

This service derives from the Storage (Repository) and Deep Archive architectural elements and relates to the Storage Infrastructure element under Technology Architecture.

Subscription Service

This service provides functionality for subscribing to data, document and software release announcements. A commonly used method/format for such services is Really Simple Syndication (RSS).

This service derives from the Search architectural element.

Transport Service

This service provides functionality for moving data across the network. This service returns search results (products) to data consumers or for pushing data to the deep archive. This service would also incorporate the transformation functions of file format conversion, coordinate transformation, subsetting and packaging.

This service derives from the Data Distribution and User Tools/Services architectural elements and relates to the Data Movement element under Technology Architecture.

11.1.2 Other Component Definitions

The component definitions that follow will provide starting points for specifying level 4 and 5 requirements for each component. The list of components is in alphabetical order:

Data Consumer Portal

This component provides a web-based interface for discovering and accessing products. The PDS-wide portal (http://pds.nasa.gov/) will provide web applications for accessing catalog-level information while the Node-specific portals will provide web applications for accessing product-level information. This component also includes existing Node-specific applications (e.g., Planetary Image ATLAS, Geosciences Orbital Data Explorer, etc.) that are candidates for transitioning to the PDS 2010 system. The search-related applications that make up the portal will utilize the Search service API where appropriate to discover registered artifacts.

This component derives from the Portal architectural element.

Data Provider API

This component provides an Application Programming Interface (API) that will allow Data Providers to interface with the Ingest service. This API will enable

incorporation into their data production pipelines for delivery of products in real time.

This component does not derive directly from an Application Architecture architectural element but does conform to best practices for providing an interface into the system.

Harvest Tool

This tool provides functionality for capturing and registering product metadata. The tool will run locally at the Discipline Node to crawl the local data repository in order to discover products and register associated metadata with the Inventory service.

This service derives from the Catalog/Data Management architectural element.

Operator Portal

This component provides a general web-based interface for managing registry policy and content. The portal will utilize the Registry service API for accessing any of the registry instances. Access to the portal will be restricted requiring authentication and authorization via the Security service.

This component derives from the Portal architectural element.

Preparation Tools

This component provides functionality for preparing data for ingestion into PDS. This includes product label design, generation, transformation and validation as well as package formation. The component consists of a suite of tools (i.e., similar to the current Validation Tool).

This component derives from the Archive Tools element.

11.1.3 Interface Definitions

The design phase will define the interface definitions but in general, where web-based service interfaces are required, a RESTful interface will be utilized. The Registry and Search service interfaces will utilize REST, which stands for Representational State Transfer. Other services that integrate COTS or open source solutions will utilize their provided interfaces.

11.2 Service Provisioning

This section details the provisioning for the services described above. This covers whether services will deployed centrally versus distributed at the nodes or whether they will have common or node-specific implementations.

For comparison purposes, the following diagram details provisioning of the "services" for the current system:

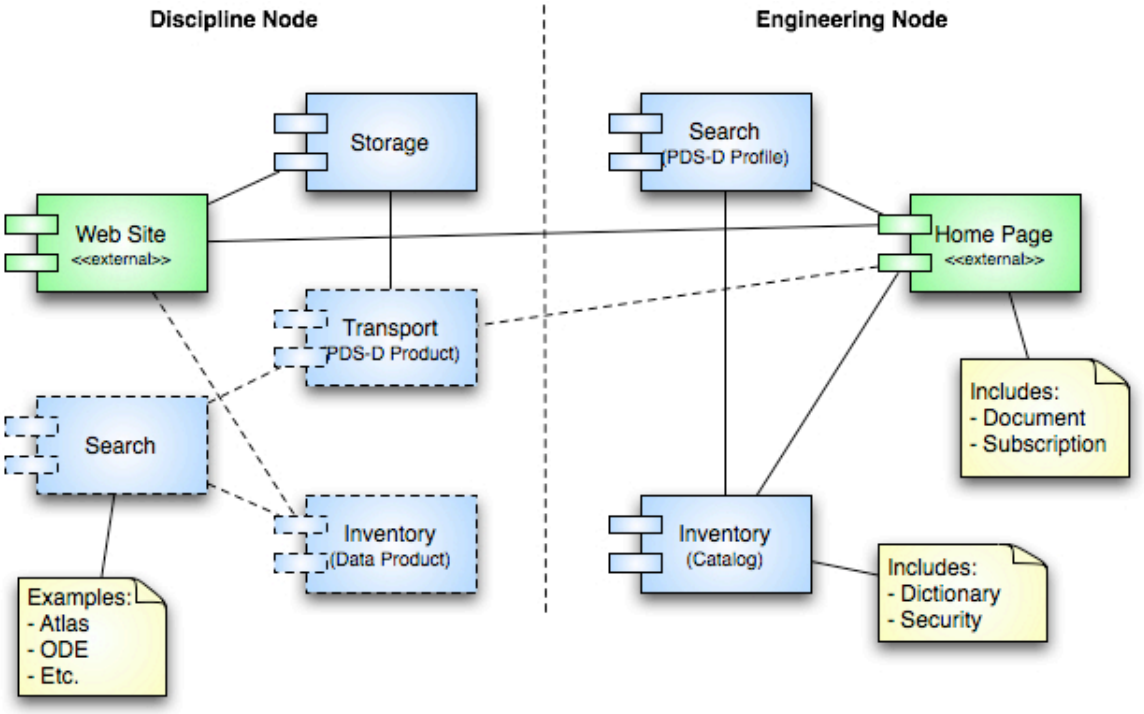

Figure 11.3: Service Provisioning (Today's PDS)

In the diagram above, the service names correspond with the proposed PDS 2010 services for consistency. Services with dashed outlines are not available at all nodes. The notes on the Engineering Node side of the diagram indicate service-type functionality that is included with that "service". For the current architecture, those services have been broken out into their own services. We are quoting the word "service" here, because most of the boxes in the diagram do not represent services as defined earlier in this section with the exception of the PDS-D Profile and Product components.

