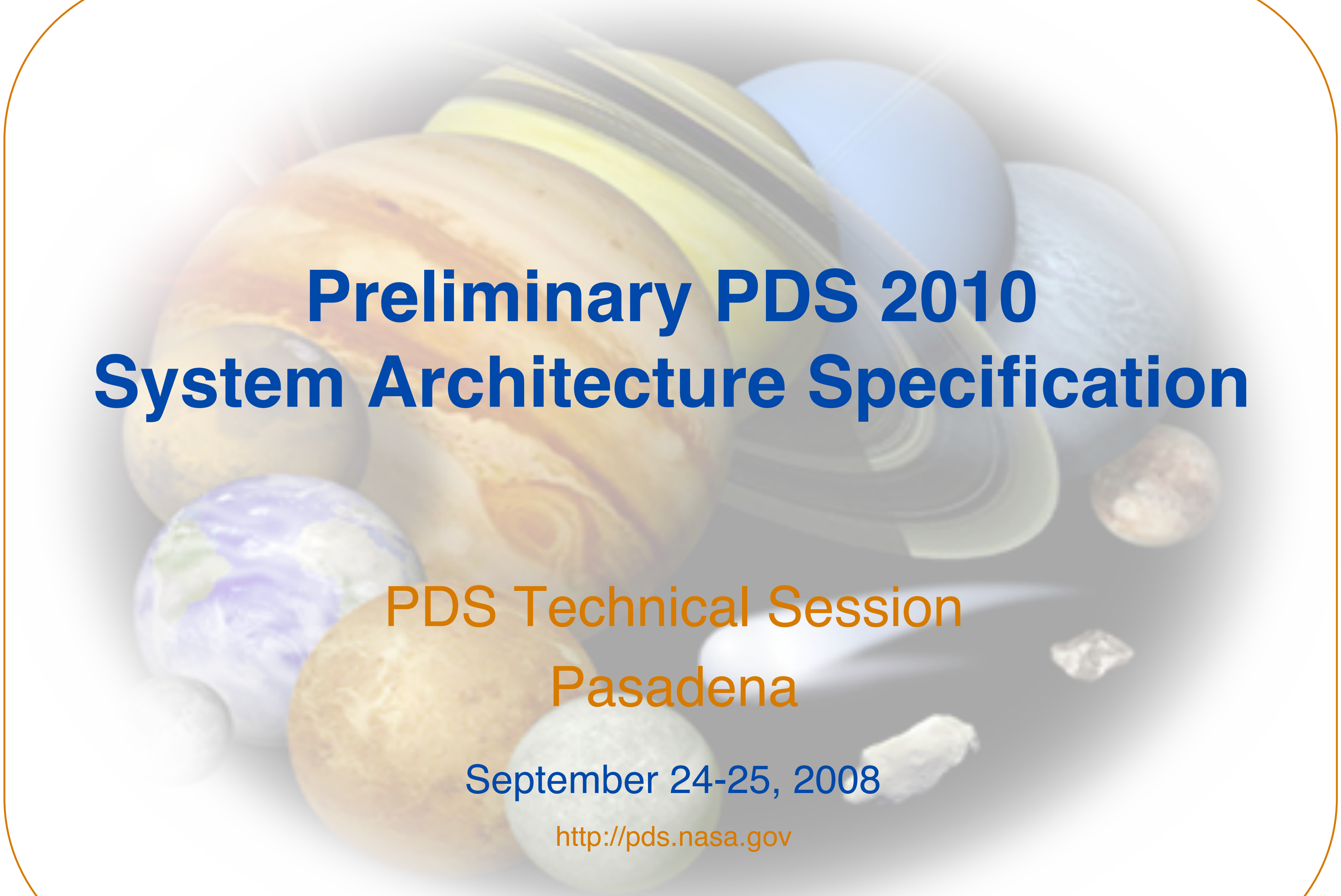


Preliminary PDS 2010 System Architecture Specification

PDS Technical Session

Pasadena

September 24-25, 2008

http://pds.nasa.gov

Topics

- Tech Session - Day 1
  - Introduction
  - Framework
  - Scope
  - Architectural Drivers
  - Architectural Principles
- Tech Session - Day 2
  - Viewpoints and Views
  - Core Architecture
  - Data Architecture
  - Application Architecture
  - Technology Architecture
  - Remaining Phases
  - References

Tech Session - Day 1

Plans, Scope and Principles

Introduction

- 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 Node as well as the Discipline Nodes.
- A System Architecture will facilitate the development of PDS-2010 via:
  - Consistent use of common terminology for ease of integration.
  - Commonly defined and implemented interfaces increase usability and portability of applications.
  - Well defined and loosely coupled services increase scalability and adaptability for future expansion.
- The end result of this effort will be a document detailing the various aspects of the PDS-2010 System Architecture.
  - This document will evolve as the system evolves.

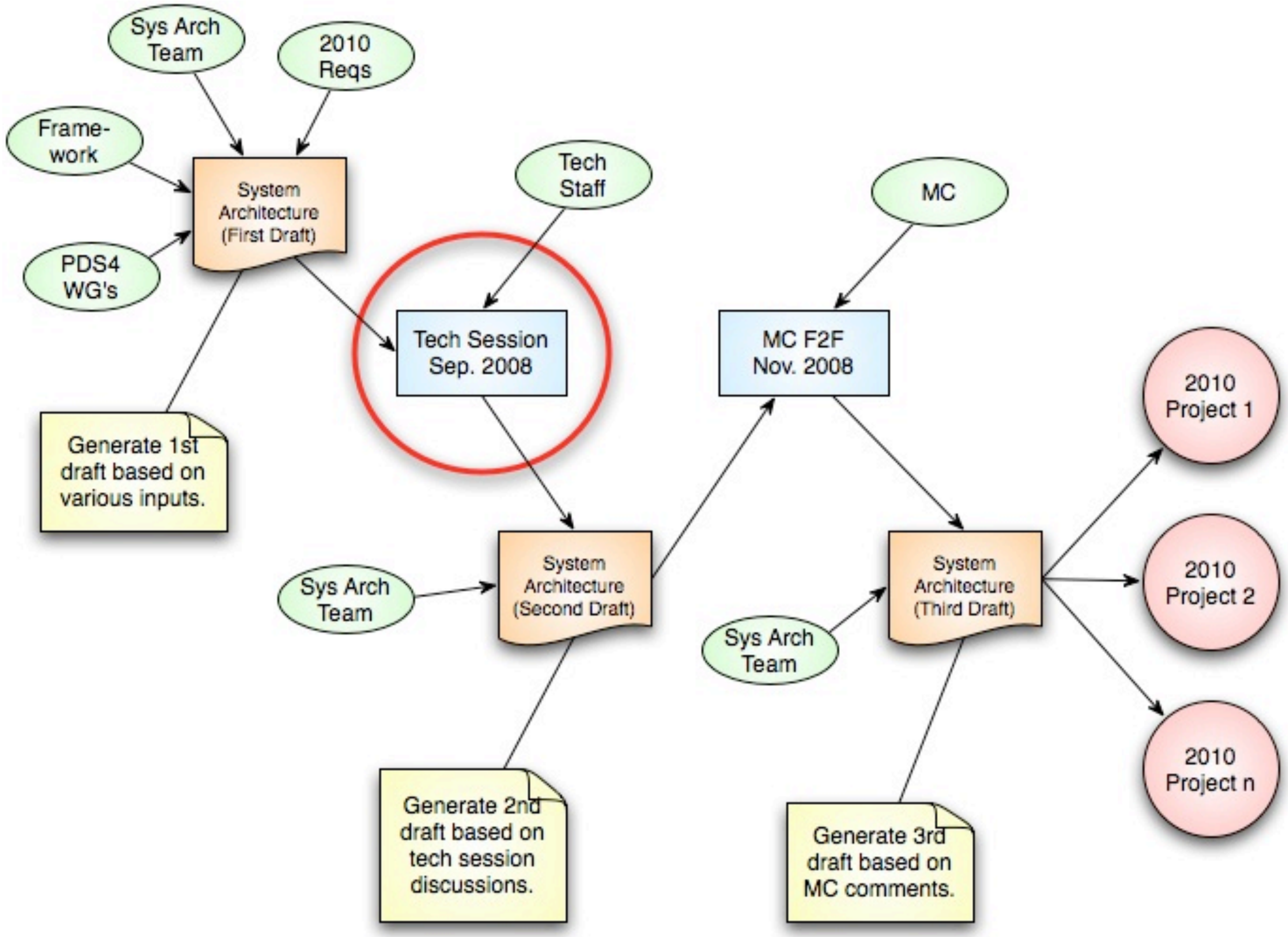
Introduction
Architecture Definitions

- 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. [1]
- 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. [2]
  - The structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time. [2]
- 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. [3]
- 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. [4]

Approach

- Formed a System Architecture Working Group (SAWG)
  - Dan Crichton, Sean Hardman, Todd King, Sue LaVoie, Mike Martin, Tom Stein
- Decided early on whether a specific industry standard approach or framework should be followed to guide this effort.
  - The Open Group Architecture Framework [2] was selected as the guiding framework.
  - More on this in the Framework section.
- Identify the artifacts that this effort should produce to move forward into development.
  - The team will determine the artifacts best suited for PDS. Still a work in progress.
- Build on the work previously produced by the PDS4 Working Groups.
  - Artifacts from the Architecture and User Support WGs have fed directly into this effort.
- Keep it simple
  - The result of this effort should not require a “System Architect” to decipher. No training will be required to move forward.

