



National Aeronautics and
Space Administration

PDS 4 Data Architecture

Developmental Approach

PDS 4 Data Architecture Team

September 2008

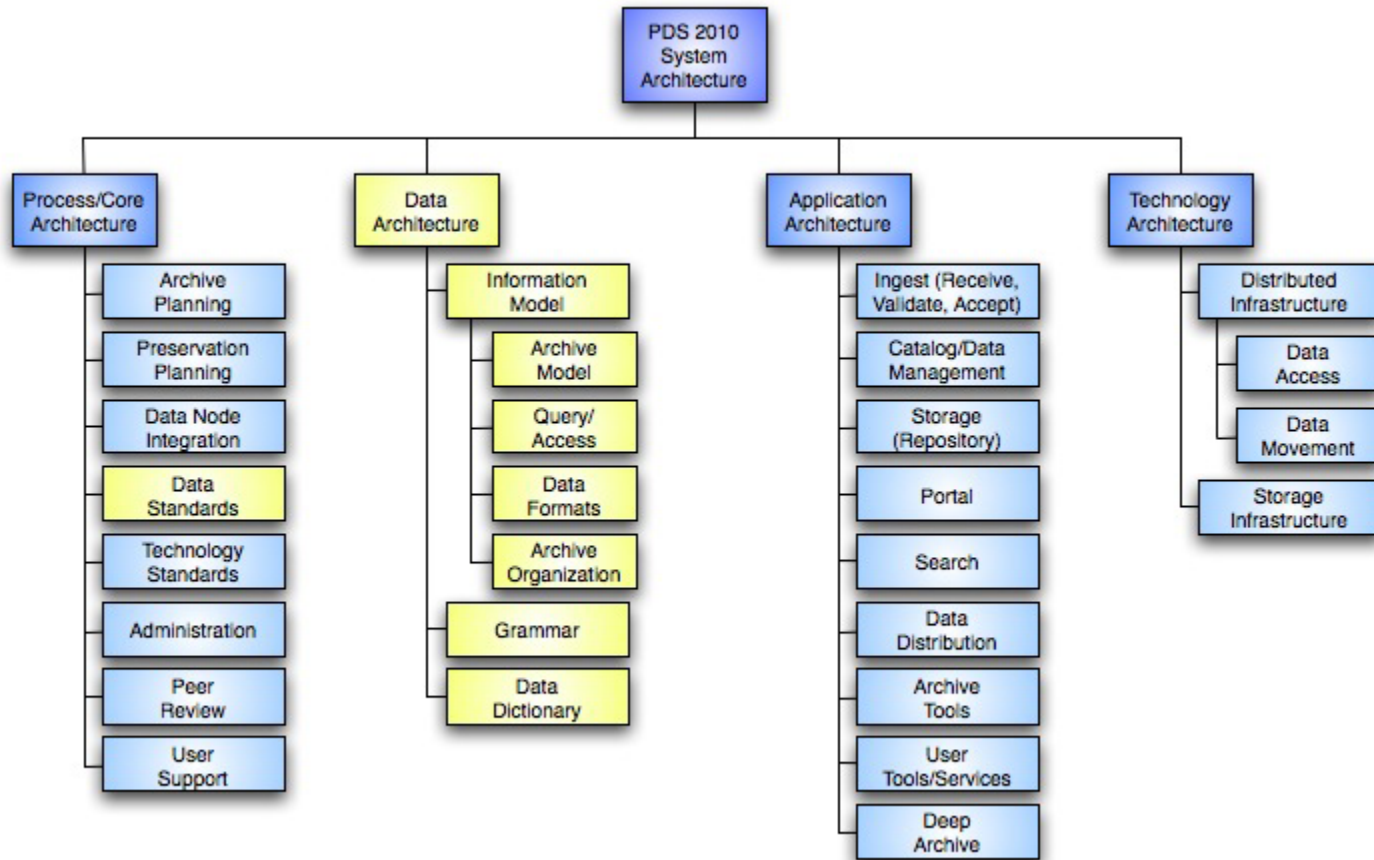


Topic Discussions

- **Data Architecture Scope**
- **Data Models**
- **Information Model Development Phases**



Data Architecture Scope





What is a Data Model?

- Everything that isn't the real thing is a *model*.
- A *data* model is an abstract model that describes how data is represented and accessed.
- *Data modeling* is the process of creating a data model instance by applying a data model theory, typically to meet a set of requirements.
- A major component of data modeling is to visually represent the rules that the community wishes to enforce on data.
- An *information model* is a set of related data models.
 - E.g. Product vs Operational; Conceptual versus Logical