In comparison to the diagram above, the following diagram details the provisioning of the services for the PDS 2010 system:

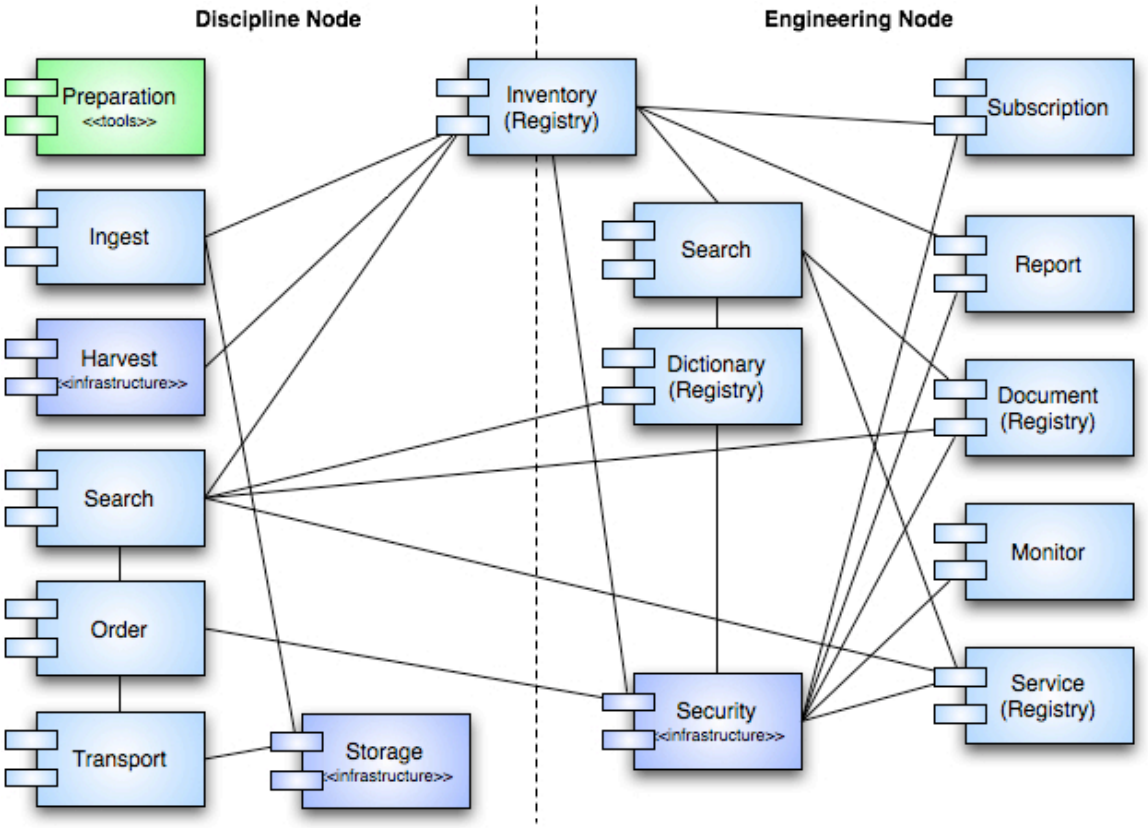

Figure 11.4: Service Provisioning (PDS 2010)

Services on the Engineering Node side of the diagram are common services and are available to all of the nodes. Services on the Discipline Node side are a mix of common and node-specific services. For example, the Search service would consist of a base implementation but allow node-specific extensions to support additional query parameters related to node-specific data. In general, services intended for Node deployment should be adaptable to support Node-specific products and metadata.

The sections that follow detail specific functional scenarios and how data would flow between the provisioned services.

11.2.1 Ingestion Scenarios

Both ingestion scenarios cover ingestion of catalog and product metadata and its associated data into the PDS. The first scenario offers a tool-based interface to Data Providers to facilitate adoption of and interfacing with the PDS 2010 system. It includes transformation of incoming data/metadata as a possible function for the Data Provider or the Discipline Node via a PDS provided tool. Introduction of a Harvest service facilitates capturing and registering product metadata. Introduction of an Inventory service facilitates tracking of metadata

submissions and their associated products. The following diagram details the ingestion flow for the first scenario:

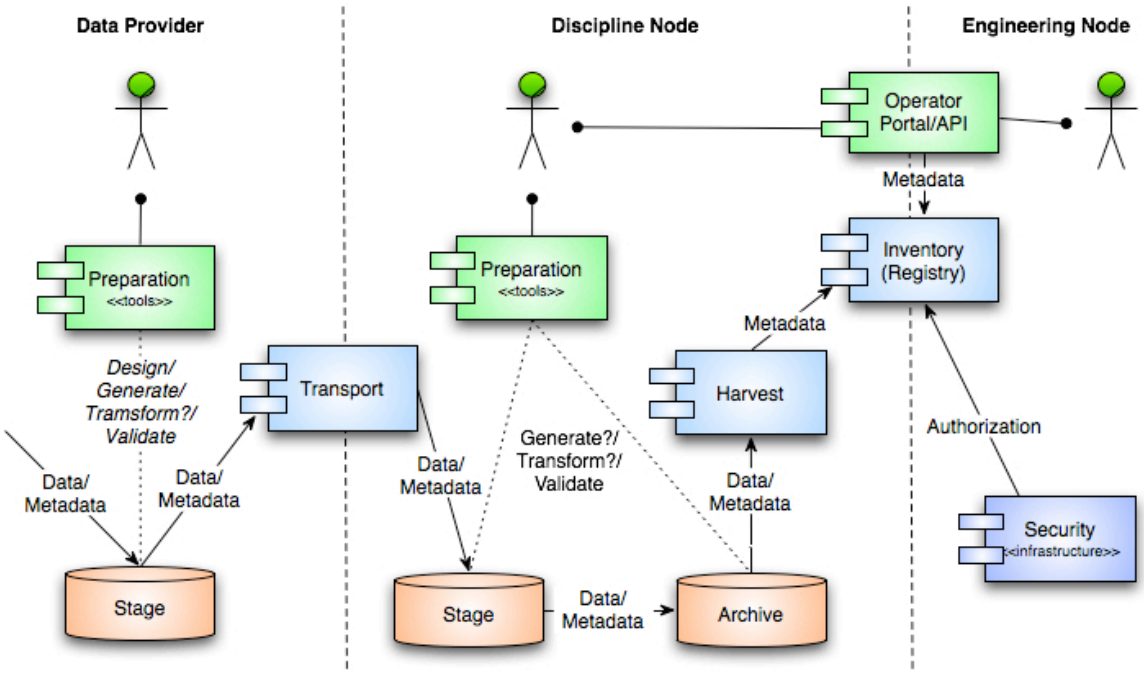

Figure 11.5: Ingestion Scenario (Harvest Service)

