Information Model Specification - From First Principles

PDS4 Information Model Specification Team

October 28, 2008

Strawman

Version 0.080827a

Contents

1 Introduction **6**

2 Audience **6**

3 Acknowledgements **6**

4 Scope **6**

5 History **6**

6 Terminology **7**

7 Document Contents **9**

8 Basic Component Classes **11**
8.1 Data_Location 13
8.2 Data_Object 13
8.3 Descriptive_Data_Elements 14
8.4 IDE_Ancillary 15
8.5 IDE_Earthbase 16
8.6 IDE_Spacecraft 16
8.7 Identification_Data_Elements 17
8.8 Label_Standards_Identifiers 18
8.9 Pointer 19
8.10 Property_Map 19

9 Data Description Classes **20**
9.1 2d_Image 21
9.2 Array 22
9.3 Array_Base 23
9.4 CSV_File 24
9.5 Character_Table 24
9.6 Character_Table_Base 25
9.7 Color_Picture 26
9.8 Columnar_Spectrum 27
9.9 DO_Child 27
9.10 DO_Taggable 28
9.11 Data_Object_Description 28
9.12 Delimited_Fields 28
9.13 Encoded_Stream_Class 29
9.14 Header 30
9.15 PDF_File 30
9.16 Picture 30

9.17 Record_Base 31
9.18 SPICE_Kernel 32
9.19 Unencoded_Stream_Base 32

10 Tagged Data Classes **34**
10.1 Tagged_2D_Image 34
10.2 Tagged_Array 34
10.3 Tagged_Data_Object 36
10.4 Tagged_Header 36

11 Product Classes **38**
11.1 Data_Product 38
11.2 Data_Product_Combined_Detached 39
11.3 Data_Product_Document 40
11.4 Product 40

12 Context Description Classes **41**
12.1 Catalog 42
12.2 Context_Child 43
12.3 Context_Core 43
12.4 Context_Description 44
12.5 Context_Supplemental 45
12.6 Data_Producer 45
12.7 Data_Set 46
12.8 Data_Set_Collection 48
12.9 Data_Set_Map_Projection 48
12.10Data_Supplier 49
12.11Directory 50
12.12Discipline_Description 50
12.13Image_Map_Projection 51
12.14Instrument 53
12.15Instrument_Host 53
12.16Mission 54
12.17Node 55
12.18Personnel 56
12.19Reference 57
12.20Software 58
12.21Software_Online 59
12.22Target 60
12.23Volume 60

13 Operational Classes **62**
13.1 Data_Set_HouseKeeping 63
13.2 Data_Set_Release 63
13.3 Inventory 64
13.4 Inventory_Data_Set_Info 64
13.5 Inventory_Node_Media_Info 64
13.6 Operational_Description 65
13.7 Resource_Information 65

14 Unification **67**

15 Specification Dictionary **67**

16 Glossary **110**

17 Review Notes **111**

List of Figures

1 PDS Information Model - Concept Map 7
2 Basic Component UML Class Diagram 12
3 Data Description UML Class Diagram 21
4 Tagged Data Object UML Class Diagram 35
5 Product UML Class Diagram 38
6 Context Description UML Class Diagram 42
7 Operations UML Class Diagram 62
8 PDS Object Unification Using OAIS Information Object . . . 67

1 Introduction

This document presents the PDS4 Information Model Specification for all components of the Planetary Data System (PDS). This specification is currently under development.

2 Audience

This specification is intended for use by programmers and data engineers who require formal definitions of various parts of the Planetary Data System in order to support development of data sets, archiving utilities, and interfaces involving PDS holdings or operations.

3 Acknowledgements

This document was written by the PDS4 Information Model Specification Working Group.

4 Scope

This document defines all classes in use in the PDS, including those classes used to define archival elements as well as classes used for high-level descriptions and operational support. It also documents the associations among classes. Figure 1 illustrates a few of the more basic classes using a Concept Map diagram.

5 History

To be updated. Original design documents were used as the baseline for development of this specification. The initial draft was then modified to reflect changes formally adopted for the PDS Standards Reference (PDSSR), and the most recent available updates to the PDS Data Dictionary (PDSDD) elements specifically referenced in the main body. Finally, de facto standards representing common contemporary practice in interpreting and applying the PDS Standards Reference to data sets in development were added to the formal specification.

The specific sources were:

PDS Catalog Design Document Version 2.0, JPL D-1152, February 13, 1990.

PDS Standards Reference, V 3.7, JPL D-7669, March 20, 2006.

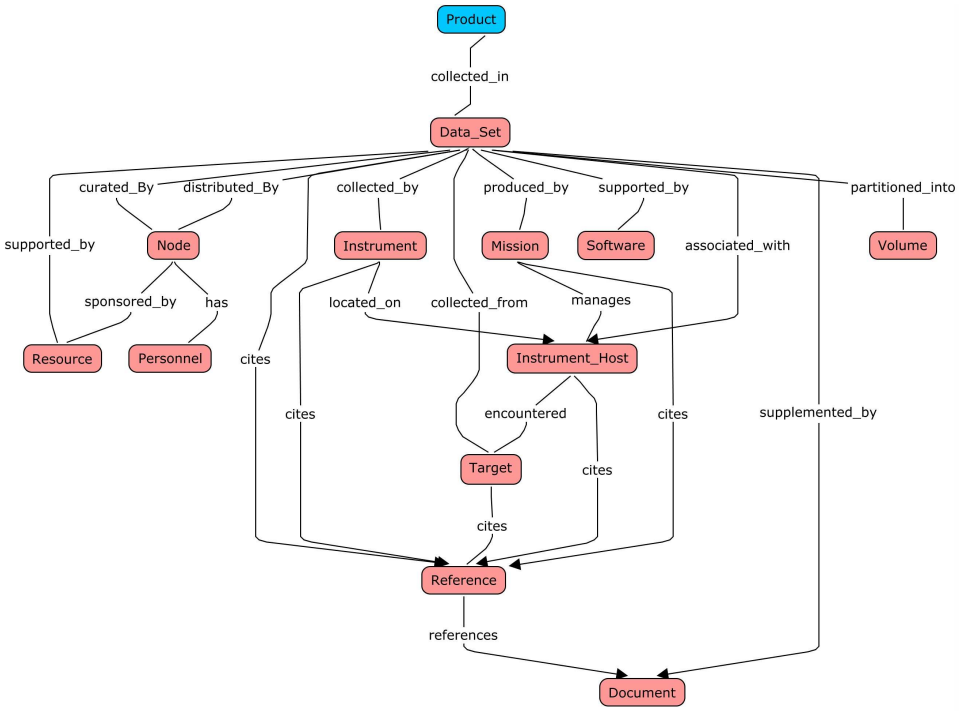

Figure 1: PDS Information Model - Concept Map

Planetary Science Data Dictionary JPL D-7116, Rev. D, July 15, 1996.

The PDS object/element database, Build 1R69, December 31, 2007.

The PDS Standards Change Request Data Base, December 31, 2007.

Reference Model for an Open Archival Information System (OAIS), CCSDS 650.0-B-1, Blue Book, January 2002.

6 Terminology

This document uses very specific engineering terminology to describe the various structures involved. It is particularly important that readers who have absorbed the PDS Standards Reference bear in mind that terms which are familiar in that context can have very different meanings in the present document. Please consult the Glossary for definitions whenever there is any possibility of confusion.

Following are some definitions of essential terms used throughout this document.

An "attribute" is a property or characteristic that allows both identification and distinction.

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.

An "object" is a specific instance of a class.

For example, an electromagnetic wave may be represented mathematically as

$$\boldsymbol{i}_x A cos(\omega t - \boldsymbol{kr} - \varphi)$$

where there are five explicit attributes: polarization $\boldsymbol{i}_x$, amplitude A, frequency ω, wave vector $\boldsymbol{k}$ (which defines the propagation direction), and phase φ. Although shown here as constants, these attributes may be complex functions of other variables; for example, there is an implicit sixth attribute "time" which defines both the beginning and end of the electromagnetic wave. Together these six attributes identify the class (i.e., the family) of all electromagnetic waves. If we then define a coordinate system, specify values for the attributes above, and impose time constraints, we would have an electromagnetic wave object. We would need a different list of attributes to identify a river, a musical score, or a television set, thus these would be different classes.

For this document we identify two special types of objects – the "data object" and the "description object." The data object contains "data," and (by itself) is not otherwise constrained. The description object contains information about another object, such as a data object. By linking a data object with a description object we create a pair which includes both the data and enough information that we can start to read and interpret the bits.

A description object can (and often does) exist without being physically accompanied by another object. The object it describes may not be physical (e.g., a space mission which, although it has physical components, is itself a concept) or it may not be practical to include the physical object (e.g., the planet Saturn).

An "association" is a defined relationship between classes. It has one

direction. The association in the opposite direction is called an inversion relation and is sometimes named by adding a postfix "_I" as in "has_Instrument_I".