Introduction
Roadmap


Sys Arch Team
2010 Reqs
Frame-work
System Architecture (First Draft)
PDS4 WG's
Tech Staff
MC
Tech Session Sep. 2008
MC F2F Nov. 2008
Generate 1st draft based on various inputs.
2010 Project 1
Sys Arch Team
System Architecture (Second Draft)
2010 Project 2
Sys Arch Team
System Architecture (Third Draft)
2010 Project n
Generate 2nd draft based on tech session discussions.
Generate 3rd draft based on MC comments.

Framework

- The Open Group Architecture Framework (TOGAF) was selected as the framework for developing the PDS-2010 system architecture. [2]

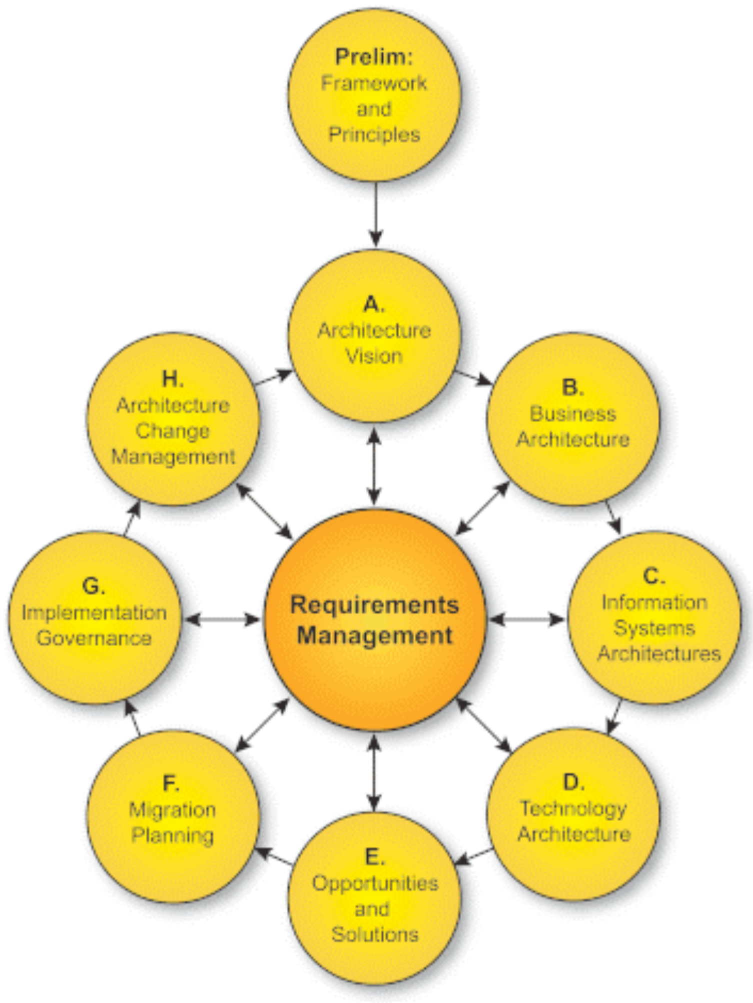
- TOGAF is essentially a tool for assisting in the acceptance, production, use and maintenance of architectures.

- It is very flexible with respect to tailoring to an organization’s unique needs. PDS has many of those.

- It is also very adaptable with regard to incorporating other frameworks and standards. These have been utilized or are being considered:
  - **Zachman Framework** [6] - Utilized for artifact identification and organization.
  - **IEEE 1471-2000** [7] - Utilized for architectural description guidelines.
  - **Reference Model for Open Distributed Processing (RM-ODP)** [8]- Being considered for the software architecture.

Framework
TOGAF Architecture Development Method

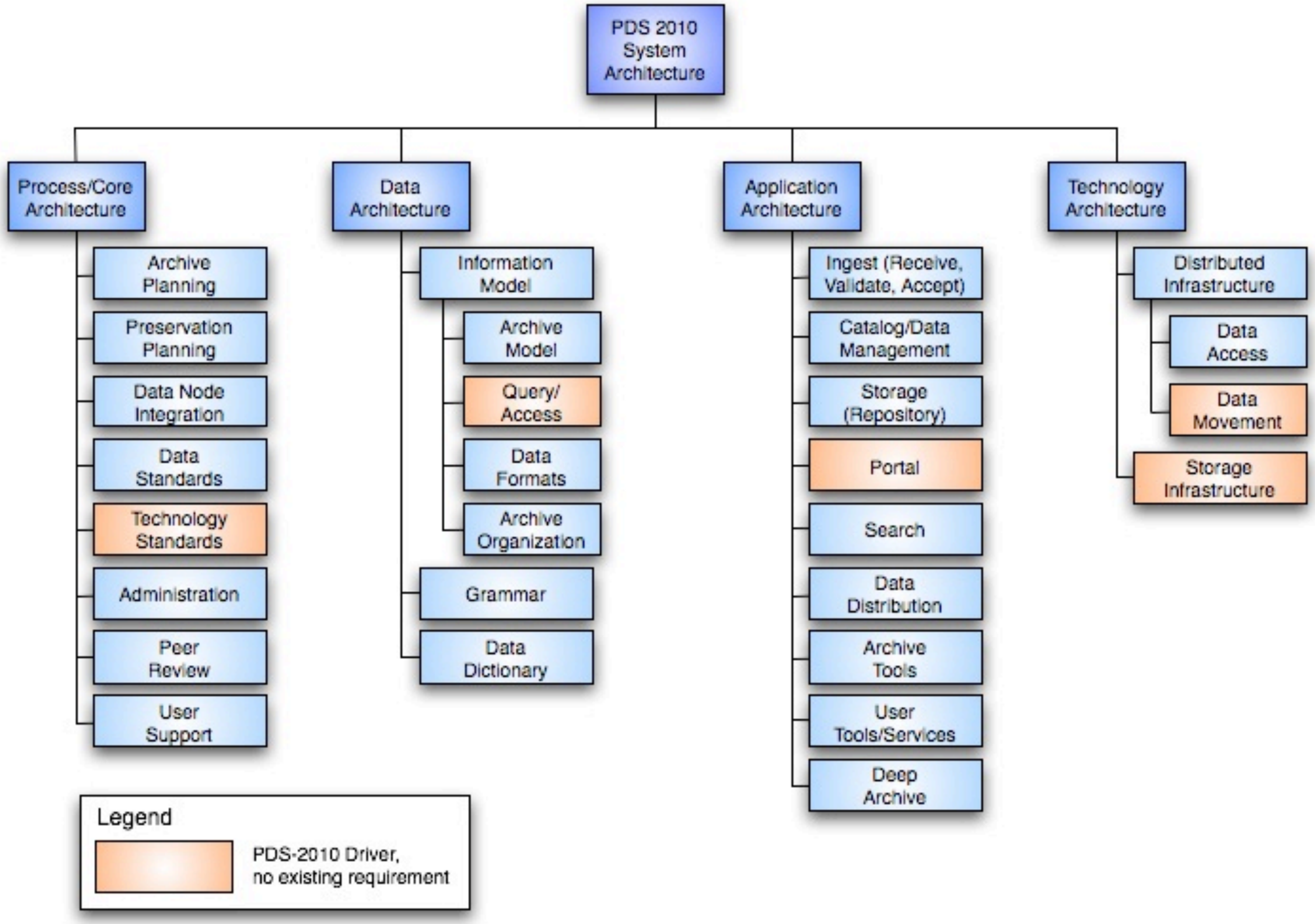
- The core of TOGAF is the Architecture Development Method (ADM).
- A step-by-step approach to develop and use an enterprise architecture.
- Each phase has a prescribed set of inputs and outputs (artifacts).
- Along with the ADM, TOGAF also contains two other sections:
  - **Enterprise Continuum** - A virtual repository of architecture assets.
  - **Resource Base** - A set of tools and techniques for applying TOGAF.