The following steps in the ingestion flow elaborate on the diagram above:

1. Data Provider receives data from the source (e.g., Project, Instrument Team, etc.).
2. Data Provider utilizes PDS provided tools to prepare the data for submission.
3. Data Provider submits transformed/labeled data to the Discipline Node via an agreed interface (e.g., FTP, Data Brick, etc.).
4. Discipline Node receives data/metadata from the Data Provider and stages it in local storage.
5. Discipline Node utilizes PDS tools or tools based on a common library to prepare the data for archive.
6. Discipline Node initiates harvesting of the archive, which registers product metadata in the Inventory service. Metadata registrations are authorized by the Security service.
7. Discipline Node manages housekeeping information and/or augments metadata for search enhancement via the Operator Portal.

The second scenario focuses on using the Ingest service instead of the Harvest service for product registration. The Ingest service, planned for a later development phase, will facilitate the incorporation of product registration into a Data Provider's pipeline. The following diagram details the ingestion flow for the second scenario:

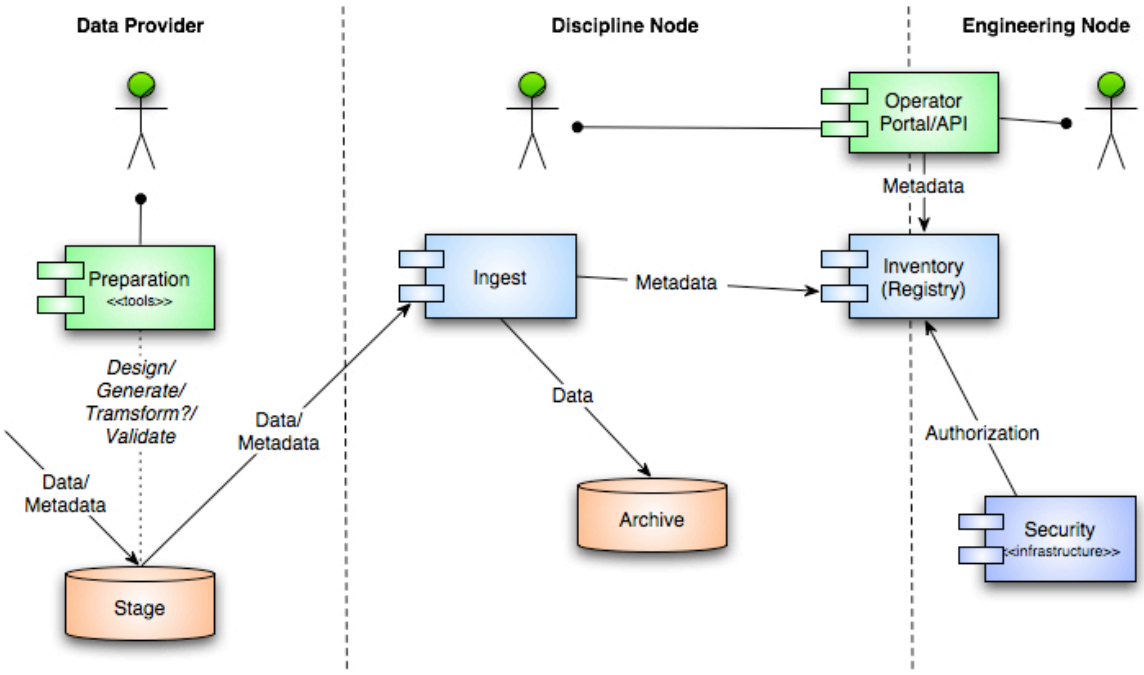

Figure 11.6: Ingestion Scenario (Ingest Service)

The following steps in the ingestion flow elaborate on the diagram above:

1. Data Provider receives data from the source (e.g., Project, Instrument Team, etc.).
2. Data Provider utilizes PDS provided tools to prepare the data for submission.
3. Data Provider submits transformed/labeled data to the Discipline Node via the Ingest service where the service registers product metadata in the Inventory service and the data are stored in the archive. Metadata registrations are authorized by the Security service.
4. Discipline Node manages housekeeping information and/or augments metadata for search enhancement via the Operator Portal.

PDS 2010 deployment will take a phased approach with the following aspects regarding ingestion:

- Utilize tool-based interfaces for design, generation and validation of product submissions. This approach allows for utilization of existing Node processes and procedures for ingestion of products. It will minimize up-front interface changes for Data Providers.
- Facilitate ingestion of catalog and product metadata using the Harvest service. This approach allows for periodic or on-demand ingestion of metadata. It is also adaptable to different directory structures and data models allowing for ingestion of PDS3 and earlier data.

- The web-accessible Ingest service, planned for later on in the development cycle, will leverage the capabilities developed for the tools and the Harvest service.

11.2.2 Distribution Scenarios

The distribution scenarios cover search of the catalog and product metadata and distribution of associated data. Introduction of a common Search service facilitates interfacing with the services hosting catalog and Node-specific product metadata. Introduction of an extensible but common, Transport service facilitates access and usability (i.e., transformation) of products. Introduction of a Data Consumer portal/API facilitates discovery and retrieval of data/metadata. The portal can be a Node developed application or developed by the Data Consumer.