Model for each Viewpoint

Models

Contextual

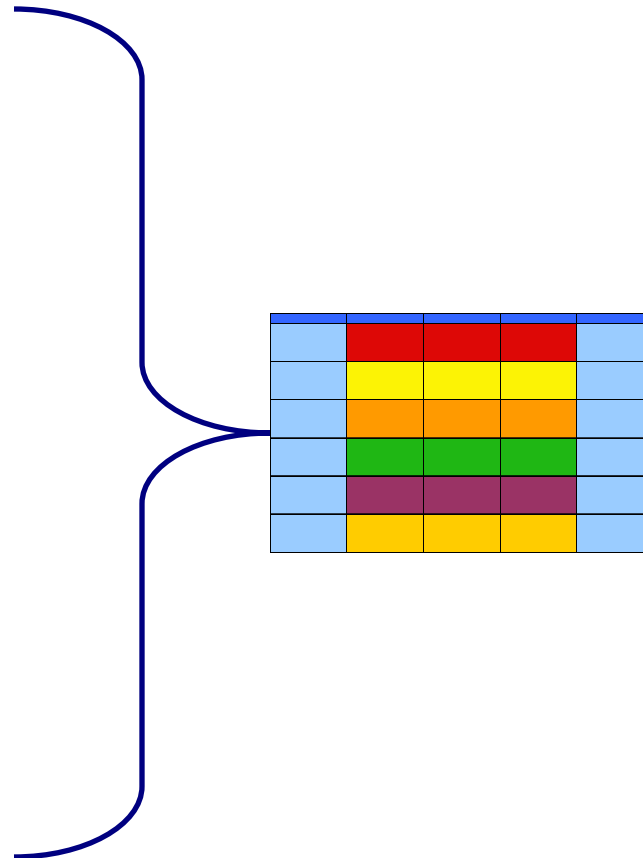
Conceptual

Logical

Physical

Definitional

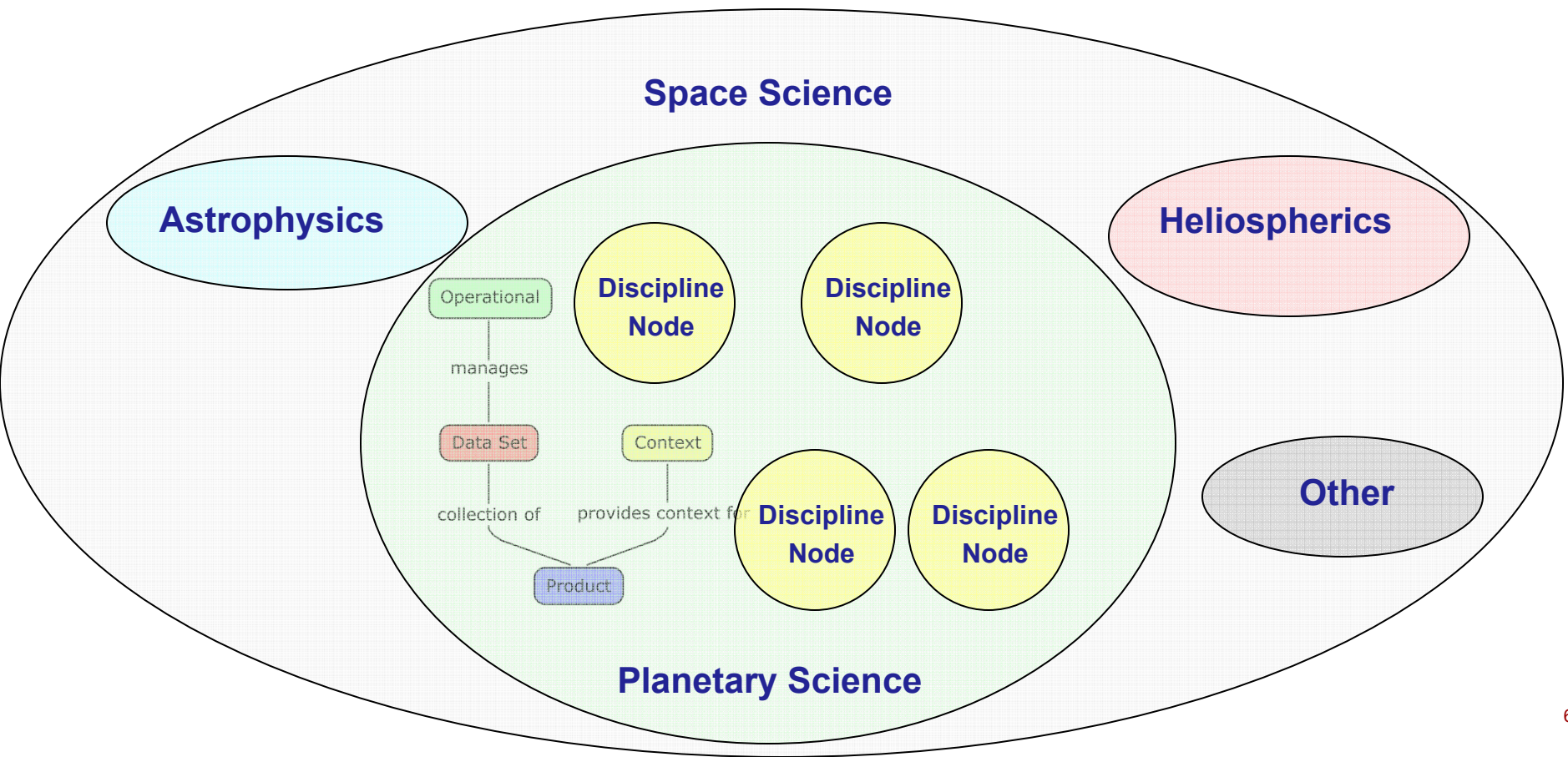
Instance





Contextual

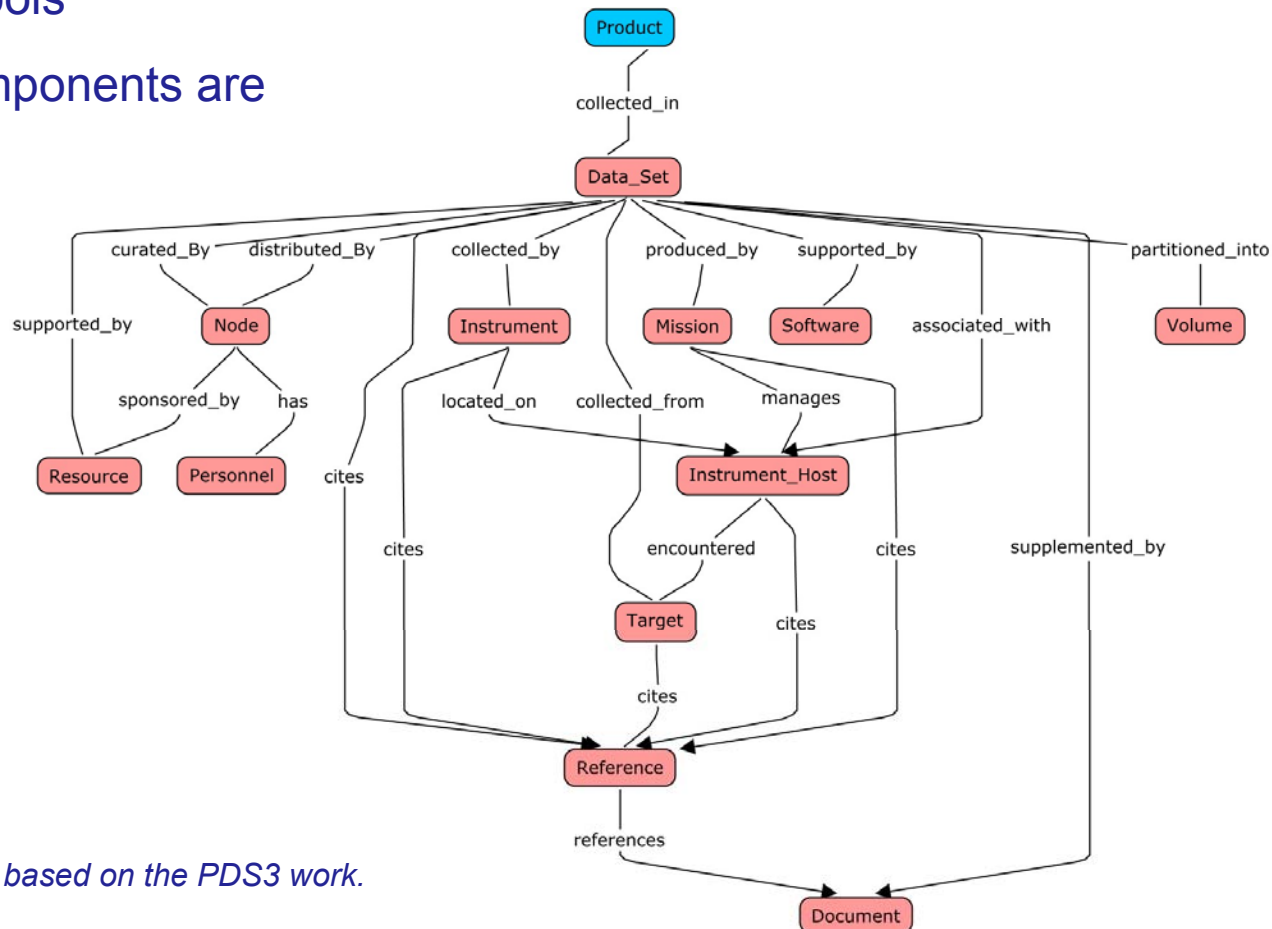
- Contextual models define the scope of data from the strategists point of view.
- They establish the context which includes definitions of the boundaries of the system.
- Do PDS Data Models and Data Models from different disciplines overlap? If they overlap, how are they managed?





Conceptual

- Conceptual models define the community model of data from a manager's point of view
- They are concerned with the language of the community – concepts, facts, words, and symbols
- What components are missing?



This example is based on the PDS3 work.