Prelim: Framework and Principles
A. Architecture Vision
B. Business Architecture
C. Information Systems Architectures
D. Technology Architecture
E. Opportunities and Solutions
F. Migration Planning
G. Implementation Governance
H. Architecture Change Management
Requirements Management

Framework
TOGAF Architecture Types

- There are four types of architecture that are commonly accepted as subsets of an overall enterprise architecture, all of which TOGAF is designed to support:
  - **Business (Process/Core) Architecture** - this defines the business strategy, governance, organization, and key business processes.
  - **Data Architecture** - this describes the structure of an organization's logical and physical data assets and data management resources.
  - **Application Architecture** - this kind of architecture provides a blueprint for the individual application systems to be deployed, their interactions, and their relationships to the core business processes of the organization.
  - **Technology Architecture** - this describes 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.

Framework
PDS System Decomposition



Scope

- By now, we have already discussed customer/driver prioritization and how we lean between consensus and distributed control regarding the data system. Here are some other scope related issues.

- Will the PDS4 information model incorporate all of the PDS operational information (e.g., deliveries, reviews, orders, housekeeping, usage statistics)?

- Will PDS-2010 impact how nodes currently conduct administrative tasks (e.g., budgeting, scheduling, calendars, status reporting)?

- Will PDS-2010 apply to sub-nodes or data nodes or might PDS 2010 be implemented just at the discipline nodes initially?

- Will PDS-2010 be enforced on existing but ongoing project interfaces or will existing projects be grandfathered?

Architectural Drivers

- Based on the available input from the roadmap, requirements and the Management Council, the PDS4 architecture working group extracted and created a categorized list of architectural drivers organized into thematic areas:

- More Data
  - PDS storage requirements are projected to increase from 40 TB to over 500 TB in just three years.
- More Complexity
  - Missions, instruments, and data are all becoming more complex.
- More Producer Interfaces
  - PDS is facing an increasing number of missions, a greater number and diversity of data providers, and smaller, focused missions.
- 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.
- Limited Funding
  - The emphasis on smaller, faster, cheaper missions which often include international partners may limit the ability to provide products suitable for analysis by the broader science community.
- Creating a “System” from the Federation
  - The current PDS nodes operate autonomously and independently with limited distributed access via PDS-D to node repositories.

Architectural Principles

- Architectural principles are often used to form a general basis for decision making of architectural choices for a system.
- The principles detailed here are based on the architectural drivers and the initial set of principles identified by the PDS4 Architecture Working Group [9], along with influences from the TOGAF example principles.
- The principles are intended to be used as a guide to drive the design and development of PDS 2010.
- 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.

Architectural Principles
Process/Core Architecture

- Data Stewardship
  - PDS will manage NASA-related planetary data in order to maintain its usability, accessibility, integrity and quality.
- Reliability
  - PDS operations are maintained in spite of system interruptions.
- Common Use Software
  - Development and integration of system-wide software (applications and APIs) is preferred over localized development of duplicative functionality.

Architectural Principles
Process/Core Architecture - 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." [10]
- Implications:
  - PDS will collaborate with data providers as early as possible in the data creation process to ensure that PDS Standards and tools are adopted and utilized effectively.
  - Data will be preserved for the long-term. Although the definition of long-term is up for debate, it can be measured in terms of at least 10s of years.
  - Data integrity must be maintained. In order to accomplish this, the PDS must adhere to a rigorous data integrity policy to ensure its data are reliable.
  - Data will be managed 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.
  - Data will not be released to the public until that data has successfully passed a peer review.

Architectural Principles
Process/Core Architecture - Reliability

- Statement: PDS operations are maintained 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. Hardware failures, natural disasters and data corruption should not disrupt nominal operations.

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

Architectural Principles

Process/Core Architecture - 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 and/or integration of off-the-shelf common software will allow PDS to maximize the benefit of their software development resources as well as to control the technical diversity of the system 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 data relocation and change of ownership.

Architectural Principles
Data Architecture

- Model Driven
  - The PDS will design and maintain a conceptual information model that is implementation independent.
- Common Vocabulary and Data Definitions
  - Data are described consistently throughout the system, and the definitions are understandable and available to all users.
- Data are an Asset
  - PDS data are an asset that has value to NASA and the larger Planetary Science Community and is managed accordingly.
- Data are Accessible
  - Data are accessible for users to perform their functions, regardless of where the data are located.

Architectural Principles
Data Architecture - 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:
  - The information model will be defined using a formal data modeling notation.
  - All data-related models will be derived from the conceptual information model.
  - Software developed for the system should be designed to evolve as the information model evolves.

Architectural Principles
Data Architecture - 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 or data dictionary is an essential component for generating and maintaining quality metadata. It also aides in the understanding of the data from the users perspective.

- Implications:
  - A data dictionary must be established and utilized uniformly throughout PDS.
  - Additions or modifications to the data dictionary will be managed via a change control board.

Architectural Principles
Data Architecture - Data are an Asset

- Statement: PDS data are an asset that has value to NASA and the larger Planetary Science Community and is 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 is closely related 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 safe-keeping 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.

Architectural Principles
Data Architecture - 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.

Architectural Principles
Application Architecture

- Modular Development
  - The system is decomposed into manageable elements and components in order to facilitate phased development and deployment.
- Technology Independence
  - Software is independent of specific technology choices and therefore can operate on a variety of technology platforms.
- Scalability
  - The system is scalable to accommodate increased data volume and complexity.
- Ease-of-Use
  - Software is easy to use.

Architectural Principles
Application Architecture - 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 decomposition of the system will be captured in the system architecture.
  - Components of the system will be designed to minimize inter-component dependencies.
  - Commonly used components will be given a higher priority in the development schedule.

Architectural Principles

Application Architecture - 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:
  - Software should be designed and developed 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.
  - Services should be designed and developed using middleware. A middleware layer can offer a generic software interface to one of those unavoidable technologies. So, in the future if the technology is replaced, only the middleware software will require modification.
  - Application Program Interfaces (APIs) will need to be developed in order to accommodate or provide a pathway for legacy applications to integrate with the new architecture.

Architectural Principles
Application 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 data 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.

Architectural Principles
Application Architecture - 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:
  - Software should be developed with an intuitive interface.
  - Applications and APIs should be well documented not only for the PDS users but also for the PDS developers.

Architectural Principles
Technology Architecture

- Requirements-Based Change
  - Modifications to the system (both software and technology) are only performed in response to new or modified requirements levied by the Management Council.
- Interoperability
  - Software and hardware should conform to appropriate standards that promote interoperability for data, applications, and technology.

Architectural Principles
Technology Architecture - Requirements-Based Change

- Statement: Modifications to the system (both software and technology) are only performed in response to new or modified requirements levied 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.
  - System software will be managed under a change management process.

Architectural Principles

Technology Architecture - 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.

Tech Session - Day 2

Views and Models

Core Services and Definitions

Migration Planning

Viewpoints and Views

- Viewpoint
  - A form of abstraction achieved using a selected set of architectural constructs and structuring rules, in order to focus on particular concerns within a system. [8]
  - Usually associated with a stakeholder of the system.
- View
  - A representation or description of the entire system from a single perspective. [7]
  - Can be represented by multiple models or descriptions.
- The viewpoints mapped to the Zachman Framework:
  - Scope (Contextual)
  - Project (Conceptual)
  - System (Logical)
  - Technology (Physical)
  - Detailed Representation (Definition)
  - Operations (Instance)

Viewpoints and Views
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:
  - Are the project-level requirements satisfied by the architecture and design of the system?
  - 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?

Viewpoints and Views
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?

Viewpoints and Views
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:
  - Are the requirements levied on the system, addressed by the architecture and design of 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?

Viewpoints and Views
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/uses 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?

Viewpoints and Views
Detailed Representation (Definition)

- Stakeholder: Data Provider
  - This stakeholder represents the mission, instrument team and NASA-funded researcher who is 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?