The first diagram details a search initiating at a Discipline Node interface:

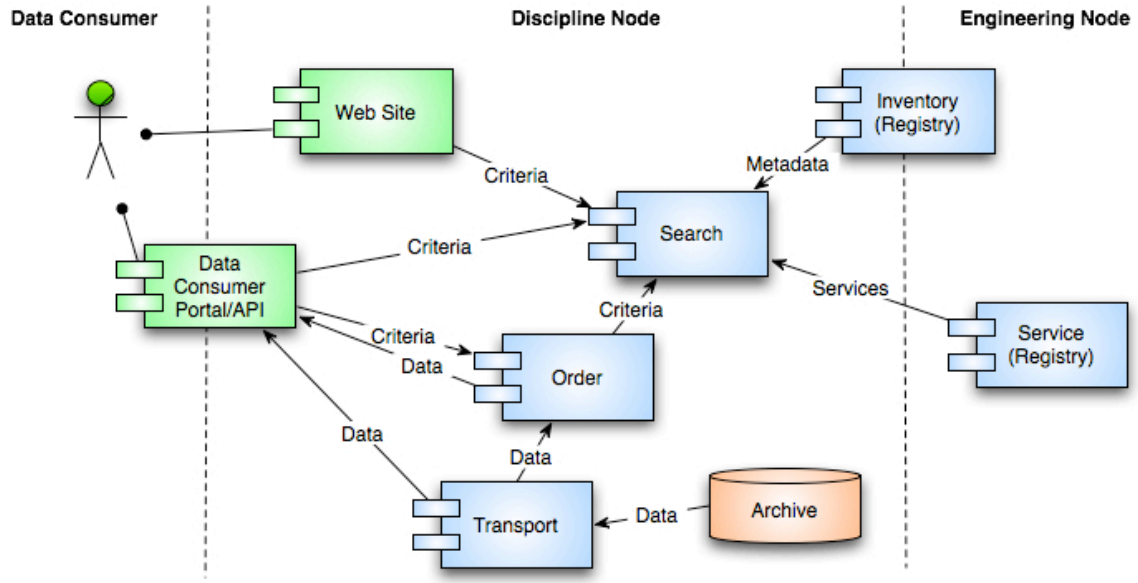

Figure 11.7: Distribution Scenario (Initiated from DN)

The following steps in the distribution flow elaborate on the diagram above:

1. Search service generates a search index utilizing the Service registry to discover the appropriate Inventory service(s) for obtaining the metadata for the index. Tailoring of the search index enables support for the Node-specific search tools.
2. Data Consumer submits a query for data through a portal/API interface or a web site.
3. Portal/API interface or web site forwards the query to the local Search service.
4. Search service returns results to the portal/API interface or web site with options for retrieving product(s) that match the query criteria. The Data

Consumer may place a special request to the Order service, if one is available at the Node, based on the results returned.

5. Data Consumer makes a request to the Transport service for delivery of the product(s).

The second diagram details a search initiating at the Engineering Node interface:

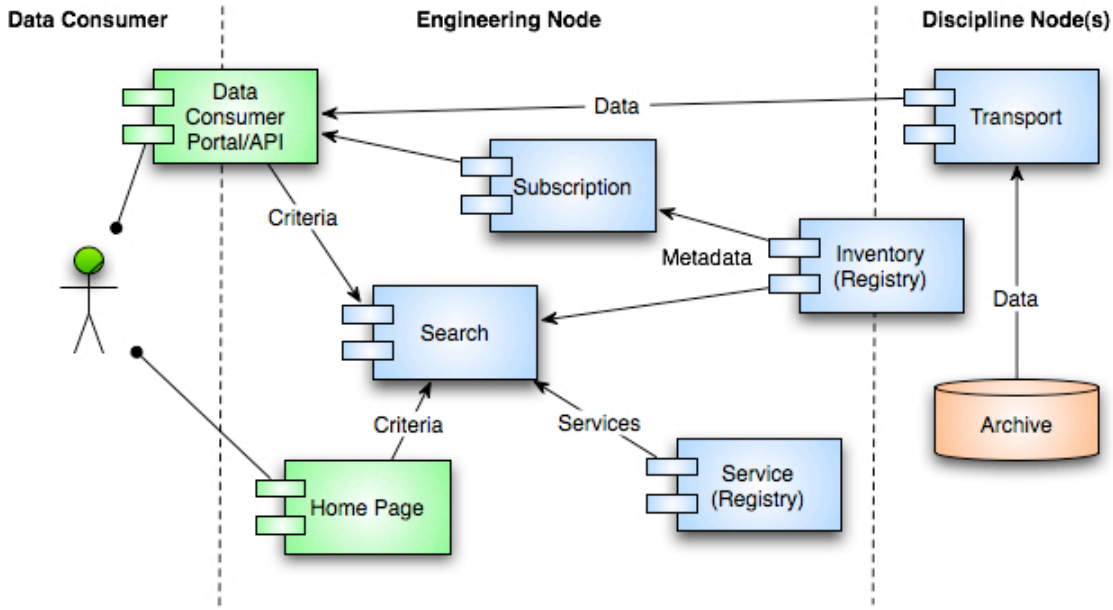

Figure 11.8: Distribution Scenario (Initiated from EN)

The following steps in the distribution flow elaborate on the diagram above:

1. Search service generates a search index utilizing the Service registry to discover the appropriate Inventory service(s) for obtaining the metadata for the index.
2. Data Consumer submits a query for data through a portal/API interface or a web site. The Data Consumer may also subscribe to release information via the Subscription service.
3. Portal/API interface or a web site forwards the query to the Search service.
4. Search service returns results to the portal/API interface or web site with options for retrieving product(s) that match the query criteria.
5. Data Consumer makes a request to the Transport service from the appropriate Node for delivery of the product(s).