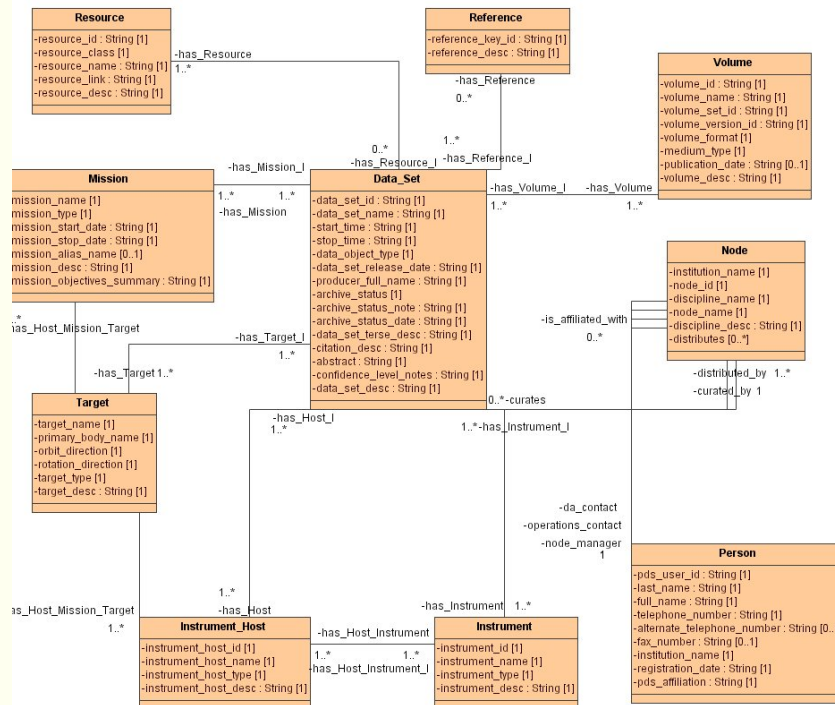
Logical

- Logical models define the system model of data from a designer's point of view
- They are concerned with entity classes, attributes, and relationships that describe the things of significance in rigorous terms

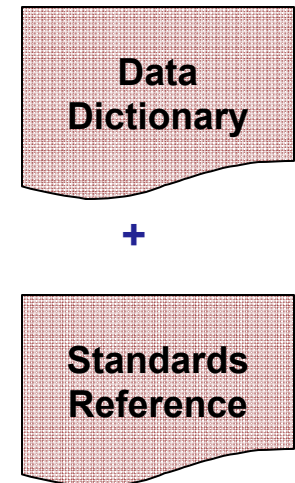
12.14 Instrument

Root Class: Context_Description
Class Description: An entity that collects data.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description			
	Context_Core			
	Instrument			
Subclass	none			
Attribute	instrument_desc	1		
	instrument_host_id	1..*	DD	
	instrument_id	1	DD	
	instrument_name	1	DD	
	instrument_type	1	DD	
	reference_key_id	1..*		0
Inherited Attribute	none			
Association	refer Host Instrument	1..*	Instrument Host	
	refer Instrument I	1..*	Data Set	
	refer Reference	1..*	Reference	
Inherited Association	none			
Referenced from	Catalog			
	Data Set			
	IDE Ancillary			
	IDE Earthbase			
	IDE Spacecraft			
	Identification Data Elements			
	Instrument Host			



Initiate



These examples are based on the PDS3 work.



Physical

- Physical models define a technology model of data from a builder's point of view
- They are concerned with design artifacts for a particular technology –relational databases, object-oriented applications, XML™ schemas

add

Target specific artifacts

Primary and Foreign Keys

ODL Objects

XML Element/Attributes

Update

Data Dictionary

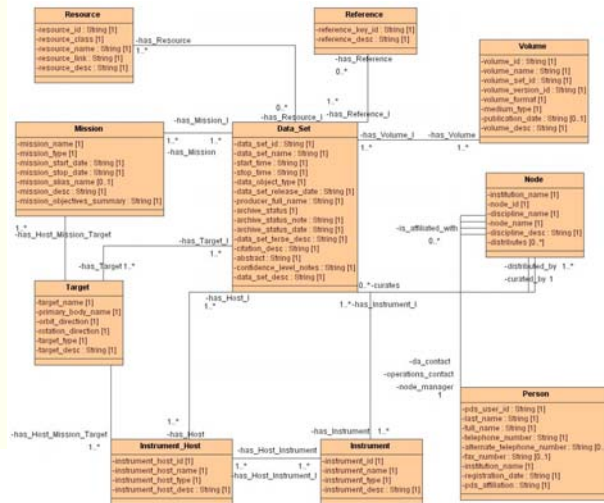
+

Standards Reference

12.14 Instrument

Root Class: Context_Description
Class Description: An entity that collects data.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description			
	Context_Core			
	Instrument			
Subclass	none			
Attribute	instrument_desc	1		
	instrument_host_id	1..*	DD	
	instrument_id	1	DD	
	instrument_name	1	DD	
	instrument_type	1	DD	
	reference_key_id	1..*		O
Inherited Attribute	none			
Association	refer Host Instrument	1..*	Instrument Host	
	refer Instrument J	1..*	Data Set	
	refer Reference	1..*	Reference	
Inherited Association	none			
Referenced from	Catalog			
	Data Set			
	IDE Ancillary			
	IDE Earthbase			
	IDE Spacecraft			
	Identification Data Elements			
	Instrument Host			





Data Definition

- Data definition models define detailed representations of data from an implementer's point of view
- They are concerned with the implementation details in a particular language – database storage specifications, object-oriented programs, XML schema definitions (XSDs)

```
Object = Image
Lines =
Line_Samples =
```

```
/**
 * @(#) Instrument.java
 */
package Context_Description;

public class Instrument extends Context_Core
{
    private String instrument_type;

    private String instrument_name;

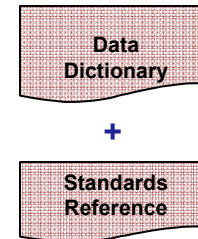
    private String instrument_id;

    private String instrument_host_id;

    private String reference_key_id;

    private String instrument_desc;
}
```