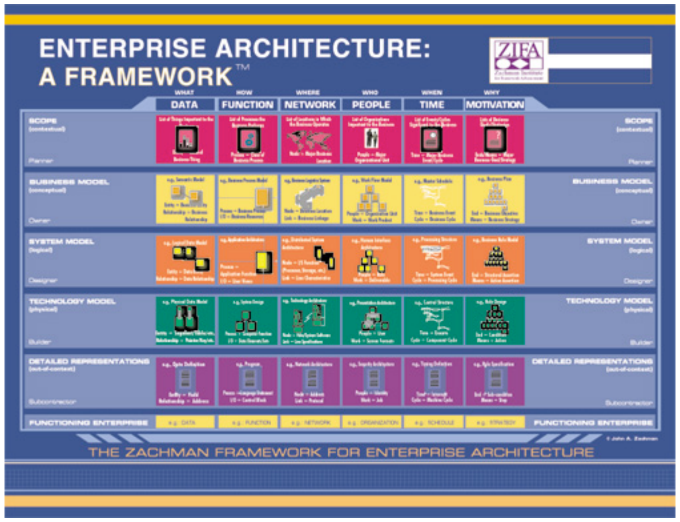
Viewpoints and Views
Operations (Instance)

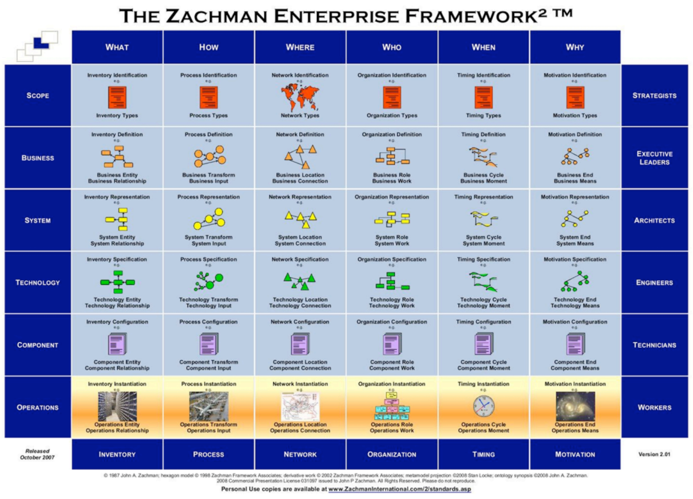
- Stakeholder: Data Consumer
  - This stakeholder represents the Planetary Scientist, which includes those experienced with solar system exploration missions and those who are mission-naïve. They include graduate students.
  - While the PDS is a NASA-funded program, many of its planetary missions are 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 the data be transformed into a format that is useful for my research?
  - Does PDS offer support and expertise for analyzing the data?

Viewpoints and Views
Zachman Framework

- Mapping the planned views to the Zachman Framework [6], provides a succinct representation.
- The framework details six aspects of the enterprise that can be described or modeled (What, How, Where, Who When and Why).
- We have focused on three of them (What, How and Where).
- The Why aspect is certainly relevant, but is covered by the Architectural Drivers and Principles.





THE ZACHMAN ENTERPRISE FRAMEWORK² ™

	WHAT	HOW	WHERE	WHO	WHEN	WHY	
SCOPE	Inventory Identification e.g. Inventory Types	Process Identification e.g. Process Types	Network Identification e.g. Network Types	Organization Identification e.g. Organization Types	Timing Identification e.g. Timing Types	Motivation Identification e.g. Motivation Types	STRATEGISTS
BUSINESS	Inventory Definition e.g. Business Entity Business Relationship	Process Definition e.g. Business Transform Business Input	Network Definition e.g. Business Location Business Connection	Organization Definition e.g. Business Role Business Work	Timing Definition e.g. Business Cycle Business Moment	Motivation Definition e.g. Business End Business Means	EXECUTIVE LEADERS
SYSTEM	Inventory Representation e.g. System Entity System Relationship	Process Representation e.g. System Transform System Input	Network Representation e.g. System Location System Connection	Organization Representation e.g. System Role System Work	Timing Representation e.g. System Cycle System Moment	Motivation Representation e.g. System End System Means	ARCHITECTS
TECHNOLOGY	Inventory Specification e.g. Technology Entity Technology Relationship	Process Specification e.g. Technology Transform Technology Input	Network Specification e.g. Technology Location Technology Connection	Organization Specification e.g. Technology Role Technology Work	Timing Specification e.g. Technology Cycle Technology Moment	Motivation Specification e.g. Technology End Technology Means	ENGINEERS
COMPONENT	Inventory Configuration e.g. Component Entity Component Relationship	Process Configuration e.g. Component Transform Component Input	Network Configuration e.g. Component Location Component Connection	Organization Configuration e.g. Component Role Component Work	Timing Configuration e.g. Component Cycle Component Moment	Motivation Configuration e.g. Component End Component Means	TECHNICIANS
OPERATIONS	Inventory Instantiation e.g. Operations Entity Operations Relationship	Process Instantiation e.g. Operations Transform Operations Input	Network Instantiation e.g. Operations Location Operations Connection	Organization Instantiation e.g. Operations Role Operations Work	Timing Instantiation e.g. Operations Cycle Operations Moment	Motivation Instantiation e.g. Operations End Operations Means	WORKERS
Released October 2007	INVENTORY	PROCESS	NETWORK	ORGANIZATION	TIMING	MOTIVATION	Version 2.01

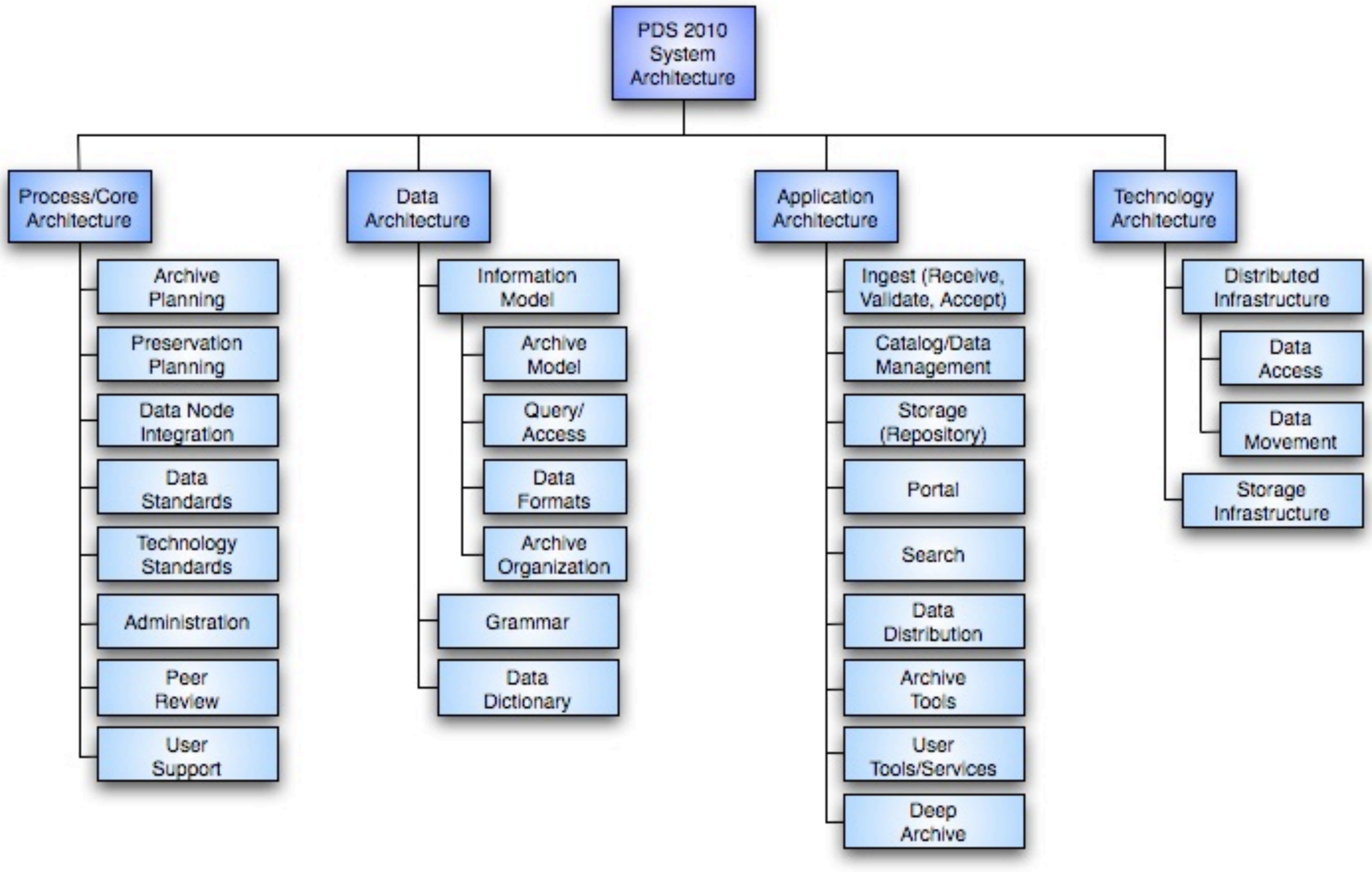
© 1987 John A. Zachman; hexagon model © 1998 Zachman Framework Associates; derivative work © 2002 Zachman Framework Associates; metamodel projection ©2008 Stan Locke; ontology synopsis ©2008 John A. Zachman. 2008 Commercial Presentation License 031097 issued to John P Zachman. All Rights Reserved. Please do not reproduce.
Personal Use copies are available at www.ZachmanInternational.com/2/standards.asp