"Cardinality" is the number of values allowed to an attribute or association in a single class. Cardinality in general is stated as a range with a minimum and maximum. For example, an attribute that may be multi-valued will have a cardinality of "1..*". A cardinality where the minimum and maximum are the same is often shown as the single value. For example, an attribute required to have exactly one value will have a cardinality of "1". When a value is required the minimum cardinality is at least 1. At least one value is always required in PDS4.

"Entity" is a generic term used to refer to specific attributes or associations listed in a class definition.

Within this document, the term "model" is used to refer to a collection of classes and associations that describe a functional subsection of the Planetary Data System.

7 Document Contents

To be updated. Sections 8 through 14 contain the specification for PDS4. The lowest level building blocks (classes) are defined first, then these are used to construct classes at higher levels; for active users of PDS4, the material in Sections 9 and 10 should seem familiar, but the terminology may be new. The classes in section 13 provide context (instrument, mission, node, etc.); however, many of the corresponding objects do not exist within the data system.

Section 8: the classes defined from first principles

Section 9: the data object and components of PDS labels

Section 10: description classes for common PDS objects

Section 11: "tagged" objects " data objects plus pointers plus descriptions

Section 12: product classes, which are formed from combinations of the above

Section 13: context classes (commonly associated with the PDS Cat-

alog)

Section 14: classes needed for operating and maintaining PDS4

Each section begins with a brief outline, including a hierarchy of the definitions which follow. In some cases a class is defined to group several subclasses when the class itself never appears in PDS (a "phantom" class). To facilitate cross-referencing, the classes are listed alphabetically within each section. Subsections begin with a note on the position within the hierarchy and a brief description of the class. The heart of each subsection is the class definition table. Sections are often accompanied by a UML diagram which shows the relationships among classes graphically.

Class definition tables comprise five columns. The left column is used to separate the table into functional blocks of contiguous rows. The "hierarchy" block restates the position of the class within the definitional hierarchy, and the "subclass" block identifies any subclasses which may exist (be derived from the current class). Attribute and Association blocks list the properties, characteristics, and relationships of the class, some of which may be inherited from parent classes. The "referenced from" block lists classes which may "call" the class being defined.

Within Attribute blocks, the "entity" column lists the properties and characteristics which identify the class and distinguish it from others. The "Indicator" column (far right) tells whether the attribute is optional (O), restricted (R), or both; a restricted attribute has been inherited from a parent class but its use is more narrow than the parent would allow. The "Cardinality" column (middle) shows the number of values allowed. A required attribute for which only one value is allowed will have cardinality "1". A required attribute for which one or more values is allowed will have cardinality "1..*". If a parent's attribute has cardinality "1..*" but the child's cardinality is "1", the Indicator column should show "R". The "Value" column (fourth) includes the indicator Data Dictionary (DD) when a set of valid values for the attribute are provided in the dictionary. A few attributes that represent types have their valid values included in this column.

The Association blocks are handled similarly. The "Entity" column lists relationships among classes using fabricated, but intuitive, names which are unique and consistent across the Specification. The "Value" column (fourth), which is rarely used in the Attribute blocks, lists the class to which the relationship is made.

During construction of the Specification some classes have been subsumed. In particular, any subclass which does nothing more than provide

multiple values for a single attribute (e.g., data_set_target) or any subclass which merely grouped non-repeating attributes (e.g., data_set_information) was subsumed. Only subclasses that grouped several attributes and that repeated were defined explicitly as separate classes (e.g., software_online).

Sections 15-18 contain supplementary information which may be useful in interpreting the remainder of the Specification. For example, the PDS Data Dictionary and PDS Standards Reference both list "PSDD' as an optional element for many PDS objects, effectively allowing every element in the PSDD to be an optional element for such objects. This approach thwarts any attempt at real specificity in the modeling process. The Specification reflects the Data Dictionary and Standards Reference listings.

8 Basic Component Classes

This section provides the PDS4 classes that have been defined from first principles. It includes the basic structure, abstract interpretation, and user interpretation classes.

The basic component class hierarchy is illustrated in the following diagram. This diagram presents the subclass relation for each class in a hierarchical (tree) format, providing a visual representation of the classes in relation to their parent classes.

```
+ Data_Object
+ Descriptive_Data_Elements
+ Identification_Data_Elements
+ + IDE_Ancillary
+ + IDE_Earthbase
+ + IDE_Spacecraft
+ Label_Standards_Identifiers
+ Pointer
+ + Data_Location
+ Property_Map
```

The class hierarchy above includes 10 unique classes.

The classes in this section are illustrated using a Unified Modeling Language (UML) class hierarchy diagram in Figure 3. The following sections present the classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

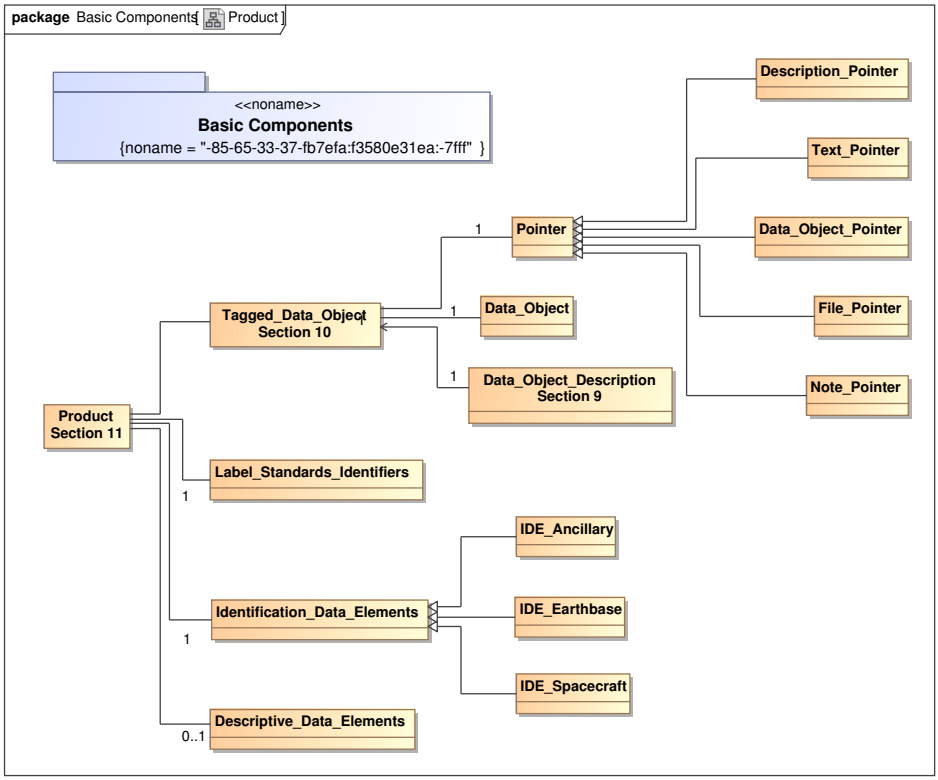

Figure 2: Basic Component UML Class Diagram

8.1 Data_Location

Root Class: Pointer
Class Description: TBD description

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Pointer . Data_Location			
Subclass	none			
Attribute	file_name offset	1 1		
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Tagged_2D_Image Tagged_Array Tagged_Data_Object Tagged_Header			

8.2 Data_Object

Root Class: Data_Object
Class Description: A sequence of digital bits.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object			
Subclass	none			
Attribute	bit_string	1		
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	TDO_Core TDO_Other TDO_Supplemental Tagged_2D_Image Tagged_Array Tagged_Array_pds3 Tagged_Collection Tagged_Data_Object Tagged_Data_Object_pds3 Tagged_Document Tagged_File_Explicit Tagged_File_Implicit Tagged_File_Implicit_Document Tagged_Gazetteer_Table Tagged_Header Tagged_Header_PDS3 Tagged_Histogram Tagged_History Tagged_Image Tagged_Index_Table Tagged_Palette Tagged_Qube Tagged_SPICE_Kernel Tagged_Series Tagged_Spectral_Qube Tagged_Spectrum Tagged_Spreadsheet Tagged_Table Tagged_Text			

8.3 Descriptive_Data_Elements

Root Class: Descriptive_Data_Elements
Class Description: In addition to the data identification elements required for various types of data, PDS strongly recommends including additional data elements related to specific types of data. These descriptive elements should include any elements needed to interpret or process the

data objects or which would be needed to catalog the data product to support potential search criteria at the product level.

	Entity	Card	Value/Class	Ind
Hierarchy	Descriptive_Data_Elements			
Subclass	none			
Attribute	PSDD	1..*		O
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Data_Product Data_Product_Combined_Detached Data_Product_Document Product			

8.4 IDE_Ancillary

Root Class: Identification_Data_Elements
Class Description: The data identification elements provide additional information about a data product that can be used to relate the product to other data products from the same data set or data set collection. This extension to the base identification class is for ancillary data products.