Use and Update





Instance

- Data instance models represent a created artifact from a user's point of view.
- They are concerned with the implemented data models using target languages. (e.g. database records, java objects, XML documents)

Object = Image

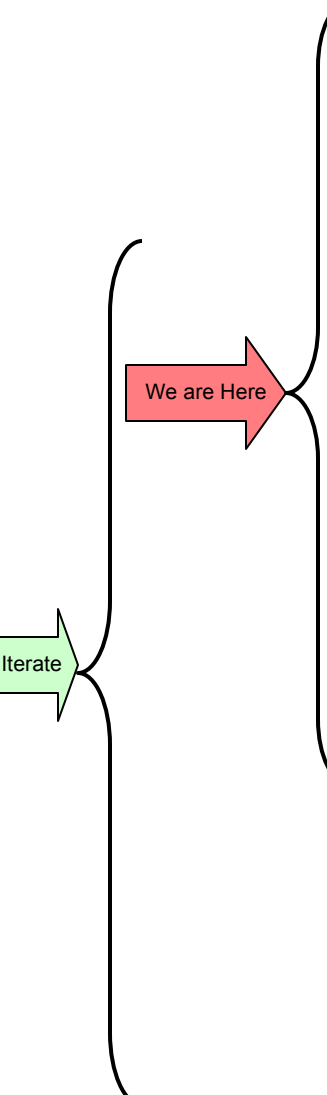
Lines = 800

Line_Samples = 800

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <psdd:Resource rdf:about="urn:oid:1.3.6.1.4.1.1306.2.10200000001">
    <dc:identifier>CLEM1-/L/E/Y-A/B/U/H/L/N-2-EDR-V1.0:LHD0001A.032</dc:identifier>
    <dc:title>CLEMENTINE: EDR IMAGE ARCHIVE: LHD0001A.032</dc:title>
    <prof:resLocation>http://starbrite.jpl.nasa.gov/prod?object=PDS.CN.Product&keywordQuery=OFSN+%3D-
    <psdd:data_set_id>CLEM1-/L/E/Y-A/B/U/H/L/N-2-EDR-V1.0</psdd:data_set_id>
    <psdd:product_id>LHD0001A.032</psdd:product_id>
    <psdd:file_specification_name>LUN032/LHXXXXXX/LHXXXXXA/</psdd:file_specification_name>
    <psdd:instrument_id rdf:resource="&psdd;HIRES"/>
    <psdd:target_name rdf:resource="&psdd;MOON"/>
    <psdd:volume_id rdf:resource="&psdd;CL_0001"/>
    <psdd:filter_name rdf:resource="&psdd;D"/>
    <psdd:mission_phase_name rdf:resource="&psdd;LUNAR_MAPPING"/>
    <psdd:revolution_number>032</psdd:revolution_number>
    <psdd:frame_sequence_number>1</psdd:frame_sequence_number>
    <psdd:start_time>1994-02-26T21:07:11.349Z</psdd:start_time>
    <psdd:exposure_duration>1.067000</psdd:exposure_duration>
    <psdd:gain_mode_id>4</psdd:gain_mode_id>
    <psdd:right_ascension>264.57</psdd:right_ascension>
    <psdd:declination>65.59</psdd:declination>
    <psdd:target_center_distance>2516.0</psdd:target_center_distance>
    <psdd:center_latitude>-85.92</psdd:center_latitude>
    <psdd:center_longitude>191.79</psdd:center_longitude>
  </psdd:Resource>
</rdf:RDF>
```

Information Model Development Phases

- Requirements - Determine the constraints that the data architecture must satisfy, or the relative weights of design parameters. (e.g. Long-Term Preservation vs Contemporary User Service?)
- Decomposition - Define the major components of the data architecture to satisfy the requirements. (e.g. Archive, Query, Data Format, Repository)
- Specification - Decide what functions sub-components must support, and the interfaces between them. This process decomposes the requirements of the whole system into the requirements for the subcomponents. (e.g. Archive, Query, Distribution, Science Use)
- Design - Choose a detailed implementation for each sub-component. This step normally involves returning to the decomposition step recursively, to define further, smaller, subcomponents.



We are Here

Iterate

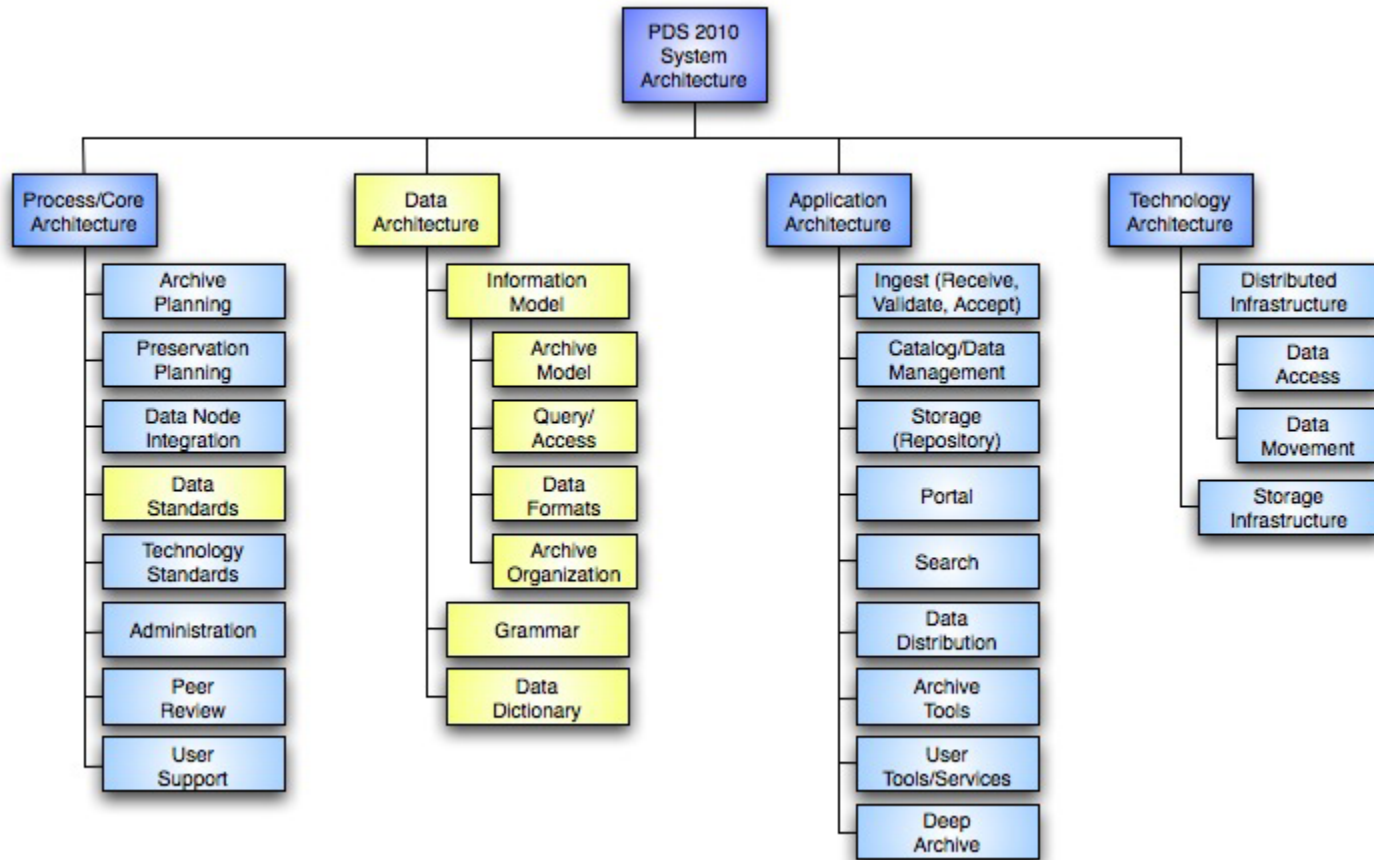


Data Architecture Components (Cont)

- *Information Model*
- **Data Dictionary**
- **Grammar**



Data Architecture Scope





Data Dictionary

- **Data Dictionary**
 - Data Element Definitions
 - Data Element Relations (Data Element used as Object Attributes)
- **Scope**
 - Discrete groups of data elements