Viewpoints and Views
PDS/Zachman Mapping

	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)

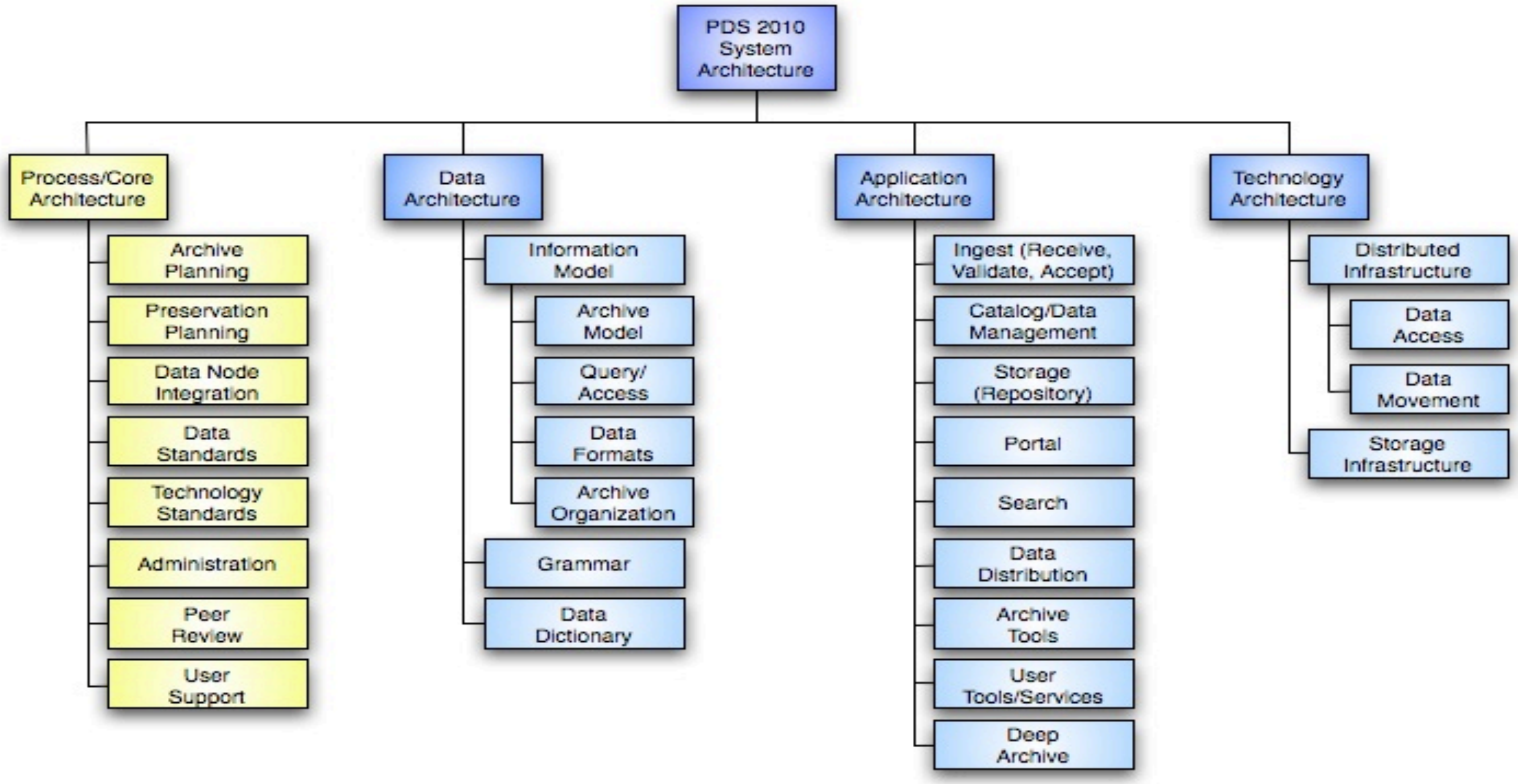
The colors have been muted at the request of the Data Architecture Working Group. ;-)

Architecture Types



Process/Core Architecture

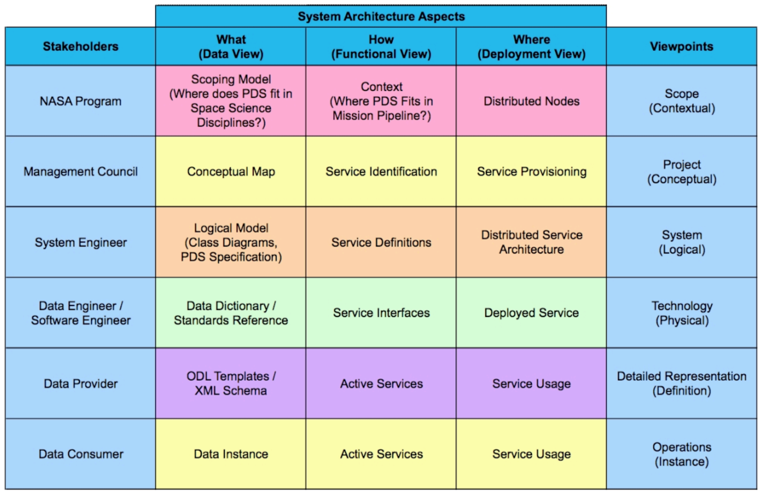
Defines the business strategy, governance, organization, and key business processes.



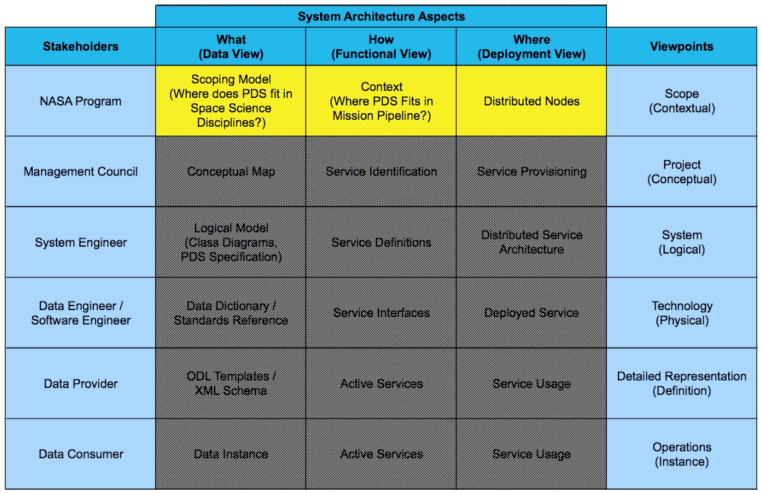
Process/Core Architecture
Plans

- Most of the elements that have been allocated to this portion are related to process and policy.

- This portion of the architecture will be addressed as needed to complement the other portions of the architecture.