	Entity	Card	Value/Class	Ind
Hierarchy	Identification_Data_Elements . IDE_Ancillary			
Subclass	none			
Attribute	none			
Inherited Attribute	data_set_id	1..*	DD	
	instrument_host_name	1..*	DD	O
	instrument_name	1..*	DD	O
	product_creation_time	1		
	product_id	1		
	start_time	1		O
	stop_time	1		O
	target_name	1..*	DD	O
Association	none			
Inherited Association	collected_about	1..*	Target	O
	collected_by	1..*	Instrument	O
	collected_in	1..*	Data_Set	
	collected_on	1..*	Instrument_Host	O
Referenced from	none			

8.5 IDE_Earthbase

Root Class: Identification_Data_Elements
Class Description: The data identification elements provide additional information about a data product that can be used to relate the product to other data products from the same data set or data set collection.This extension to the base identification class is for data products associated with Earthbase observatories.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Identification_Data_Elements . IDE_Earthbase			
Subclass	none			
Attribute	none			
Inherited Attribute	instrument_host_name	1..*	DD	R
	instrument_name	1..*	DD	R
	start_time	1		R
	stop_time	1		R
	target_name	1..*	DD	R
	data_set_id	1..*	DD	
	product_creation_time	1		
	product_id	1		
Association	none			
Inherited Association	collected_about	1..*	Target	O
	collected_by	1..*	Instrument	O
	collected_in	1..*	Data_Set	
	collected_on	1..*	Instrument_Host	O
Referenced from	none			

8.6 IDE_Spacecraft

Root Class: Identification_Data_Elements
Class Description: The data identification elements provide additional information about a data product that can be used to relate the product to other data products from the same data set or data set collection. This extension to the base identification class is for data products associated with Spacecraft.

	Entity	Card	Value/Class	Ind
Hierarchy	Identification_Data_Elements . IDE_Spacecraft			
Subclass	none			
Attribute	spacecraft_clock_start_count spacecraft_clock_stop_count	1 1		
Inherited Attribute	instrument_host_name instrument_name start_time stop_time target_name data_set_id product_creation_time product_id	1..* 1..* 1 1 1..* 1..* 1 1	DD DD DD DD 	R R R R R
Association	none			
Inherited Association	collected_about collected_by collected_in collected_on	1..* 1..* 1..* 1..*	Target Instrument Data_Set Instrument_Host	O O O
Referenced from	none			

8.7 Identification_Data_Elements

Root Class: Identification_Data_Elements
Class Description: The data identification elements provide additional information about a data product that can be used to relate the product to other data products from the same data set or data set collection.

	Entity	Card	Value/Class	Ind
Hierarchy	Identification_Data_Elements			
Subclass	IDE_Ancillary IDE_Earthbase IDE_Spacecraft			
Attribute	data_set_id instrument_host_name instrument_name product_creation_time product_id start_time stop_time target_name	1..* 1..* 1..* 1 1 1 1 1..*	DD DD DD DD	 O O O O O
Inherited Attribute	none			
Association	collected_about collected_by collected_in collected_on	1..* 1..* 1..* 1..*	Target Instrument Data_Set Instrument_Host	O O O
Inherited Association	none			
Referenced from	Data_Product Data_Product_Combined_Detached Data_Product_Document Product			

8.8 Label_Standards_Identifiers

Root Class: Label_Standards_Identifiers
Class Description: Each PDS label must begin with the PDS_VERSION_ID data element. This element identifies the published version of the Standards to which the label adheres, for purposes of both validation as well as software development and support. For labels adhering to the standards described in this document the appropriate value is PDS3. The DD_VERSION_ID element identifies the version of the PDS Data Dictionary to which a label complies. Current PDS practice is to identify a Data Dictionary version with the identifier used for the PDS catalog build in which it resides, e.g., pdscat1r47, pdscat1r48, and so on. This keyword will use the upper case representation of the catalog identifier, e.g., PDSCAT1R47, PDSCAT1R48, etc. The LABEL_REVISION_NOTE element is a free form, unlimited-length character string providing information regarding the revision status and authorship of a PDS label. It should include at least the latest revision date and the author of the current version, but may include a complete editing history.

	Entity	Card	Value/Class	Ind
Hierarchy	Label_Standards_Identifiers			
Subclass	none			
Attribute	dd_version_id label_revision_note pds_version_id	1 1 1	 PDS3	O O
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Data_Product Data_Product_Combined_Detached Data_Product_Document Product			

8.9 Pointer

Root Class: Pointer
Class Description: A pointer within an ODL label is used to provide the location of the data object or additional metadata.

	Entity	Card	Value/Class	Ind
Hierarchy	Pointer			
Subclass	Data_Location			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

8.10 Property_Map

Root Class: Property_Map
Class Description: TBD description

	Entity	Card	Value/Class	Ind
Hierarchy	Property_Map			
Subclass	none			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	2d_Image Array Header			

9 Data Description Classes

Data Format Classes are used to interpret and define the structure of data objects. For example, an Image class uses attributes to define an image data object as a two-dimensional array of values, all of the same type, each of which is referred to as a sample.

The de facto Data Format Class hierarchy for PDS 3 is illustrated in the following diagram. This diagram presents the subclass relation for each class in a hierarchical (tree) format and provides a visual representation of the classes in relation to their parent classes.

The PDS 3 standards often state that certain classes are subclasses. For example, the Index Table object is stated as being a subclass of Table. However the data modeling formalism used for this specification precluded the definition of many of these classes as subclasses. In addition, the Time Series object has not been included in the specification since it was never defined as a PDS object.

```
+ Data_Object_Description
+ + DO_Child
+ + DO_Taggable
+ + + Array_Base
+ + + + Color_Picture
+ + + + Picture
+ + + + + 2d_Image
+ + + + + Array
+ + + Encoded_Stream_Class
+ + + + Header
+ + + + PDF_File
+ + + + SPICE_Kernel
+ + + Record_Base
+ + + + Character_Table_Base
+ + + + + Character_Table
+ + + + + Columnar_Spectrum
+ + + Unencoded_Stream_Base
+ + + + Delimited_Fields
+ + + + + CSV_File
```

The class hierarchy above includes 19 unique classes.

The data format classes are illustrated using a UML class hierarchy diagram in Figure 4. This diagram defines the classes that are used to describe how the data object is structured. The following sections present the data format classes in a table format. The table includes the class

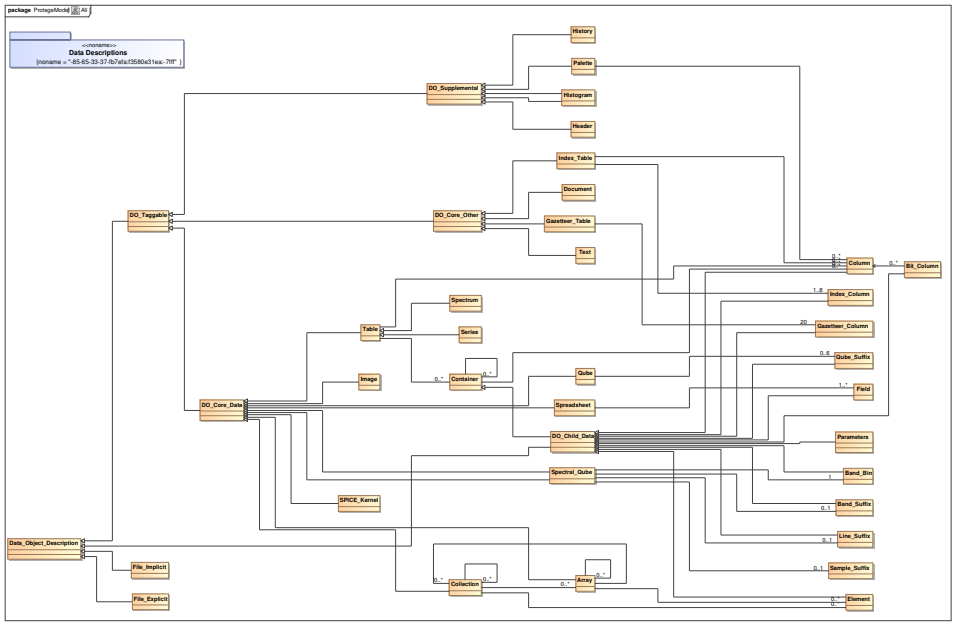

Figure 3: Data Description UML Class Diagram

hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

9.1 2d_Image

Root Class: Data_Object_Description
Class Description: The data class ultimately used by data preparers and seen by users would include all the parameters required by the picture class augmented by additional attributes that would be used to explain, define or manipulate this image. Exposure time and filter parameters are a given, and I would also expect to see orientation parameters that would define, for example, North, or the center-of-image coordinates on a target body. – The range of attributes ultimately included in this definition will be determined by a long series of design decisions based on things like how many optional attributes we want to track for any given object class, how many object classes we want to have to deal with, the relationship between objects and product labels (if a product has more than one object, can it have more than one target, e.g.), and so on.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Array_Base . . . Picture . . . . 2d_Image			
Subclass	none			
Attribute	exposure_time	1		O
	filter_parameters	1		O
	maximum	1		O
	median	1		O
	minimum	1		O
	name	1		O
	standard_deviation	1		O
	start_time	1		O
Inherited Attribute	axis_length	1..*		
	element_type	1		
	axis_name	1..*	DD	
	axis_scale	1..*		O
	axis_type	1..*		O
	axis_unit	1..*	DD	O
	element_bytes	1		
	element_offset	1		
	element_scaling_factor	1		
	element_unit	1		
	number_of_axes	1	2	R
Association	property_map	1	Property_Map	O
Inherited Association	none			
Referenced from	Tagged_2D_Image			