 - Common Data Elements
 - Discipline Data Elements
 - Mission/Instrument Data Elements
 - Ground Data System/Instrument Team Operational Data Elements
 - PDS Operational Data Elements
- **Control Authority**
 - The PDS is the registration authority.
 - Each group has its own submitter/steward.
 - Physical location of each group is a software architecture/implementation issue.
- **Data Dictionary Model**
 - Current PDS DD model is very limited
 - Use Standard Data Dictionary Model
 - E.g. ISO/IEC 11179 Metadata Registry Specification
 - Data dictionary is expressed into target languages for specific functions. (e.g. ODL Tool Data Dictionary)



Grammar (Language)

- **The language (s) into which we express the PDS Information Model.**
 - E.g. PDS3-ODL, PDS4-ODL, PVL, XML, Other
- **Information Model concepts are expressed into a target language.** (e.g. Image Class -> ODL Object)
 - If two or more languages used, then the mapping is done from the model to each language individually. I.e. We do not derive XML elements from ODL objects

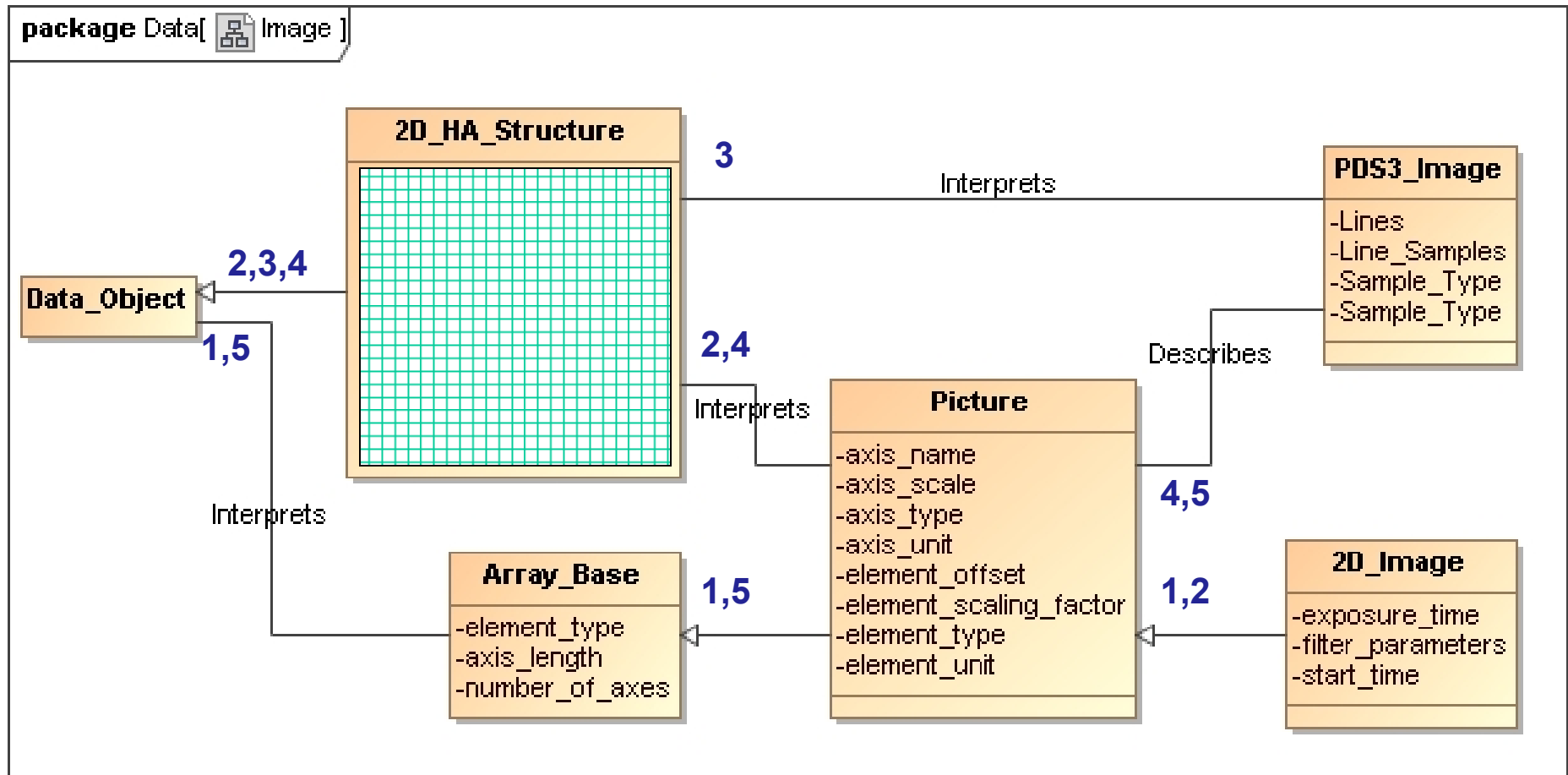


National Aeronautics and
Space Administration

Backup



Simple Image Models





Model Dynamics

- Model centric perspectives characterize the information models themselves and are concerned with their structure, formalism and dynamics.

Perspective	One Extreme	Other Extreme
Level of Authoritativeness	Least authoritative, broader shallowly defined information models	Most authoritative, narrower, more deeply defined information models
Source of Structure	Passive (Transcendent) - Structure originates outside the system	Active (Immanent) - Structure emerges from data or behavior
Degree of Formality	Informal or primarily taxonomic	Formal, having rigorously defined types and relations
Model Dynamics	Read-only, information models are static	Volatile, information models are fluid and changing
Instance Dynamics	Read-only, model instances are static	Volatile, model instances change continuously



Model Dynamics - PDS

Discrete subgroups of data elements and classes – e.g. Common, Discipline Specific, Mission/Instrument, Ground Data System/Instrument Team Operations, PDS Operations

PDS
Authority level is set for each distinct subgroup. Each subgroup has its own submitter and steward. Broadly defined common subgroup (e.g. data set) and deeply defined discipline and mission specific subgroups. (e.g. camera model)
Structure originates outside the system - Science usage dictates structure.
Formal, having rigorously defined types and relations.
Dynamic is set for each subgroup. In general base classes are read-only (static) with extensions allowed. New subgroups are create as necessary.
Read-only, resource instances are static



Application Characteristics

- Application centric perspectives are concerned with how applications use and manipulate information models.

Perspective	One Extreme	Other Extreme