11.2.3 Monitoring Scenario

This scenario details the monitoring of the services within the system. The following diagram details the monitoring flow:

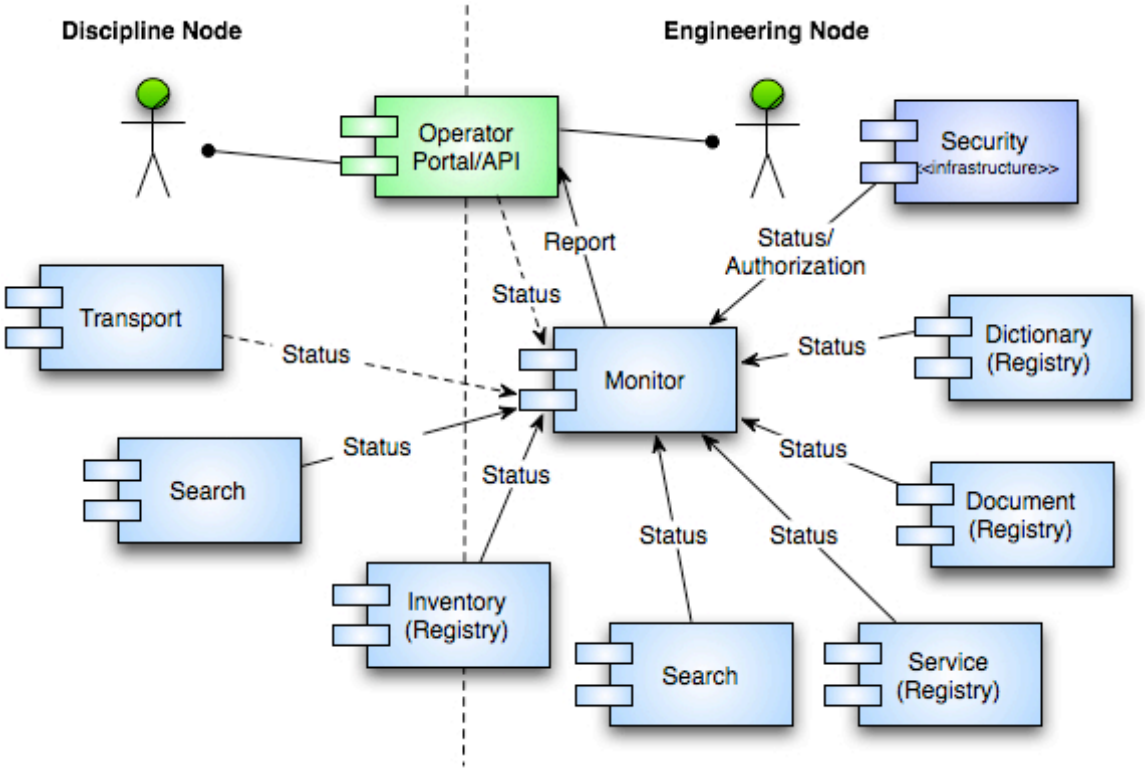

Figure 11.9: Monitoring Scenario

The following steps in the monitoring flow elaborate on the diagram above:

1. Operator (Discipline Node or Engineering Node) accesses a portal/API interface, authorized by the Security service, to view the system status.
2. Monitor service receives constant status updates from the system services.

11.2.4 Reporting Scenario

This scenario details the reporting of metrics within the system. The following diagram details the reporting flow:

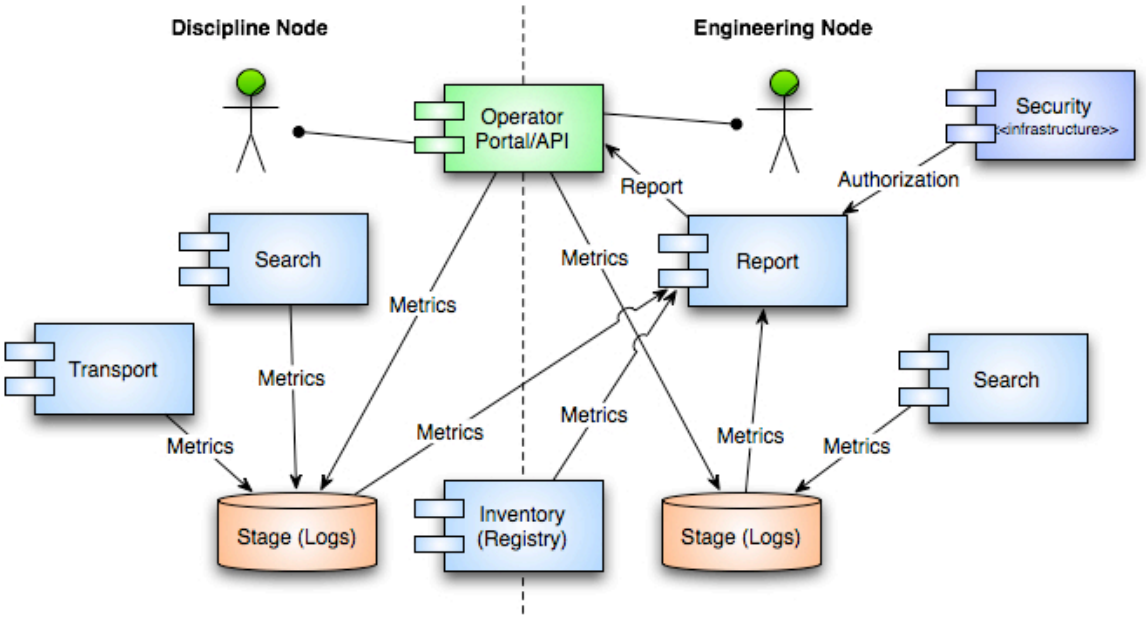

Figure 11.10: Reporting Scenario

The following steps in the reporting flow elaborate on the diagram above:

1. Operator (Discipline Node or Engineering Node) accesses a portal/API interface, authorized by the Security service, to generate a report.
2. The Report service receives periodic metrics submissions from the system services.