9.2 Array

Root Class: Data_Object_Description
Class Description: TBD description

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Array_Base . . . Picture . . . . Array			
Subclass	none			
Attribute	none			
Inherited Attribute	axis_length	1..*		
	element_type	1		
	axis_name	1..*	DD	
	axis_scale	1..*		O
	axis_type	1..*		O
	axis_unit	1..*	DD	O
	element_bytes	1		
	element_offset	1		
	element_scaling_factor	1		
	element_unit	1		
	number_of_axes	1	2	R
Association	property_map	1	Property_Map	O
Inherited Association	none			
Referenced from	Tagged_Array			

9.3 Array_Base

Root Class: Data_Object_Description
Class Description: Heterogeneous N-dimensional array of scalars – This low-level class defines the base arrangement of the bytes of the data object in the physical file. Basic constraints on storage order, element types, and maximum number and length of axes are defined here. Methods would be provided to create/destroy the storage space for the data based on the attributes.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Array_Base			
Subclass	Picture Color_Picture			
Attribute	axis_length element_type number_of_axes	1..* 1 1		
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

9.4 CSV_File

Root Class: Data_Object_Description
Class Description: All the attributes of the DELIMITED_FIELDS class, with this constraint: delimiter_style = CSV

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Unencoded_Stream_Base . . . Delimited_Fields . . . . CSV_File			
Subclass	none			
Attribute	none			
Inherited Attribute	delimiter_style field_name field_type maximum_record_length number_of_fields number_of_records stream_type	1 1..* 1..* 1 1 1 1	CSV ASCII Binary	R R
Association	none			
Inherited Association	none			
Referenced from	none			

9.5 Character_Table

Root Class: Data_Object_Description
Class Description: For general tables that just contain straight tabula-

tions of information, it is probably sufficient to use just the attributes in the above abstract class, except that you don't use abstract classes themselves - thus this extra user class would be required but probably wouldn't actually add anything. Whether there are any additional attributes would ultimately depend on where we decide to locate attributes like targetting information.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description . DO_Taggable . . Record_Base . . . Character_Table_Base . . . . Character_Table			
Subclass	none			
Attribute	none			
Inherited Attribute	field_format field_location field_name field_type field_unit number_of_fields number_of_records record_bytes	1..* 1..* 1..* 1..* 1..* 1 1 1		
Association	none			
Inherited Association	none			
Referenced from	none			

9.6 Character_Table_Base

Root Class: Data_Object_Description
Class Description: This class would include the constraint that all fields must be in ASCII, and uses the record_base class to allocate whatever is needed for reading the data from the file into memory. Attributes describe everything needed to read and write the data. At a minimum, methods would be provided to read and display records one at a time. Additional methods might return an entire column, sort based on one or more column values, generate a structure definition for inclusion in source code, and so on.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Record_Base . . . Character_Table_Base			
Subclass	Character_Table Columnar_Spectrum			
Attribute	field_format field_location field_name field_type field_unit	1..* 1..* 1..* 1..* 1..*		
Inherited Attribute	number_of_fields number_of_records record_bytes	1 1 1		
Association	none			
Inherited Association	none			
Referenced from	none			

9.7 Color_Picture

Root Class: Data_Object_Description
Class Description: TBD description

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Array_Base . . . Color_Picture			
Subclass	none			
Attribute	axis_name element_offset element_scaling_factor element_unit	1..* 1 1 1	DD	O O O O
Inherited Attribute	axis_length element_type number_of_axes	1..* 1 1	 3	 R
Association	none			
Inherited Association	none			
Referenced from	none			

9.8 Columnar_Spectrum

Root Class: Data_Object_Description
Class Description: This class would stipulate that the spectrum must be organized so that one column contains frequency, and that frequency column must be identified so that it can be recognized and used for generating plots, e.g. Methods might be provided to plot a column vs the frequency column, perhaps using a third column for an error bar.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Record_Base . . . Character_Table_Base . . . . Columnar_Spectrum			
Subclass	none			
Attribute	frequency_field frequency_unit	1 1		
Inherited Attribute	field_format field_location field_name field_type field_unit number_of_fields number_of_records record_bytes	1..* 1..* 1..* 1..* 1..* 1 1 1		
Association	none			
Inherited Association	none			
Referenced from	none			

9.9 DO_Child

Root Class: Data_Object_Description
Class Description: TBD description

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Child			
Subclass	none			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

9.10 DO_Taggable

Root Class: Data_Object_Description
Class Description: This abstract class is the parent of classes that can be a component of a tagged_data_object.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description . DO_Taggable			
Subclass	Record_Base Encoded_Stream_Class Array_Base Unencoded_Stream_Base			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Tagged_Data_Object Tagged_Data_Object_pds3			

9.11 Data_Object_Description

Root Class: Data_Object_Description
Class Description: This abstract class is the parent of all classes used to describe data in the PDS.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description			
Subclass	DO_Child DO_Taggable			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

9.12 Delimited_Fields

Root Class: Data_Object_Description
Class Description: storage_class =¿ unencoded_stream_base(stream_type = ASCII)

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Unencoded_Stream_Base . . . Delimited_Fields			
Subclass	CSV_File			
Attribute	delimiter_style field_name field_type maximum_record_length number_of_fields number_of_records	1 1..* 1..* 1 1 1		
Inherited Attribute	stream_type	1	ASCII Binary	R
Association	none			
Inherited Association	none			
Referenced from	none			

9.13 Encoded_Stream_Class

Root Class: Data_Object_Description
Class Description: This class just carries the encoding type identification. – In fact, this class would likely do double-duty as both storage class and the abstract class, because about the only thing you can abstract from encoded files is their encoding type. That's the only thing you need to know to process the bytes - everything else is interpretation, documentation or cataloging info.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Encoded_Stream_Class			
Subclass	PDF_File Header SPICE_Kernel			
Attribute	external_standard	1		
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

9.14 Header

Root Class: Data_Object_Description
Class Description: TBD description

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description . DO_Taggable . . Encoded_Stream_Class . . . Header			
Subclass	none			
Attribute	name	1		O
Inherited Attribute	external_standard	1		
Association	property_map	1	Property_Map	O
Inherited Association	none			
Referenced from	Tagged_Header			

9.15 PDF_File

Root Class: Data_Object_Description
Class Description: This class would document the contents of a PDF file. The encoding_type attribute would be constrained to a small set of appropriate values (possibly only one). Methods might be provided to display the document info, render the PDF, or convert it to some other format.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description . DO_Taggable . . Encoded_Stream_Class . . . PDF_File			
Subclass	none			
Attribute	content_type	1		O
Inherited Attribute	external_standard	1		
Association	none			
Inherited Association	none			
Referenced from	none			

9.16 Picture

Root Class: Data_Object_Description
Class Description: This abstract class will provide the detail to actually read the data into memory and apply any automatic conversions that are required, turning an array of numbers into a displayable picture. Here the dimensionality of the base storage class is constrained (to 2 axes). Constraints on display orientation (i.e., the location of the [0,0] pixel and

how the raster is drawn) are made at this level. Also, there would likely be additional constraints on element type to exclude types that do not lend themselves to straight-forward pixel rendering, like characters or complex numbers.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Data_Object_Description . DO_Taggable . . Array_Base . . . Picture			
Subclass	2d_Image Array			
Attribute	axis_name axis_scale axis_type axis_unit element_bytes element_offset element_scaling_factor element_unit	1..* 1..* 1..* 1..* 1 1 1 1	DD DD 	 O O O
Inherited Attribute	axis_length element_type number_of_axes	1..* 1 1	 2	 R
Association	none			
Inherited Association	none			
Referenced from	none			

9.17 Record_Base

Root Class: Data_Object_Description
Class Description: Heterogeneous repeating record structure – At this level there isn't really anything you can know about the record strcuture, except that it will have some number of fields. Depending on the implementation, you might be able to allocate some initial memory to hold, for example, the head node for a linked list of field descriptors, and you might want to allocate enough memory to hold the entire object contents in some applications.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Record_Base			
Subclass	Character_Table_Base			
Attribute	number_of_fields number_of_records record_bytes	1 1 1		
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

9.18 SPICE_Kernel

Root Class: Data_Object_Description
Class Description: This class would be used for SPICE kernel files. The kernel_type information might, in fact, be better used as the encoding_type, depending on how many attributes are shared among kernel files. Methods might be provided to perform the binary/character conversion, where appropriate, or to extract comments from inside the kernels.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Encoded_Stream_Class . . . SPICE_Kernel			
Subclass	none			
Attribute	kernel_type_new	1		
Inherited Attribute	external_standard	1		
Association	none			
Inherited Association	none			
Referenced from	none			

9.19 Unencoded_Stream_Base

Root Class: Data_Object_Description
Class Description: Once again, not a lot of information to record here, but you may need to know the stream_type to open a file handle. Whenever an unencoded byte stream is used, there will have to be an abstract class above it to define how to parse the stream.