Control/Degree of Manageability	Externally focused, public(little or no control)	Internally focused, private(full control)
Application Changeability	Static (with periodic updates)	Dynamic
Coupling	Loosely-coupled	Tightly-coupled
Integration Focus	Information integration	Application integration
Lifecycle Usage	Design Time	Run Time



Object-Oriented Model Design Principles

- Unique Object Identifiers - Every PDS object (class instance) shall have a globally unique identifier. This is required for each "identifiable" object (product, data set, PDS document, etc) that is to be unambiguously accessed internally within the PDS distributed system and externally by the planetary science community and general public.
- Composite Objects - Objects may contain other objects. (e.g. Table and Column).
- References and Integrity - One object shall be able to reference another object and the integrity of these references shall be maintained. (e.g. Data Set and Document)
- Object-type hierarchy - An object shall be allowed to use the attributes defined in a parent (i.e. super or generalized) object. (e.g. Table, Series, Time Series)
- Object Encapsulation - An object shall be able to be encapsulated to support reusability, interoperability, and modularization. For the PDS this includes both the actual data and structure and interpretation metadata. Large data blocks are required for the actual data. (e.g. Separate class definitions for an objects structure, interpretation, and the digital bits.)
- Ordered Sets - An object's relationships shall be orderable. (e.g. Columns in a Table)
- Localized Changes - A change to a class of objects should have minimal impact on other classes of objects and the implementation. (e.g. Adding a "required" attribute to Series should only effect Series and Time Series objects. The impact of the change should be easy to identify)
- Data Manipulation Language (Query Language) - At least one standard language shall be available for querying the objects. (There are plenty of standard query languages. Why write a new one?)



Information Model Design Principles

- The PDS shall maintain a conceptual information model that is implementation independent.
- All implementation models will be derived from the conceptual information model.
 - Data models for subsystem implementation
 - Applications for generating and validating PDS data holdings
 - Logical and physical models expressed in any language or notation.
- The information model shall be defined using a formal data modeling notation.



Requirements, Components, Functions

PDS Level 3 Requirements	Suggested Components	Suggested Function
1.4.4, 1.4.5, 1.4.8, 2.2.2, 2.3.1, 2.6.2, 3.3.3, 3.3.5	Data Product	Ingest, Archive, Tracking, Validation, Distribution, Analysis, Query
2.3.1, 3.3.3, 3.3.5	Data Structure	Distribution, Analysis
1.4.8, 2.3.1, 3.3.3, 3.3.5	Data Description	Distribution, Analysis
1.4.4, 1.4.5, 1.4.8, 2.2.2, 2.3.1, 2.6.2, 3.3.3, 3.3.5	Ancillary Product	Ingest, Archive, Tracking, Validation, Distribution, Analysis, Query