11.2.5 Deep Archive Scenario

This scenario details the delivery of data from storage at a Discipline Node to the deep archive. The following diagram details the deep archive flow:

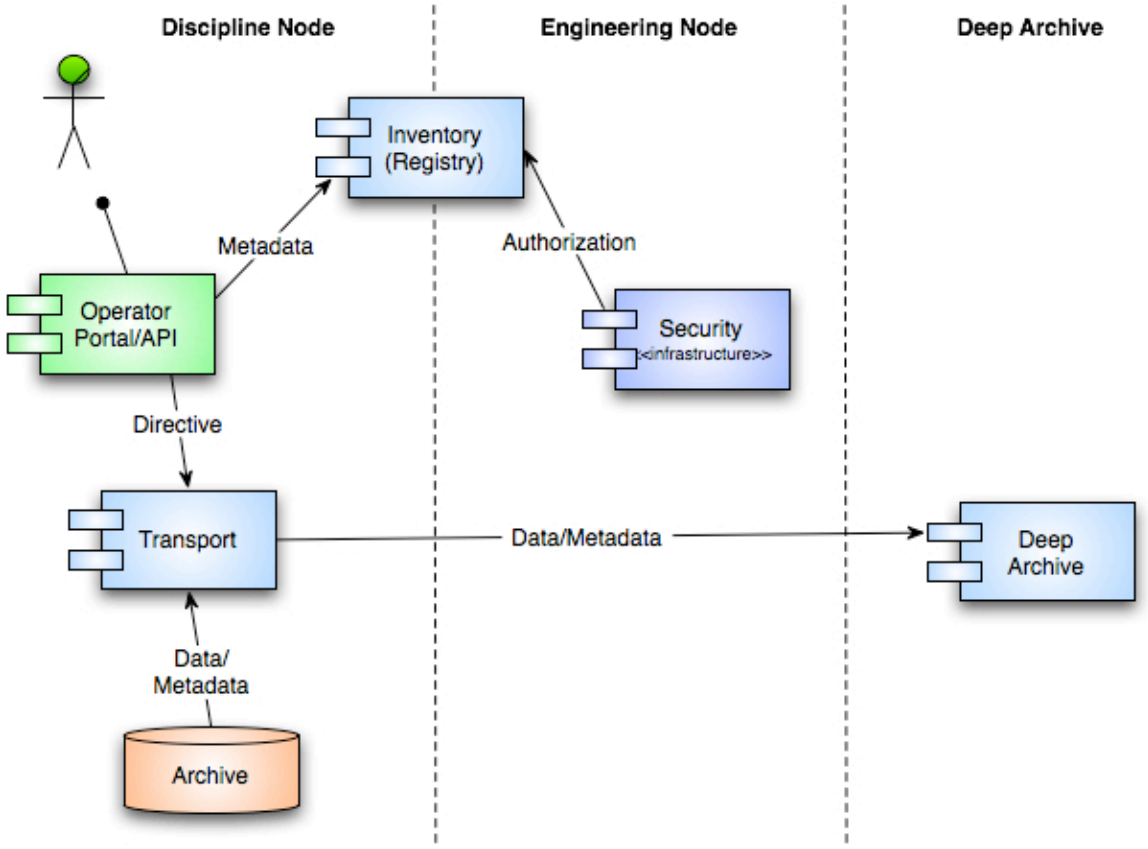

Figure 11.11: Deep Archive Scenario

The following steps in the distribution flow elaborate on the diagram above:

1. Operator submits a request for a delivery to a portal/API interface.
2. Portal/API interface sends a directive to the Transport service to package and transmit the requested delivery to the Deep Archive.
3. Operator updates the Inventory service to reflect the submission to the Deep Archive. Metadata updates are authorized by the Security service.

12.0 TECHNOLOGY ARCHITECTURE

The Technology Architecture represents the logical software and hardware capabilities that are required to support the deployment of business, data, and application services. This includes IT infrastructure, middleware, networks, communications, processing, standards, etc. The highlighted portion of the architecture decomposition diagram below indicates the elements associated with this portion of the system architecture:

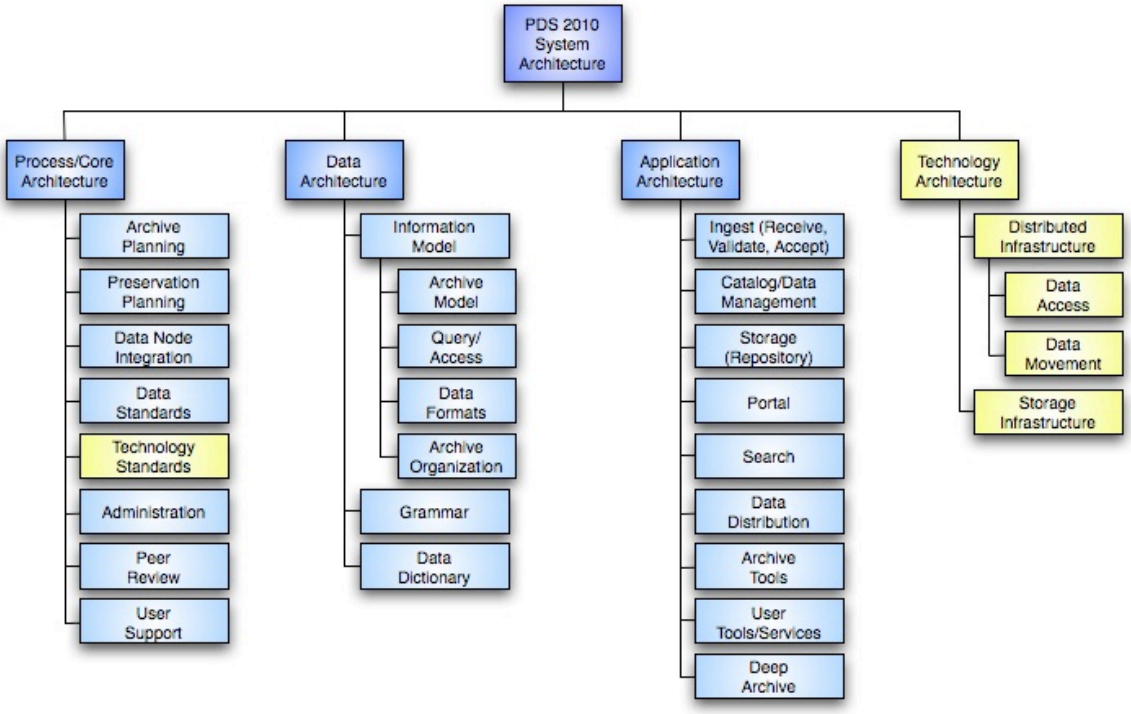

Figure 12.1: Architectural Decomposition

As defined in the Viewpoints and Views section, there are a number of views utilized to represent the technology architecture. The content that follows falls mostly in the Service Interfaces area of the PDS/Zachman mapping (see section 7.2). Up until this point in the document, we have focused on system decomposition along with function and service identification. This section focuses on technology aspects of the architecture and uses a layered approach for conveying this information. The defined system layers are as follows:

Client Layer
The client layer represents applications or interfaces utilized and/or developed by the user community.