	Entity	Card	Value/Class	Ind
Hierarchy	Data_Object_Description . DO_Taggable . . Unencoded_Stream_Base			
Subclass	Delimited_Fields			
Attribute	stream_type	1	ASCII Binary	
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

10 Tagged Data Classes

This section provides the classes that define the key building blocks of a data product. It includes the classes that link a data description to a data object.

The Label Class hierarchy is illustrated in the following diagram. This diagram presents the subclass relation for each class in a hierarchical (tree) format, providing a visual representation of the classes in relation to their parent classes.

```
+ Tagged_Data_Object
+ + Tagged_2D_Image
+ + Tagged_Array
+ + Tagged_Header
```

The class hierarchy above includes 4 unique classes.

The classes are illustrated using a Unified Modeling Language (UML) class hierarchy diagram in Figure 5. This diagram defines the tagged data classes. The following sections present the classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

10.1 Tagged_2D_Image

Root Class: Tagged_Data_Object
Class Description: TBD description

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Tagged_Data_Object . Tagged_2D_Image			
Subclass	none			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	data_object_description data_location data_object	1 1..* 1	2d_Image Data_Location Data_Object	R
Referenced from	none			

10.2 Tagged_Array

Root Class: Tagged_Data_Object
Class Description: TBD description

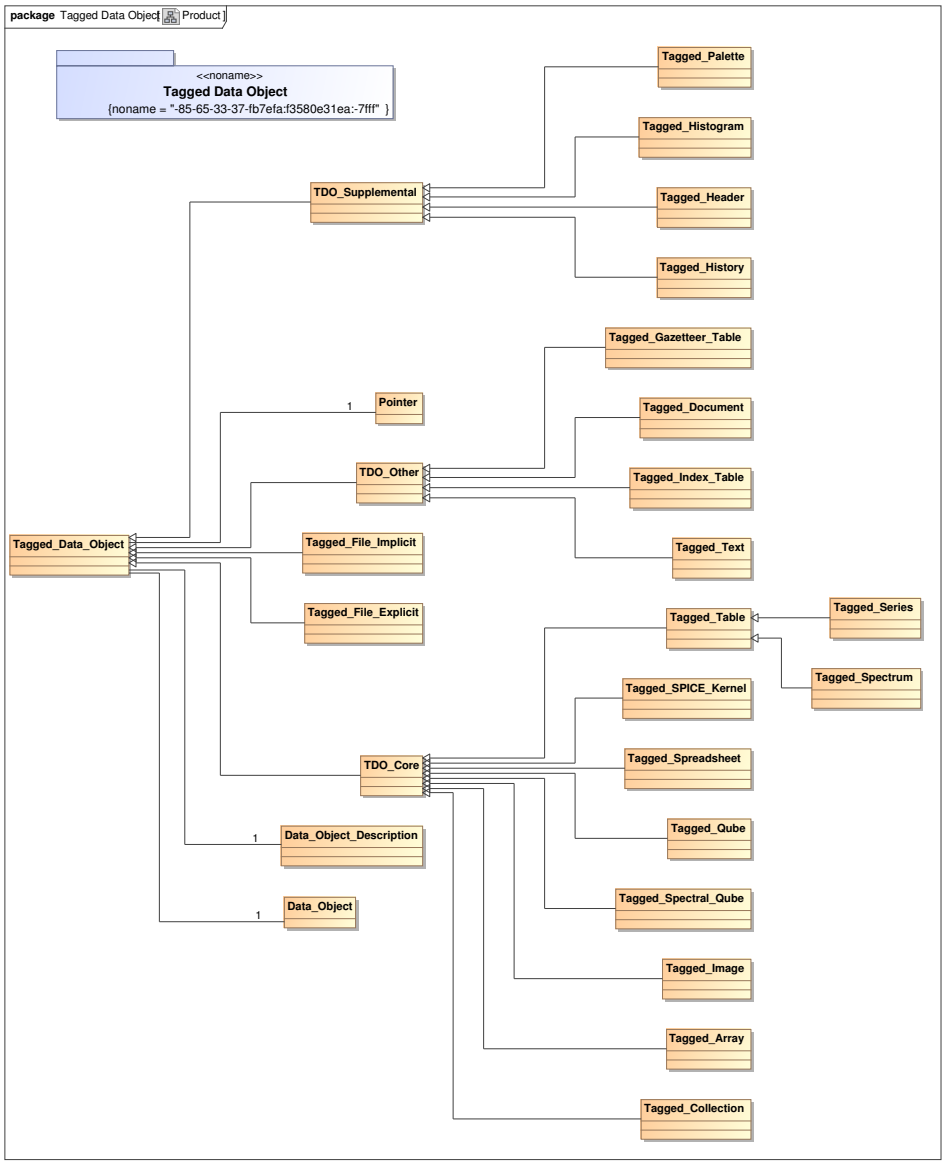

Figure 4: Tagged Data Object UML Class Diagram

	Entity	Card	Value/Class	Ind
Hierarchy	Tagged_Data_Object . Tagged_Array			
Subclass	none			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	data_object_description data_location data_object	1 1..* 1	Array Data_Location Data_Object	R
Referenced from	none			

10.3 Tagged_Data_Object

Root Class: Tagged_Data_Object
Class Description: A tagged data object consists of a data object in association with a data object description and a pointer to the data object.

	Entity	Card	Value/Class	Ind
Hierarchy	Tagged_Data_Object			
Subclass	Tagged_Array Tagged_2D_Image Tagged_Header			
Attribute	none			
Inherited Attribute	none			
Association	data_location data_object data_object_description	1..* 1 1	Data_Location Data_Object DO_Taggable	
Inherited Association	none			
Referenced from	none			

10.4 Tagged_Header

Root Class: Tagged_Data_Object
Class Description: TBD description

	Entity	Card	Value/Class	Ind
Hierarchy	Tagged_Data_Object . Tagged_Header			
Subclass	none			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	data_location data_object data_object_description	1..* 1 1	Data_Location Data_Object Header	 R
Referenced from	none			

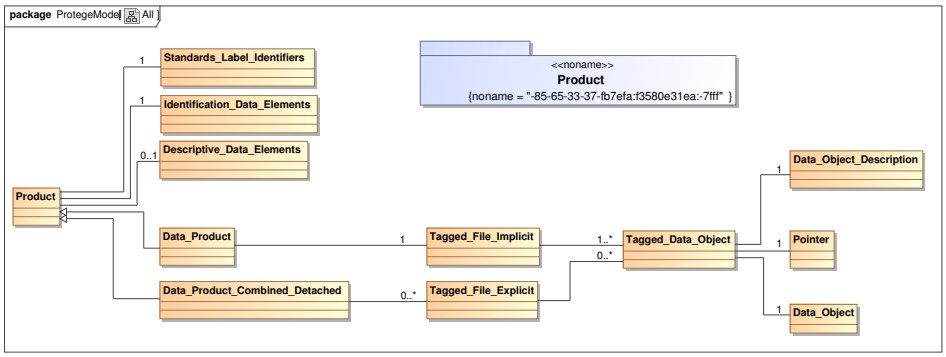

Figure 5: Product UML Class Diagram

11 Product Classes

This section provides a draft set of classes for products. It uses component classes to define the de facto set of product classes in PDS 3. The path from the product class to a specific tagged_data_object (e.g. Tagged_Image) is Product to Data Product to Tagged_File_Implicit to TDO_Core to Tagged_Image.

The Data Product Class hierarchy is illustrated in the following diagram. This diagram presents the subclass relation for each class in a hierarchical (tree) format, providing a visual representation of the classes in relation to their parent classes.

```
+ Product
+ + Data_Product
+ + Data_Product_Combined_Detached
+ + Data_Product_Document
```

The class hierarchy above includes 4 unique classes.

The data product classes are illustrated using a Unified Modeling Language (UML) Class Hierarchy diagram in Figure 6. This diagram defines the classes that comprise a data product. The following sections present the data product classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

11.1 Data_Product

Root Class: Product
Class Description: At its simplest, a data product consists of a PDS

label and the data object that it describes. More complex data products may contain several mutually dependent data objects, a primary object and one or more secondary objects, or both. In all cases, a single label is used to describe all parts of the product (even if they are held in separate physical files). A single PRODUCT_ID value is defined for the entire set in that PDS label.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Product . Data_Product			
Subclass	none			
Attribute	none			
Inherited Attribute	SFDU URI	1 1	SFDU	O O
Association	has_DCS has_TFI_TDO	1..* 1	Context_Supplemental Tagged_File_Implicit	O
Inherited Association	has_DDE has_IDE has_LSI	1 1 1	Descriptive_Data_Elements Identification_Data_Elements Label_Standards_Identifiers	O
Referenced from	none			

11.2 Data_Product_Combined_Detached

Root Class: Product
Class Description: A single PDS detached data product label file is used to describe the contents of more than one data product file. The combined detached label contains pointers to individual data products.