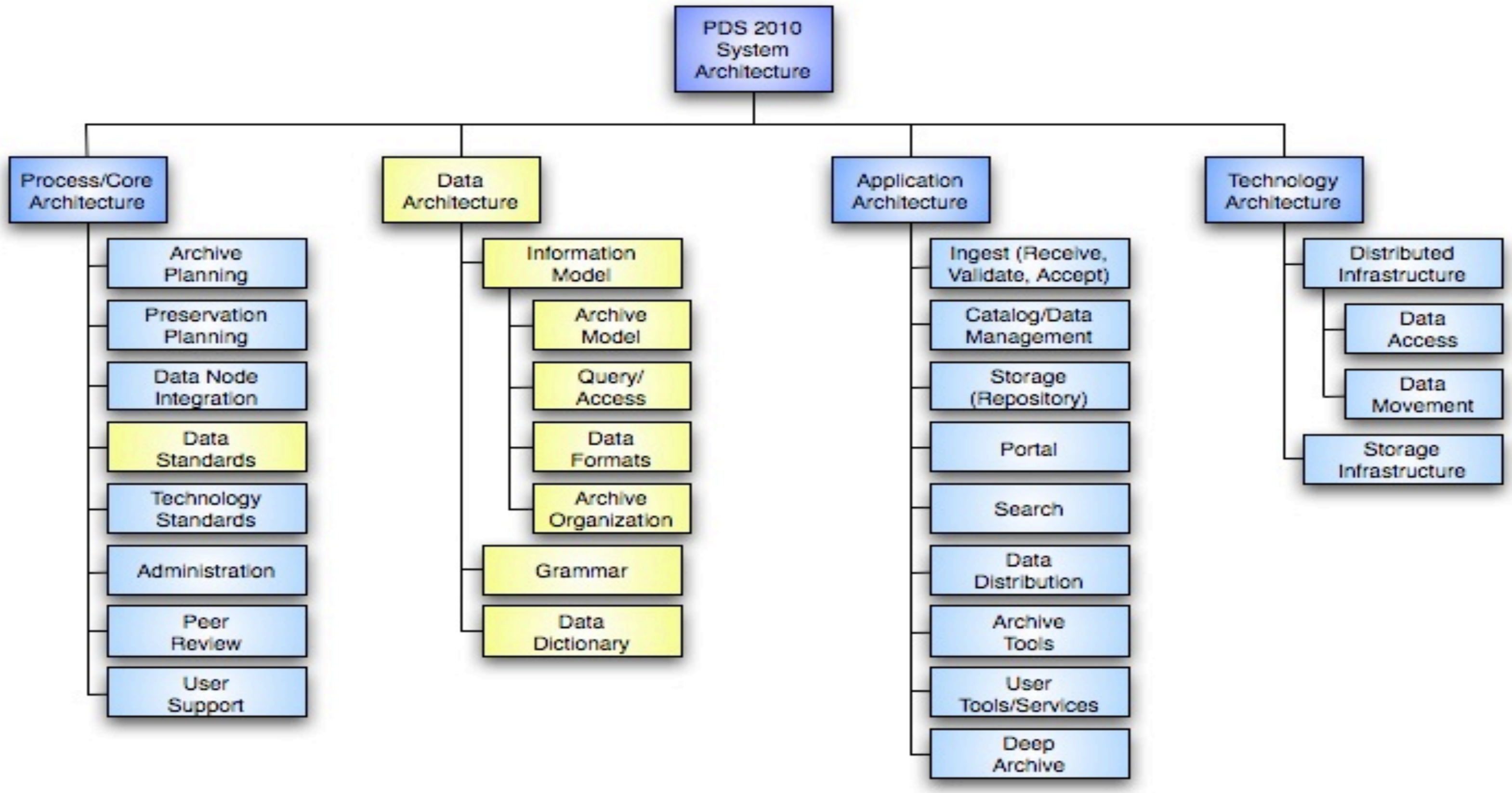
	System Architecture Aspects			
Stakeholders	What (Data View)	How (Functional View)	Where (Deployment View)	Viewpoints
NASA Program	Scoping Model (Where does PDS fit in Space Science Disciplines?)	Context (Where PDS Fits in Mission Pipeline?)	Distributed Nodes	Scope (Contextual)
Management Council	Conceptual 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)



	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)

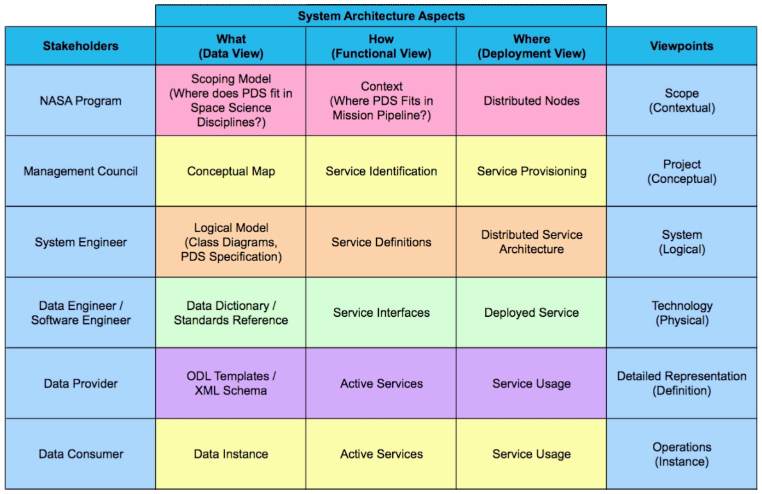
Data Architecture

Describes the structure of an organization's logical and physical data assets and data management resources.

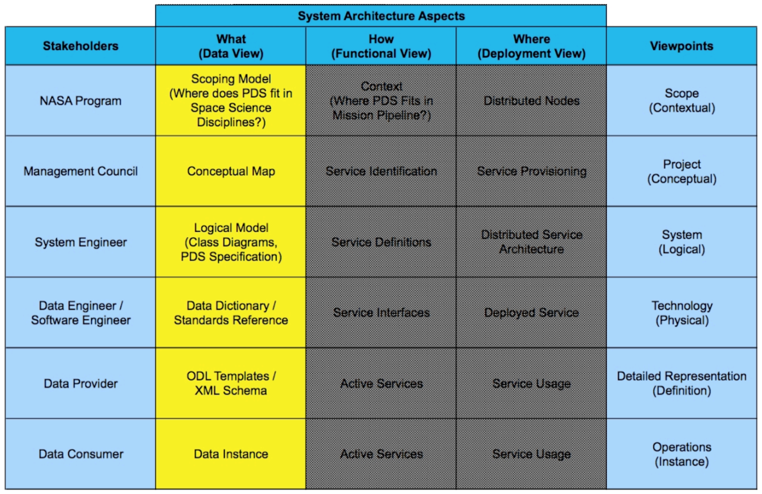


Data Architecture
Plans

- The bulk of this effort will be addressed by the Data Architecture Working Group.



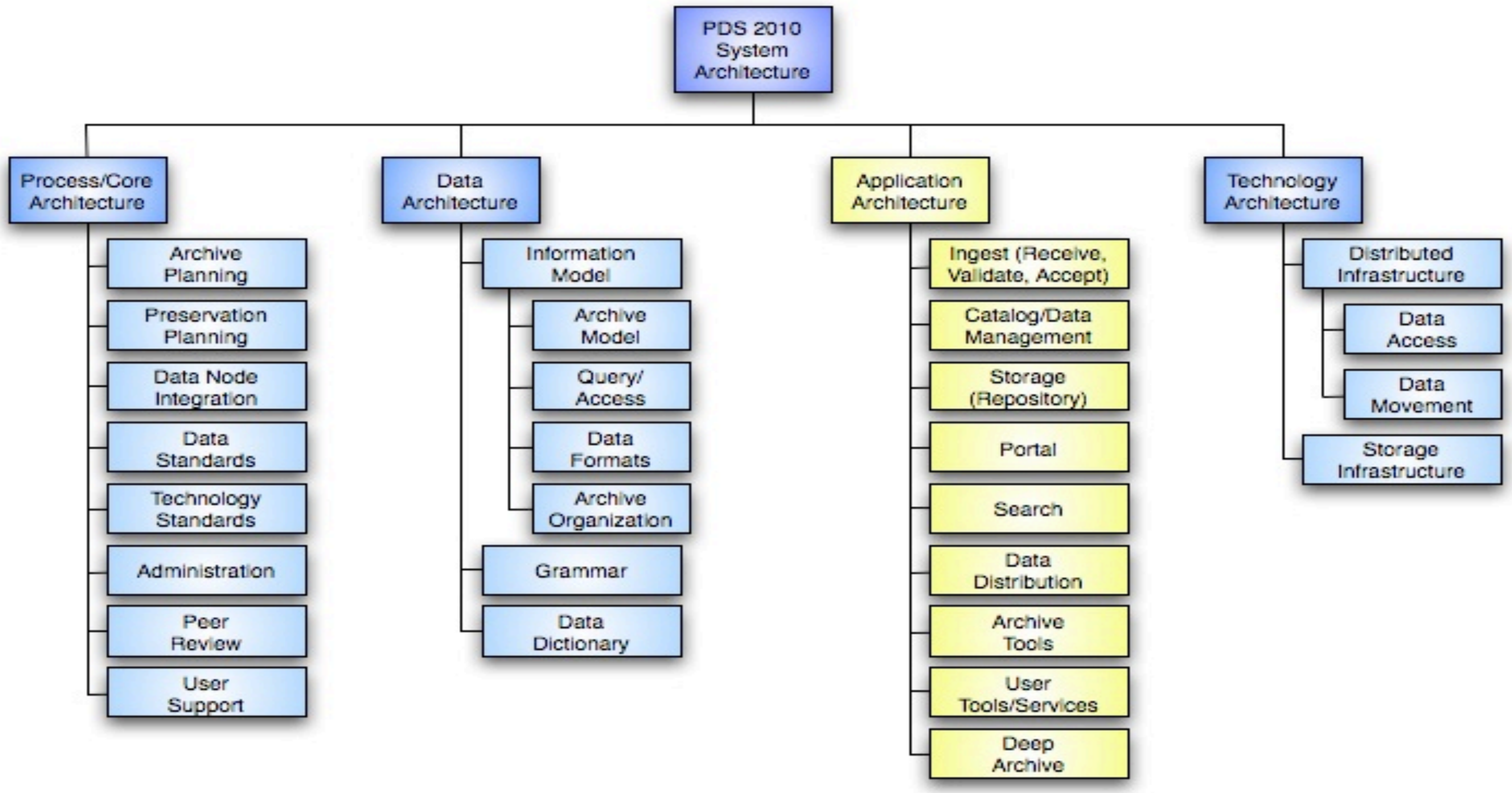
Stakeholders	What (Data View)	How (Functional View)	Where (Deployment View)	Viewpoints
	System Architecture Aspects			
NASA Program	Scoping Model (Where does PDS fit in Space Science Disciplines?)	Context (Where PDS Fits in Mission Pipeline?)	Distributed Nodes	Scope (Contextual)
Management Council	Conceptual 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)