1.4.1, 1.4.4, 1.4.5, 1.4.8, 2.2.2, 2.3.1, 2.6.2	Data_Set	Ingest, Validation, Archive, Distribution, Tracking, Validation, Query
1.4.1, 1.4.4, 1.4.5, 1.4.8, 2.2.2, 2.3.1, 2.6.2	Context (Instrument, Mission, ...)	Ingest, Validation, Archive, Distribution, Tracking, Validation, Query
1.4.2	Data Dictionary	Validation, Query, Archive
1.4.8	Query is a subset of the archive model but extensible for discipline specific needs.	Query
1.4.4, 1.4.5, 2.2.2, 2.3.1, 2.6.2	Volume (releases, transfer packaging)	Tracking, Ingest, Archive, Validation, Query
2.3.1	Grammar	Validation, Ingest
2.7.1	Repository	Archive and Long-term Storage
1.4.4, 1.4.8, 2.3.1, 2.6.2, 3.3.4, 3.3.5	Coordinate systems	Query, Analysis
1.4.4, 1.4.5, 1.4.8, 2.3.1, 2.6.2, 3.3.5	Map Projection	Ingest, Archive, Analysis, Validation, Query
1.4.4, 1.4.5, 1.4.8, 2.3.1, 2.6.2, 3.3.5	SPICE	Ingest, Archive, Tracking, Validation, Distribution, Analysis, Query
1.4.4, 1.4.5, 1.4.8, 2.3.1, 2.6.2, 3.3.5.	Software	Ingest, Archive, Tracking, Validation, Distribution, Analysis, Query
1.4.4, 1.4.5, 1.4.8, 2.3.1, 2.6.2, 3.3.5	Document	Ingest, Archive, Tracking, Validation, Distribution, Analysis, Query



Detailed PDS4 Design List

- **PDS DOIs -Citing external references; products we create.**



PDS3 Data Architecture Characteristics

Item	Yes/No
Describes data structures	Yes
Defines data structures	No
Flexible data structures	Yes
Provides clear boundaries of what structures are allowed	No
Flexible data descriptions	Yes
Provides clear boundaries of what descriptions are allowed	No
Locally identified instances	Yes
Global identified instances (exception Profiles)	No
Generic Classes	Yes
Base Classes	No
Two levels of class hierarchy (*_Type keyword)	Yes
Unbounded class hierarchies (extensibility)	No
Unnamed hierarchical associations	Yes
Named class associations as needed	No
Few aliases	Yes
Alternative Names for all Identifiables	No
Attribute Name Spaces	Yes
Data Model Name Spaces	No
Simple Attribute classification	Yes
General classification + registration authority, submitters, and stewards	No
Simple Grouping of Attributes	Yes
Class Definitions	No
Standard Value Sets	Yes
Standard Value Concepts	No
Grammar	Yes
Well Defined Grammar	No
Information Model	Yes
Independent Information Model	No



PDS4 Management Council Questions

- How will PDS-4 enable "one-stop shopping", i.e., seamless access to data that reside at multiple nodes?
- **How will PDS-4 help users by delivering derived data products in the format**, coordinate system, and map projection the user requests?
- How will PDS-4 help data providers by automating the design, production, and delivery of PDS data sets?
- **How will PDS-4 ensure that PDS standards are simple, straightforward, and consistent so that data providers and users can easily understand and apply them?**
- How will PDS-4 ensure that data sets can be safely and efficiently archived in NSSDC and retrieved on demand?
- How will PDS-4 improve the data transfer, data integrity, and