	Entity	**Card**	**Value/Class**
Hierarchy	Product . Data_Product_Combined_Detached		
Subclass	none		
Attribute	none		
Inherited Attribute	SFDU URI	1 1	SFDU
Association	has_TFE_TDO	1..*	Tagged_File_Explicit
Inherited Association	has_DDE has_IDE has_LSI	1 1 1	Descriptive_Data_Elemer Identification_Data_Elem Label_Standards_Identifi
Referenced from	none		

11.3 Data_Product_Document

Root Class: Product
Class Description: A Document Data Product consists of tagged Document files, label standard identifiers, identification data elements, and additional data elements and classes that describe the data product.

	Entity	**Card**	**Value/Class**
Hierarchy	Product . Data_Product_Document		
Subclass	none		
Attribute	none		
Inherited Attribute	SFDU URI	1 1	SFDU
Association	has_TFI_TDO	1	Tagged_File_Implicit_Document
Inherited Association	has_DDE has_IDE has_LSI	1 1 1	Descriptive_Data_Elements Identification_Data_Elements Label_Standards_Identifiers
Referenced from	none		

11.4 Product

Root Class: Product
Class Description: A product comprises at least one data object and descriptive information about that data object.

	Entity	**Card**	**Value/Class**
Hierarchy	Product		
Subclass	Data_Product Data_Product_Document Data_Product_Combined_Detached		
Attribute	SFDU URI	1 1	SFDU
Inherited Attribute	none		
Association	has_DDE has_IDE has_LSI	1 1 1	Descriptive_Data_Element Identification_Data_Eleme Label_Standards_Identifier
Inherited Association	none		
Referenced from	Data_Set		

12 Context Description Classes

The catalog model defines the classes that exist in the planetary science community and that provide a context within which science data products are collected, located, and used. For example, the Mars Viking Digital Image Mosaic is a data set created from images that were collected by the two vidicon cameras that flew on the Viking Orbiters. The catalog model includes classes such as mission, instrument, and data set that are subsequently used to create objects that describe the Viking mission, the two Vidicon cameras, and the resulting data set. These objects and their relationships provide the context for the digital images collected.

The catalog class hierarchy is illustrated in the following diagram. This diagram presents the subclassOf relation for each class in a hierarchical (tree) format and provides a visual representation of the classes in relation to their parent classes.

```
+ Context_Description
+ + Context_Child
+ + + Software_Online
+ + Context_Core
+ + + Data_Set
+ + + Data_Set_Collection
+ + + Instrument
+ + + Instrument_Host
+ + + Mission
+ + + Node
+ + + Personnel
+ + + Reference
+ + + Software
+ + + Target
+ + + Volume
+ + Context_Supplemental
+ + + Catalog
+ + + Data_Producer
+ + + Data_Set_Map_Projection
+ + + Data_Supplier
+ + + Directory
+ + + Discipline_Description
+ + + Image_Map_Projection
```

The class hierarchy above includes 23 unique classes.

The catalog model is illustrated using the UML class hierarchy diagram in Figure 7. This diagram shows the classes that belong to the

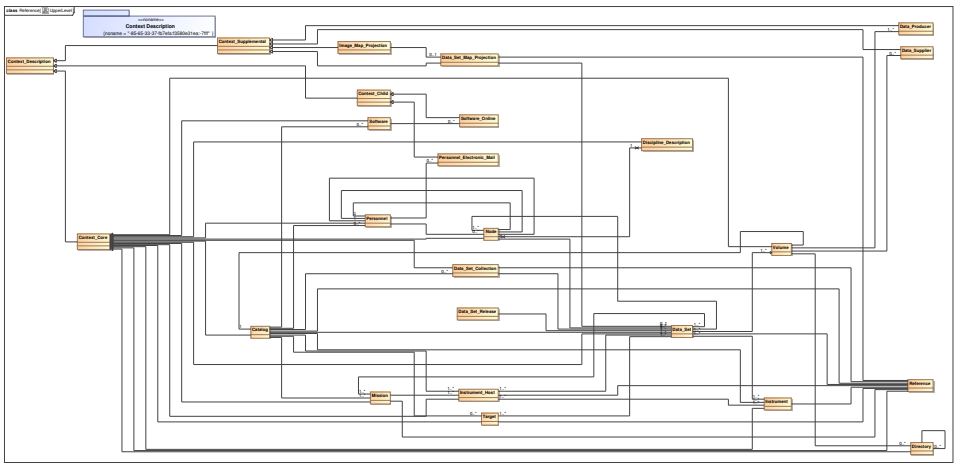

Figure 6: Context Description UML Class Diagram

planetary science domain and that provide a context in which scientific data products are collected, located, and used. The associations between classes are one directional. Inverse associations are defined when necessary. For example, to model the many-to-many relation between the data set and target classes, the has_Target association relates the data set class to the target class. The inverse, has_Target_I, relates the target class back to the data set class. The following sections present the catalog classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate. The following sections present the catalog classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

12.1 Catalog

Root Class: Context_Description
Class Description: The CATALOG object is used within a VOLUME object to reference the completed PDS high-level catalog object set. The catalog object set provides additional information related to the data sets on a volume.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Supplemental . . Catalog			
Subclass	none			
Attribute	PSDD data_set_id logical_volume_path_name logical_volumes	1..* 1..* 1..* 1..*	 DD 	O O O
Inherited Attribute	none			
Association	refer_Catalog_Data_Set refer_Catalog_Data_Set_Coll... refer_Catalog_Instrument refer_Catalog_Instrument_Host refer_Catalog_Mission refer_Catalog_Personnel refer_Catalog_References refer_Catalog_Target refer_Software	1..* 1..* 1..* 1..* 1..* 1..* 1..* 1..* 1..*	Data_Set Data_Set_Collection Instrument Instrument_Host Mission Personnel Reference Target Software	 O O O O O
Inherited Association	none			
Referenced from	Volume			

12.2 Context_Child

Root Class: Context_Description
Class Description: This abstract class is the parent for context classes that play the role of either child classes or repeating groups.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Child			
Subclass	Software_Online Personnel_Electronic_Mail			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

12.3 Context_Core

Root Class: Context_Description
Class Description: This abstract class is the parent class of the core context classes. Where data description classes describe data objects, con-

text classes describe other physical or conceptual things in the Planetary Science domain.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core			
Subclass	Volume Mission Target Data_Set_Collection Software Personnel Node Reference Instrument Data_Set Instrument_Host			
Attribute	URI	1		O
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

12.4 Context_Description

Root Class: Context_Description
Class Description: This abstract class is the parent class for all context classes. Where data description classes describe data objects, context classes describe other physical or conceptual things in the Planetary Science domain.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description			
Subclass	Context_Core Context_Child Context_Supplemental			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

12.5 Context_Supplemental

Root Class: Context_Description
Class Description: This abstract class is the parent class for supplemental classes.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Context_Description . Context_Supplemental			
Subclass	Data_Supplier Image_Map_Projection Data_Set_Map_Projection Catalog Data_Producer Directory Discipline_Description			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Data_Product			

12.6 Data_Producer

Root Class: Context_Description
Class Description: The DATA_PRODUCER object is used within a PDS object, such as VOLUME, to provide information about the producer of a PDS data set.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Supplemental . . Data_Producer			
Subclass	none			
Attribute	PSDD	1..*		O
	address_text	1		
	discipline_name	1	DD	O
	electronic_mail_id	1		O
	electronic_mail_type	1	DD	O
	facility_name	1	DD	
	full_name	1		
	node_institution_name	1	DD	
	node_name	1	DD	O
	telephone_number	1		O
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Volume			

12.7 Data_Set

Root Class: Context_Description
Class Description: A collection of related data products. [ANO:010_080204_001_DataSetProductMap]