Stakeholders	What (Data View)	How (Functional View)	Where (Deployment View)	Viewpoints
	System Architecture Aspects			
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)

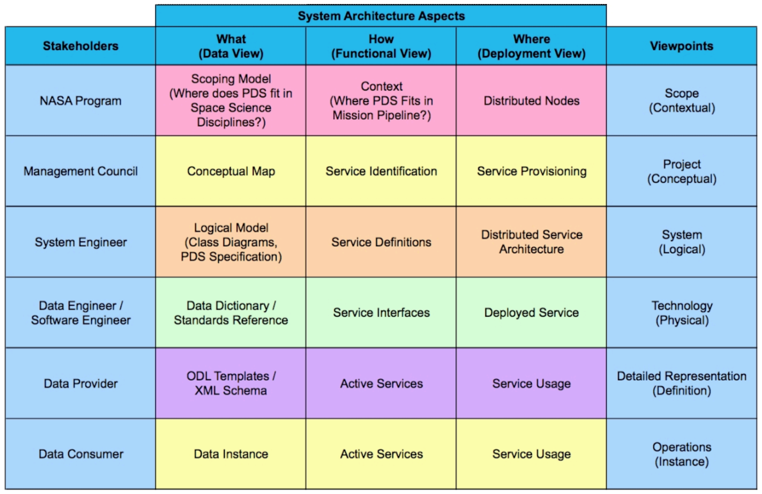
Application Architecture

Provides a blueprint for the individual application systems to be deployed, their interactions, and their relationships to the core business processes of the organization.

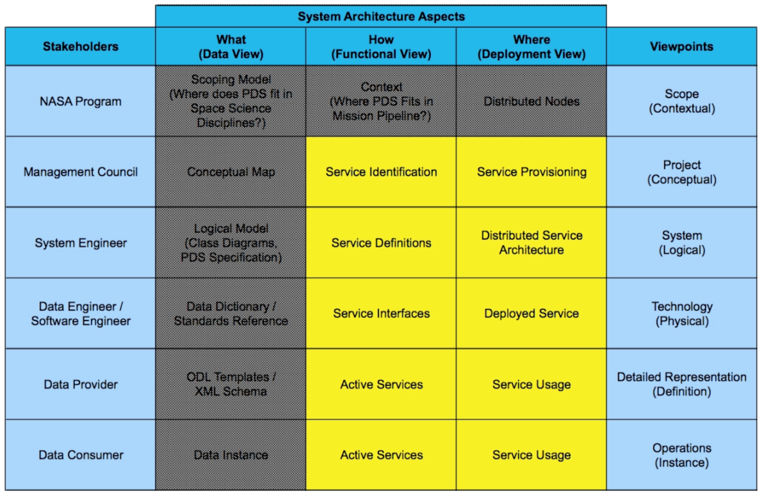


Application Architecture
Plans

- Define the software services to be developed for PDS-2010 along with their interfaces and any dependencies.
  - The service definitions will be a great starting point for Level 4 and 5 requirements.
- Provision the services.
  - Determine which services should have common implementations versus Node-specific implementations.
- Map those services to one of the PDS-2010 projects.
  - The services should be prioritized based on importance to the overall function of the system and any dependencies from other services.

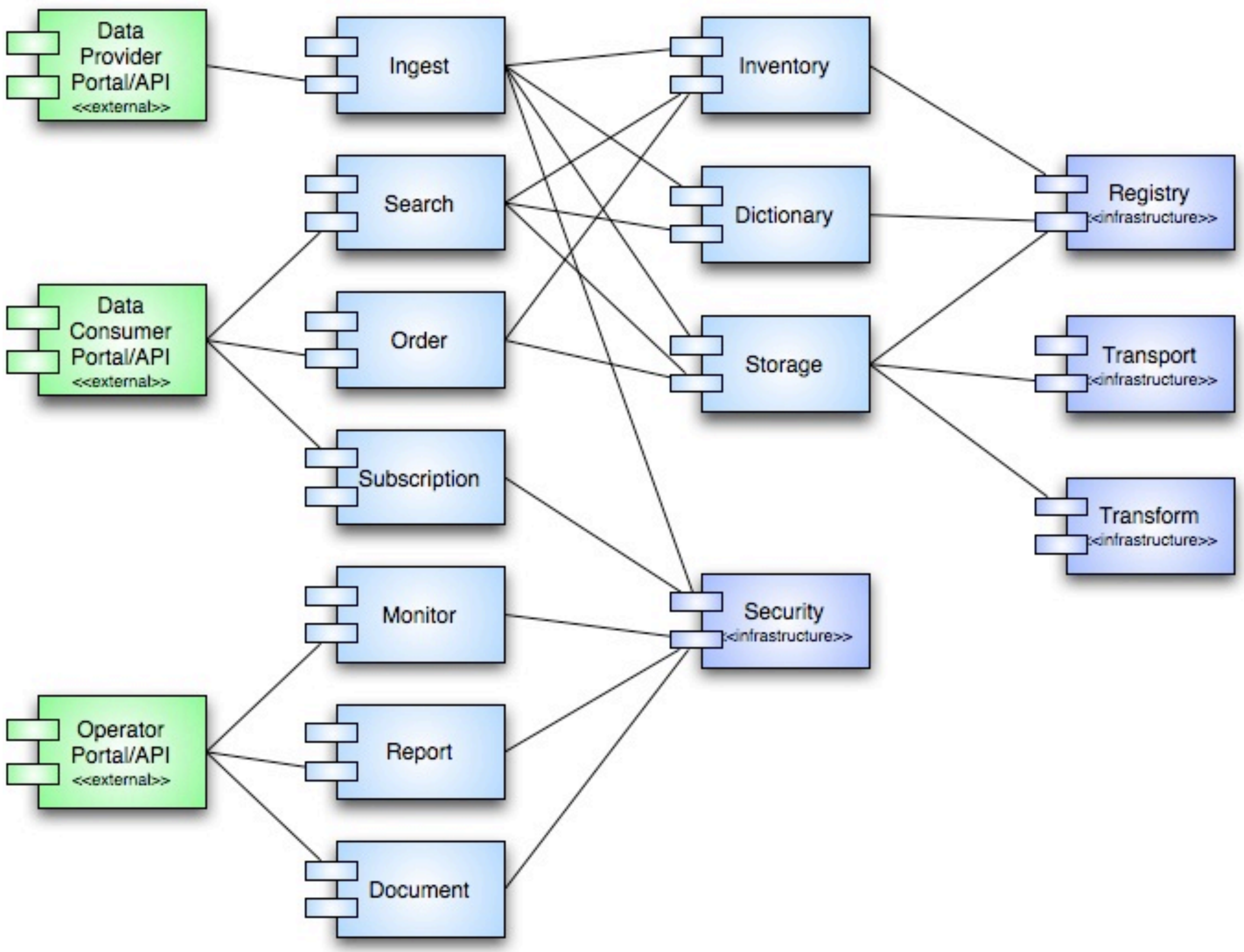


	System Architecture Aspects			
Stakeholders	What (Data View)	How (Functional View)	Where (Deployment View)	Viewpoints
NASA Program	Scoping Model (Where does PDS fit in Space Science Disciplines?)	Context (Where PDS Fits in Mission Pipeline?)	Distributed Nodes	Scope (Contextual)
Management Council	Conceptual 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)



	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)

Application Architecture
Services



Application Architecture
Dictionary Service

- This service provides functionality for managing data dictionaries, schemas and their elements.
- Also encompasses configuration management of said items and online access for use in validation.
- A commonly used standard for such registries is ISO/IEC 11179.

Application Architecture
Document Service

- This service provides functionality for managing documents (e.g., APG, PAG, etc.).

- This would most likely be satisfied by a Commercial Off-The-Shelf (COTS) product.

Application Architecture
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.
- The service acts the interface to the Inventory and Storage services for the data provider.

Application Architecture
Inventory Service

- This service provides functionality for managing the metadata repository.
- This includes insertion of metadata from the Ingest Service as well as responding to queries from the Search Service.
- The service would encompass Data Set and Data Product metadata.
  - It will likely consist of two distinct components, one for managing Data Set metadata and the other for managing Data Product metadata. The Data Set component could continue to be centralized while the Data Product component could be distributed at the Nodes.

Application Architecture
Monitor Service

- This service provides functionality for monitoring service status across the system.