Presentation Layer
The presentation layer represents the interface to the PDS community (Data Providers and Data Consumers). This is where general and

customized search applications reside (Data Provider, Data Consumer and Operator Portals) along with tools for preparing and manipulating the data.

Logic Layer
The logic layer represents the PDS-wide and DN-specific software and is where the services reside. Direct connection between client and logic layers are allowed depending on the interface and service accessed.

Resource Layer
The resource layer represents the PDS-supported platforms and networks hosting the system.

The following diagram details the layered architecture providing details as to how the different components (services, tools and applications) in the system interact and build on each other:

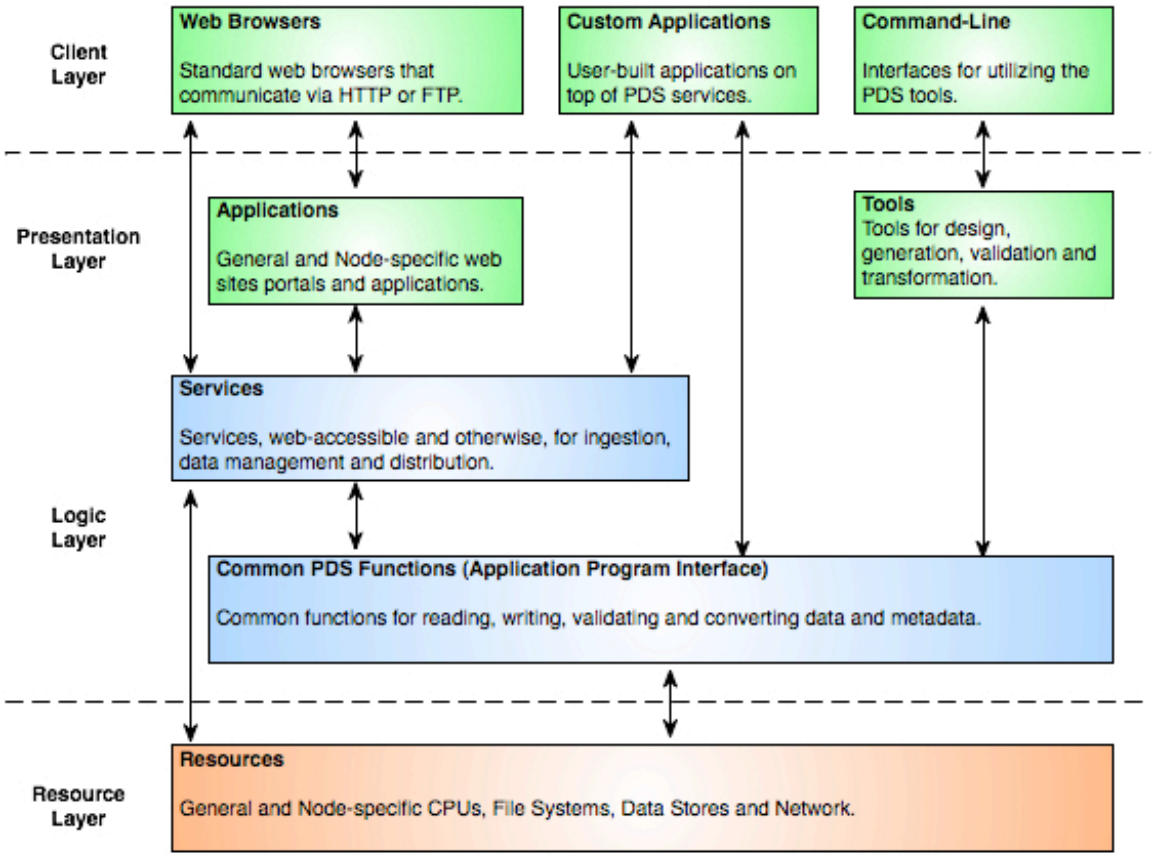

Figure 12.2: Layered Architecture (Components)

Several of the architectural principles defined in section 5 of this document pertain to technology, like Common Use Software, Technology Independence and Interoperability. In order to satisfy these principles, development of PDS 2010 will utilize certain technologies and standards where appropriate to facilitate

interfaces and reduce software development effort. The following diagram details several technologies, related standards and their application within the layers of the system:

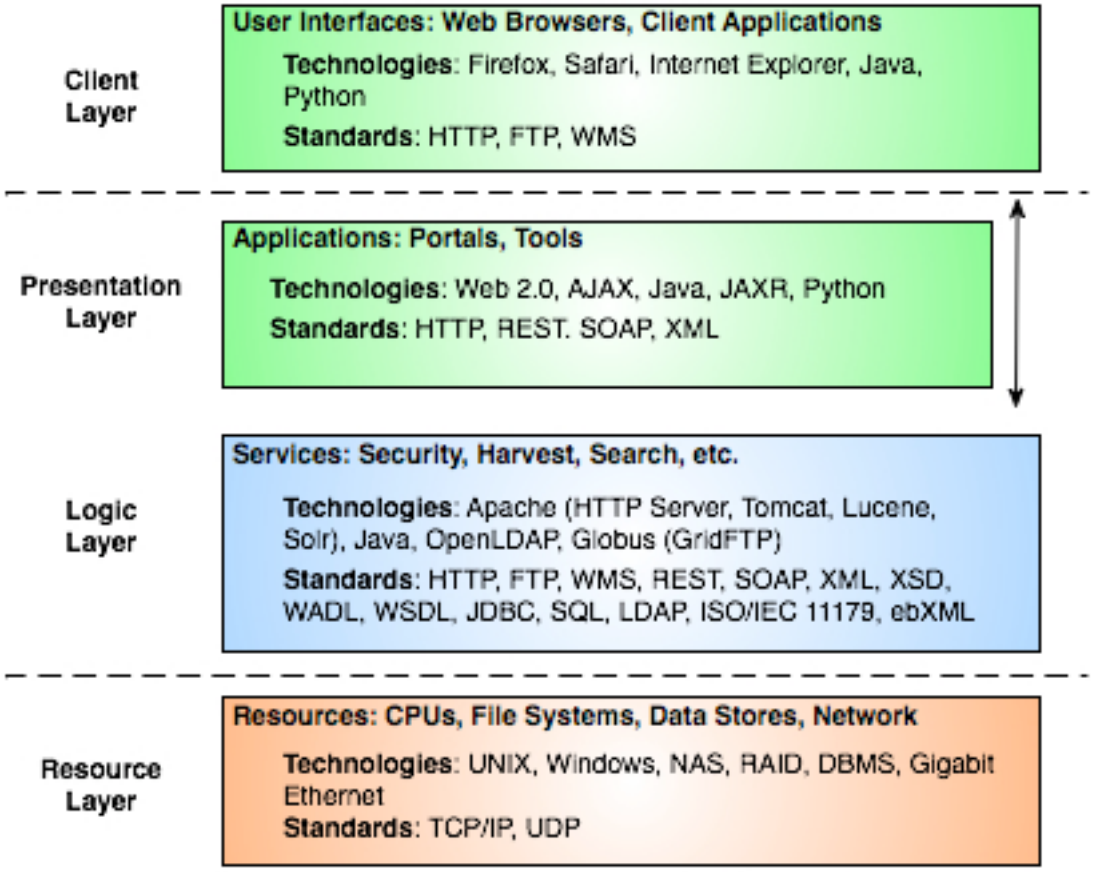

Figure 12.3: Layered Architecture (Technologies and Standards)

The technologies and standards listed in the diagram above represent some of the possible options for PDS 2010. The design phase will detail the actual options utilized in development.

13.0 REMAINING PHASES

The remaining phases of the TOGAF process will be addressed as the architecture progresses.

13.1 Opportunities and Solutions

This phase involves identification of the major implementation projects. The PDS 2010 project plan presented at the MC meeting in July 2008 and finalized with the preliminary design reviews presented at the MC meetings in August and December 2009, identifies the implementation projects. The PDS 2010 Project Plan document [14] captures the plan and serves as the controlling document for implementation of PDS 2010.

13.2 Transition and Migration Planning

This phase involves identifying the process for transitioning the current system to the PDS 2010 system as well as migrating existing data (e.g., PDS3 conformant) in the PDS system to the new PDS4 information model. The Project Plan document [14] also covers the transition plan. The same document identifies the approaches for migration but the actual plan has yet to be developed.

13.3 Implementation Governance

This phase involves reviewing and approving future implementation projects to make sure they conform to the system architecture. This is something that should be brought to the MC to determine the appropriateness for PDS.

13.4 Architecture Change Management

This phase involves the maintenance of the system architecture. See section 1.5.

APPENDIX A REFERENCES

[1] Planetary Data System Strategic Roadmap 2006 - 2016, PDS Management Council, February 2006.

[2] Planetary Data System (PDS) Level 1, 2 and 3 Requirements, August 2006.

[3] PDS4 Architecture Preliminary Recommendations for 2008 and Beyond, PDS4 Architecture Working Group, November 28, 2007.

[4] PDS4 Drivers for the PDS System Architecture, PDS4 Architecture Working Group, November 5, 2007.

[5] The Open Group Architecture Framework (TOGAF), Version 8.1.1, Enterprise Edition, April 2007.

[6] Zachman Framework for Enterprise Architecture, The Zachman Institute for Framework Advancement, (http://www.zifa.com/).

[7] Service Oriented Architecture for Dummies, J. Hurwitz, R. Bloor, C. Baroudi, 2006.

[8] Service-Oriented Architecture Definition, The Open Group, 1.1, (http://www.opengroup.org/projects/soa/doc.tpl?gdid=10632).

[9] ISO/IEC 42010 (IEEE 1471-2000), Systems and software engineering - Recommended practice for architectural description of software-intensive systems, July 15, 2007.

[10] ISO/IEC 10746, The Reference Model of Open Distributed Processing (RM-ODP).

[11] Federal Enterprise Architecture (FEA), Office of Management and Budget (OMB), (http://www.whitehouse.gov/omb/egov/a-1-fea.html).

[12] Applied Software Architecture, C. Hofmeister, R. Nord, D. Soni, 2000.

[13] PDS4 Information Model Specification, PDS4 Information Model Specification Team, Version 0.090609p, February 2, 2010.

[14] Planetary Data System (PDS) 2010 Project Plan, February 2010.

APPENDIX B ACRONYMS

The following acronyms pertain to this document:

ADM	Architecture Development Method
API	Application Programming Interface
AJAX	Asynchronous JavaScript and XML
COTS	Commercial Off-The-Shelf
DBMS	Database Management System
DN	Discipline/Data Node (PDS)
ebXML	Electronic Business using XML
EN	Engineering Node (PDS)
ESDIS	Earth Science Data and Information System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
IT	Information Technology
JAXR	Java API for XML Registries
JDBC	Java Database Connectivity
JPL	Jet Propulsion Laboratory
LDAP	Lightweight Directory Access Protocol
NAS	Network Attached Storage
NASA	National Aeronautics and Space Administration
NSSDC	National Space Science Data Center
PDS	Planetary Data System
PDS3	Version 3 of the PDS Standards
PDS4	Version 4 of the PDS Standards
RAID	Redundant Array of Independent Disks
REST	Representational State Transfer
RM-ODP	Reference Model of Open Distributed Processing
RSS	Really Simple Syndication
SAWG	System Architecture Working Group
SDSC	San Diego Supercomputing Center
SDWG	System Design Working Group
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
TB	Terabyte
TCP/IP	Transmission Control Protocol/Internet Protocol
TOGAF	The Open Group Architecture Framework
UDP	User Datagram Protocol
WADL	Web Application Description Language

WMS	Web Map Service
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XSD	XML Schema Definition