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Data_Set			
Subclass	none			
Attribute	abstract_desc	1		
	archive_status	1	DD	O
	citation_desc	1		
	confidence_level_note	1		
	data_object_type	1	DD	
	data_set_collection_member_flg	1	DD	
	data_set_desc	1		
	data_set_id	1	DD	
	data_set_name	1	DD	
	data_set_release_date	1		
	data_set_terse_desc	1		
	detailed_catalog_flag	1	DD	
	instrument_host_id	1..*	DD	
	instrument_id_new	1..*		
	mission_name	1..*	DD	
	producer_full_name	1		
	reference_key_id	1..*		O
	start_time	1		
	stop_time	1		
	target_name	1..*	DD	
Inherited Attribute	URI	1		O
Association	curated_by	1	Node	
	distributed_by	1..*	Node	
	refer_Host	1..*	Instrument_Host	
	refer_Instrument	1..*	Instrument	
	refer_Mission	1..*	Mission	
	refer_Product_Implicit	1..*	Product	O
	refer_Reference	1..*	Reference	
	refer_Resource	1..*	Resource_Information	
	refer_Target	1..*	Target	
	refer_Volume	1..*	Volume	
Inherited Association	none			
Referenced from	Catalog Data_Set_Collection Data_Set_Map_Projection Data_Set_Release IDE_Ancillary IDE_Earthbase IDE_Spacecraft Identification_Data_Elements Instrument Instrument_Host Inventory_Data_Set_Info Mission Node Resource_Information Target Volume			

12.8 Data_Set_Collection

Root Class: Context_Description
Class Description: A Data Set Collection is a set of Data Sets that have been selected for some specific purpose.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Data_Set_Collection			
Subclass	none			
Attribute	data_set_collection_desc data_set_collection_id data_set_collection_name data_set_collection_release_dt data_set_collection_usage_desc data_sets producer_full_name reference_key_id start_time stop_time	1 1 1 1 1 1 1 1..* 1 1	 DD DD	 O
Inherited Attribute	URI	1		O
Association	refer_Data_Set refer_Reference	1..* 1..*	Data_Set Reference	
Inherited Association	none			
Referenced from	Catalog			

12.9 Data_Set_Map_Projection

Root Class: Context_Description
Class Description: The IMAGE_MAP_PROJECTION object is one of two distinct objects that define the map projection used in creating the digital images in a PDS data set. The name of the other associated object that completes the definition is DATA_SET_MAP_PROJECTION. The map projection information resides in these two objects, essentially to reduce data redundancy and at the same time allow the inclusion of elements needed to process the data at the image level. Basically, static information that is applicable to the complete data set reside in the DATA_SET_MAP_PROJECTION object, while dynamic information that is applicable to the individual images reside in the IMAGE_MAP_PROJECTION object.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Supplemental . . Data_Set_Map_Projection			
Subclass	none			
Attribute	data_set_id map_projection_desc map_projection_type reference_key_id rotational_element_desc	1 1 1 1..* 1	DD DD 	 O O O O
Inherited Attribute	none			
Association	refer_Data_Set_Projection refer_Reference_Projection	1 1	Data_Set Reference	O O
Inherited Association	none			
Referenced from	Image_Map_Projection			

12.10 Data_Supplier

Root Class: Context_Description
Class Description: The DATA_SUPPLIER object is used within a PDS object, such as VOLUME, to provide information about the supplier of a PDS data set.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Supplemental . . Data_Supplier			
Subclass	none			
Attribute	PSDD address_text discipline_name electronic_mail_id electronic_mail_type facility_name full_name node_institution_name node_name telephone_number	1..* 1 1 1 1 1 1 1 1 1	 DD DD DD DD DD 	O O O
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Volume			

12.11 Directory

Root Class: Context_Description
Class Description: The DIRECTORY object is used to define a hierarchical file organization on a linear (i.e., sequential) medium such as tape. The DIRECTORY object identifies all directories and subdirectories below the root level. It is a required sub-object of the VOLUME object for volumes delivered on sequential media.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Context_Description . Context_Supplemental . . Directory			
Subclass	none			
Attribute	PSDD name record_type sequence_number	1..* 1 1 1	 DD 	O O O
Inherited Attribute	none			
Association	refer_Directory refer_File	1..* 1..*	Directory File_Explicit	O
Inherited Association	none			
Referenced from	Directory Volume			

12.12 Discipline_Description

Root Class: Context_Description
Class Description: The Discipline Description catalog object is completed for the submission of a scientific discipline description to the PDS.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Context_Description . Context_Supplemental . . Discipline_Description			
Subclass	none			
Attribute	discipline_desc discipline_name	1 1	 DD	
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Node			

12.13 Image_Map_Projection

Root Class: Context_Description
Class Description: The IMAGE_MAP_PROJECTION object is one of two distinct objects that define the map projection used in creating the digital images in a PDS data set. The name of the other associated object that completes the definition is DATA_SET_MAP_PROJECTION. The map projection information resides in these two objects, essentially to reduce data redundancy and at the same time allow the inclusion of elements needed to process the data at the image level. Basically, static information that is applicable to the complete data set reside in the DATA_SET_MAP_PROJECTION object, while dynamic information that is applicable to the individual images reside in the IMAGE_MAP_PROJECTION object.

	Entity	Card	Value/Class	In
Hierarchy	Context_Description . Context_Supplemental . . Image_Map_Projection			
Subclass	none			
Attribute	a_axis_radius	1		
	b_axis_radius	1		
	c_axis_radius	1		
	center_latitude	1		
	center_longitude	1		
	coordinate_system_name	1	DD	
	coordinate_system_type	1	DD	
	data_set_id	1	DD	O
	eastern_most_longitude	1		
	first_standard_parallel	1		O
	horizontal_framelet_offset	1		O
	image_id	1		O
	line_first_pixel	1		
	line_last_pixel	1		
	line_projection_offset	1		
	map_projection_rotation	1		
	map_projection_type	1	DD	
	map_resolution	1		
	map_scale	1		
	maximum_latitude	1		
	minimum_latitude	1		
	positive_longitude_direction	1	DD	
	reference_latitude	1		O
	reference_longitude	1		O
	sample_first_pixel	1		
	sample_last_pixel	1		
	sample_projection_offset	1		
	second_standard_parallel	1		O
	vertical_framelet_offset	1		O
	western_most_longitude	1		
Inherited Attribute	none			
Association	refer_Data_Set_Map_Projection	1	Data_Set_Map_Projection	
Inherited Association	none			
Referenced from	File_Explicit Spectral_Qube			

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 instrument_host_id instrument_id_new instrument_name instrument_type reference_key_id	1 1..* 1 1 1 1..*	 DD DD DD 	 O
Inherited Attribute	URI	1		O
Association	refer_Host_Instrument refer_Instrument_I refer_Reference	1..* 1..* 1..*	Instrument_Host Data_Set Reference	
Inherited Association	none			
Referenced from	Catalog Data_Set IDE_Ancillary IDE_Earthbase IDE_Spacecraft Identification_Data_Elements Instrument_Host			

12.15 Instrument_Host

Root Class: Context_Description
Class Description: An entity upon which an instrument is mounted.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Instrument_Host			
Subclass	none			
Attribute	instrument_host_desc instrument_host_id instrument_host_name instrument_host_type reference_key_id	1 1 1 1 1..*	 DD DD SPACECRAFT EARTH_BASED ROVER DATA BASE	 O
Inherited Attribute	URI	1		O
Association	refer_Host_I refer_Host_Instrument_I refer_Reference	1..* 1..* 1..*	Data_Set Instrument Reference	
Inherited Association	none			
Referenced from	Catalog Data_Set IDE_Ancillary IDE_Earthbase IDE_Spacecraft Identification_Data_Elements Instrument Mission			

12.16 Mission

Root Class: Context_Description
Class Description: An entity reponsible for managing a project directed toward the collection of data.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Mission			
Subclass	none			
Attribute	instrument_host_id mission_alias_name mission_desc mission_name mission_objectives_summary mission_start_date mission_stop_date reference_key_id	1..* 1 1 1 1 1 1 1..*	DD DD DD	
Inherited Attribute	URI	1		O
Association	refer_Mission_Host refer_Mission_I refer_Reference	1..* 1..* 1..*	Instrument_Host Data_Set Reference	
Inherited Association	none			
Referenced from	Catalog Data_Set			

12.17 Node

Root Class: Context_Description
Class Description: An entity responsible for the management of science data that is associated with a specific planetary science discipline.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Node			
Subclass	none			
Attribute	discipline_name node_desc node_id node_institution_name node_name	1 1 1 1 1	DD DD DD DD	
Inherited Attribute	URI	1		O
Association	curates da_contact_pds_users_id distributes node_manager_pds_users_id operations_contact_pds_user... refer_Discipline	1..* 1 1..* 1 1 1	Data_Set Personnel Data_Set Personnel Personnel Discipline_Description	
Inherited Association	none			
Referenced from	Data_Set Inventory Personnel			

12.18 Personnel

Root Class: Context_Description
Class Description: A person who has an association with the planetary science community.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Personnel			
Subclass	none			
Attribute	address_text alternate_telephone_number fax_number full_name last_name node_id node_institution_name pds_address_book_flag pds_affiliation pds_user_id registration_date telephone_number	1 1 1 1 1 1 1 1 1 1 1 1	 DD DD DD 	
Inherited Attribute	URI	1		O
Association	has_Electronic_Mail is_affiliated_with	1..* 1..*	Personnel_Electronic_Mail Node	O O
Inherited Association	none			
Referenced from	Catalog Node			