Application Architecture
Order Service

- This service provides functionality for ordering data.

Application Architecture
Registry Service

- This service 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 could also include functionality for generating unique permanent identifiers for identifiable objects.

- There are a number of standards associated with registry services including:
  - The Universal Description Discovery and Integration (UDDI) specification defines a way to publish and discover information about Web services.
  - ebXML (Electronic Business using eXtensible Markup Language)
  - ISO/IEC 11179 Metadata Registry (MDR) standard

Application Architecture
Report Service

- This service provides functionality for capturing and reporting metrics.

Application Architecture
Search Service

- This service provides functionality for accepting queries from data consumers for Data Set and Data Product resources.
- Also includes functionality for retrieving search results.
- This service acts as the interface to the Inventory and Storage Services for the data consumer.

Application Architecture
Security 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).

Application Architecture
Storage 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 to the deep archive (e.g., NSSDC, SDSC) or other external repositories.

Application Architecture
Subscription 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).

Application Architecture
Transform Service

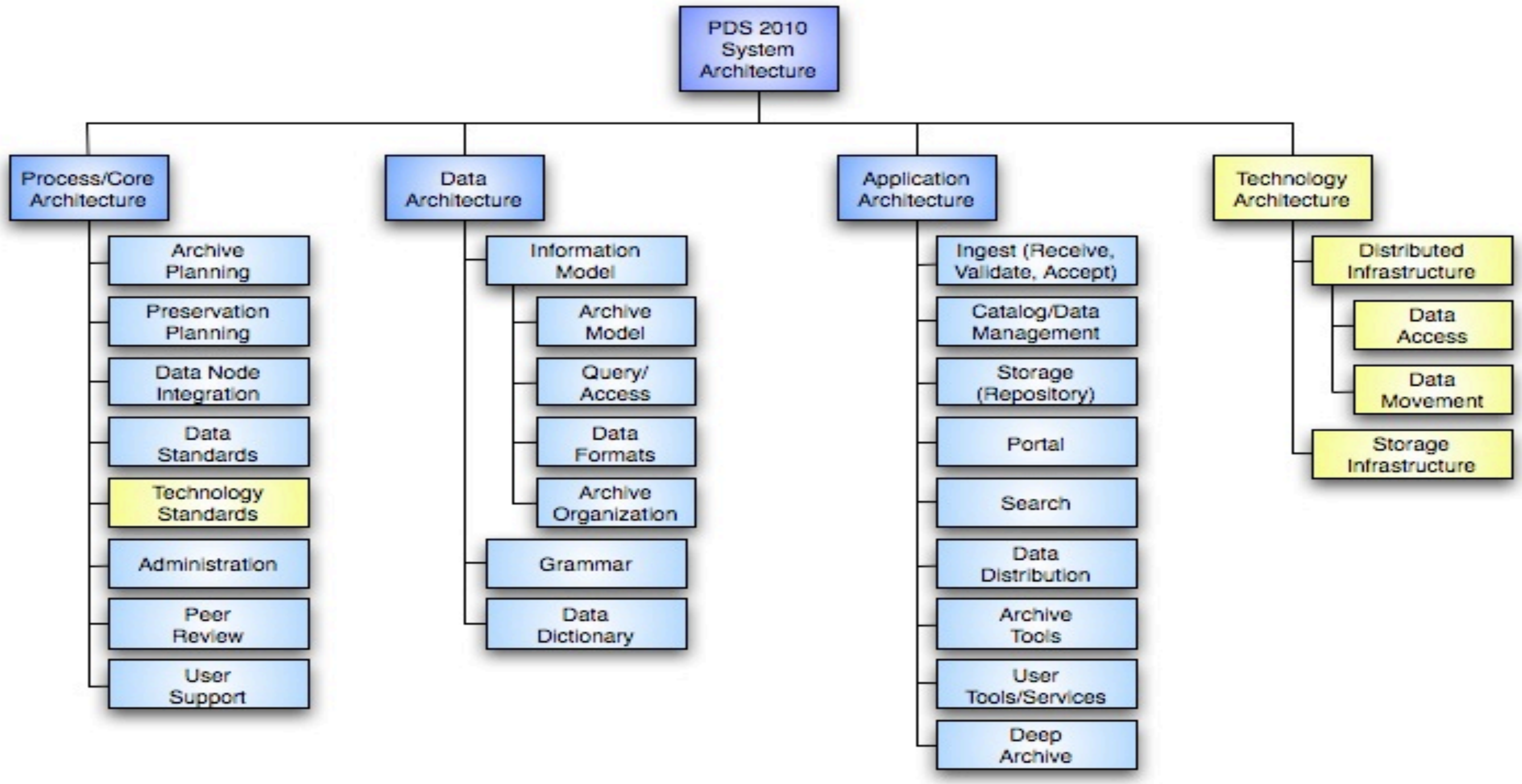
- Provides functionality for file format conversion, coordinate transformation, subsetting and packaging of results from Data Product queries.

Application Architecture
Transport Service

- Provides functionality for moving data (either from search results or to the deep archive) across the network.
- This service could be utilized for returning search results to data consumers or for pushing data to the deep archive.

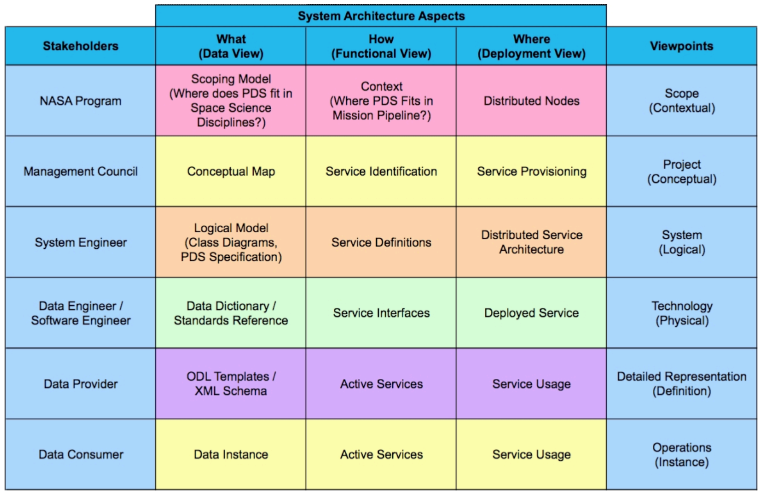
Technology Architecture

Describes 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.

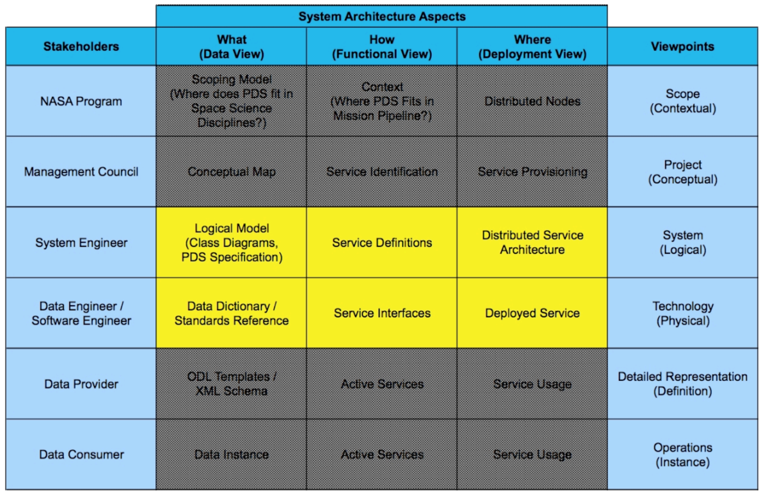


Technology Architecture
Plans

- Determine technology options for a Service Oriented Architecture (SOA) approach to be used in development.
  - Decisions made when defining the system scope and constraints will have an impact here. For example, any platform or programming language limitations.



	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)



	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)

Remaining Phases

- Opportunities and Solutions
  - This phase involves identification of the major implementation projects.
  - Much of this has already been accomplished with the PDS-2010 project plan presented at the last MC meeting.
  - That plan should be reconciled with the outputs from the previous phases to make sure they are in sync.
- Migration Planning
  - This phase involves creating an implementation roadmap.
- 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.
- Architecture Change Management
  - This phase involves the maintenance of the system architecture.

References

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

[2] The Open Group Architecture Framework (TOGAF), The Open Group, 2007. (http://www.opengroup.org/)

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

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

[5] PDS-2010 Project Planning, PDS Planning and Assessment WG, April 3, 2008. (http://pds-engineering/projects/PDS4/pds2010-project-overview-20080401.pdf)

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

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

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

[9] PDS4 Architecture Preliminary Recommendations for 2008 and Beyond (Draft), PDS4 Architecture Working Group, November 28, 2007.

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

Questions/Comments