12.19 Reference

Root Class: Context_Description
Class Description: The REFERENCE catalog object is completed for each reference document.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Reference			
Subclass	none			
Attribute	reference_desc reference_key_id	1 1		
Inherited Attribute	URI	1		O
Association	none			
Inherited Association	none			
Referenced from	Catalog Data_Set Data_Set_Collection Data_Set_Map_Projection Instrument Instrument_Host Mission Target			

12.20 Software

Root Class: Context_Description
Class Description: The SOFTWARE catalog object provides general information about a software tool including description, availability information, and dependencies.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Software			
Subclass	none			
Attribute	data_format node_id pds_user_id required_storage_bytes software_desc software_id software_license_type software_name software_purpose software_version_id technical_support_type	1 1 1 1 1 1 1 1 1 1 1	DD DD DD DD DD	
Inherited Attribute	URI	1		O
Association	has_Software_Online	1..*	Software_Online	O
Inherited Association	none			
Referenced from	Catalog			

12.21 Software_Online

Root Class: Context_Description
Class Description: The SOFTWARE_ONLINE object, a sub-object of SOFTWARE catalog object, provides identifying information for each PDS node providing access to a particular SOFTWARE object.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Child . . Software_Online			
Subclass	none			
Attribute	node_id on_line_identification on_line_name platform protocol_type	1 1 1 1..* 1	DD DD DD	
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Software			

12.22 Target

Root Class: Context_Description
Class Description: An entity which is the object of data collection.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Target			
Subclass	none			
Attribute	orbit_direction primary_body_name reference_key_id rotation_direction target_desc target_name target_type	1 1 1..* 1 1 1 1	DD DD DD DD DD	 O
Inherited Attribute	URI	1		O
Association	refer_Reference refer_Target_I	1..* 1	Reference Data_Set	O
Inherited Association	none			
Referenced from	Catalog Data_Set IDE_Ancillary IDE_Earthbase IDE_Spacecraft Identification_Data_Elements			

12.23 Volume

Root Class: Context_Description
Class Description: The VOLUME object describes a physical or logical unit used to store or distribute data products that contain directories and files.

	Entity	Card	Value/Class	Ind
Hierarchy	Context_Description . Context_Core . . Volume			
Subclass	none			
Attribute	PSDD	1..*		O
	block_bytes	1		O
	data_set_collection_id	1	DD	O
	data_set_id	1	DD	
	description	1		
	files	1		O
	hardware_model_id	1	DD	O
	logical_volume_path_name	1..*		O
	logical_volumes	1..*		O
	medium_format	1	DD	O
	medium_type	1	DD	
	note	1		O
	operating_system_id	1	DD	O
	product_type	1	DD	
	publication_date	1		
	transfer_command_text	1		O
	volume_format	1	DD	
	volume_id	1		
	volume_insert_text	1		O
	volume_name	1		
	volume_series_name	1	DD	
	volume_set_id	1	DD	
	volume_set_name	1	DD	
	volume_version_id	1		
	volumes	1		
Inherited Attribute	URI	1		O
Association	refer_Catalog	1	Catalog	
	refer_Data_Producer	1..*	Data_Producer	
	refer_Data_Supplier	1..*	Data_Supplier	O
	refer_Directory	1..*	Directory	O
	refer_File	1..*	File_Explicit	O
	refer_Volume_I	1..*	Data_Set	
Inherited Association	none			
Referenced from	Data_Set			

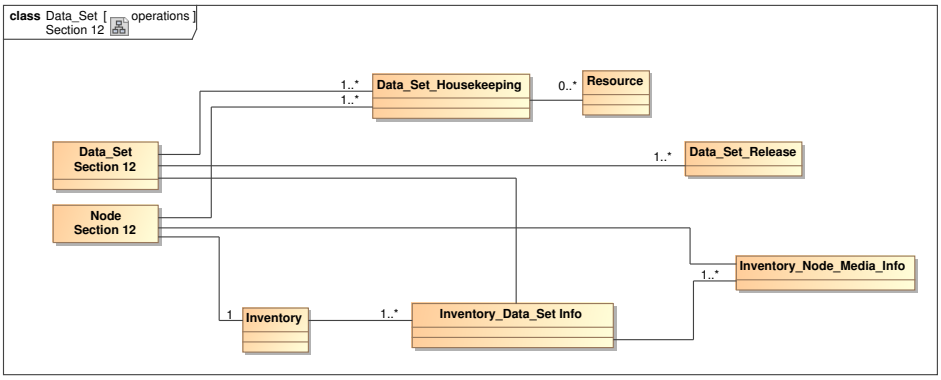

Figure 7: Operations UML Class Diagram

13 Operational Classes

This section provides the set of classes used for PDS operations. These classes include the data set release class that is used to notify users that data are available from the PDS and the inventory node media class that associates a data set with a PDS discipline node. The data set housekeeping and inventory classes were implemented for operations but were never defined as PDS Objects. They have been defined in this document.

The operations class hierarchy is illustrated in the following diagram. This diagram presents the subclassOf relation for each class using a hierarchical (tree) format, providing a visual representation of the classes in relation to their parent classes.

```
+ Operational_Description
+ + Data_Set_HouseKeeping
+ + Data_Set_Release
+ + Inventory
+ + Inventory_Data_Set_Info
+ + Inventory_Node_Media_Info
+ + Resource_Information
```

The class hierarchy above includes 7 unique classes.

The operations classes are illustrated using a Unified Modeling Language (UML) Class Hierarchy diagram in Figure 8. This diagram defines the classes that are used in an operational context. The following sections present the operations classes in a table format. The table includes the class hierarchy, class attributes, and class associations. The class attributes and associations listed include both those used to define the class and those inherited from parent classes. Cardinalities are provided where appropriate.

13.1 Data_Set_HouseKeeping

Root Class: Operational_Description
Class Description: The HouseKeeping class is used to associate discipline node resources to a data set.

	Entity	Card	Value/Class	Ind
Hierarchy	Operational_Description . Data_Set_HouseKeeping			
Subclass	none			
Attribute	curating_node_id data_set_id	1 1	DD DD	
Inherited Attribute	none			
Association	resource	1..*	Resource_Information	O
Inherited Association	none			
Referenced from	none			

13.2 Data_Set_Release

Root Class: Operational_Description
Class Description: The DATA_SET_RELEASE object provides information on the release of a data set or portion of a data set being made available for online access.

	Entity	Card	Value/Class	Ind
Hierarchy	Operational_Description . Data_Set_Release			
Subclass	none			
Attribute	archive_status data_provider_name data_set_id description distribution_type product_type release_date release_id release_medium release_parameter_text	1 1 1 1 1 1 1 1 1 1	DD DD DD 	
Inherited Attribute	none			
Association	relates_to_data_set	1	Data_Set	
Inherited Association	none			
Referenced from	none			

13.3 Inventory

Root Class: Operational_Description
Class Description: One INVENTORY catalog object is completed for each node responsible for orderable data sets from the PDS catalog. This object provides the inventory information necessary to facilitate the ordering of these data sets.

	Entity	**Card**	**Value/Class**	**Ind**
Hierarchy	Operational_Description . Inventory			
Subclass	none			
Attribute	node_id	1	DD	
Inherited Attribute	none			
Association	has_Inventory_Data_Set_Info has_Node_Inventory	1..* 1	Inventory_Data_Set_Info Node	
Inherited Association	none			
Referenced from	none			

13.4 Inventory_Data_Set_Info

Root Class: Operational_Description
Class Description: The INVENTORY_DATA_SET_INFO object, sub-object of the INVENTORY catalog object, identifies a data set through the DATA_SET_ID. This object is repeated once for each orderable and cataloged PDS data set.

	Entity	**Card**	**Value/Class**
Hierarchy	Operational_Description . Inventory_Data_Set_Info		
Subclass	none		
Attribute	product_data_set_id	1	
Inherited Attribute	none		
Association	has_inventory_Node_Media_Info refer_Data_Set_Inventory	1..* 1	Inventory_Node_Media_Info Data_Set
Inherited Association	none		
Referenced from	Inventory		

13.5 Inventory_Node_Media_Info

Root Class: Operational_Description
Class Description: The INVNODEMEDIA (Inventory Node Media Information) catalog object provides information about data set distribution medium, data set size, data set cost, and a maximum number of copies a Node is willing to distribute per medium for a PDS cataloged data set.

This catalog object is repeated for each type of distribution medium.

	Entity	Card	Value/Class	Ind
Hierarchy	Operational_Description . Inventory_Node_Media_Info			
Subclass	none			
Attribute	copies inventory_special_order_note medium_desc medium_type	1 1 1 1	 DD	
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	Inventory_Data_Set_Info			

13.6 Operational_Description

Root Class: Operational_Description
Class Description: This abstract class is the parent of all classes used to describe operational things in the PDS.

	Entity	Card	Value/Class	Ind
Hierarchy	Operational_Description			
Subclass	Data_Set_HouseKeeping Data_Set_Release Inventory_Node_Media_Info Resource_Information Inventory_Data_Set_Info Inventory			
Attribute	none			
Inherited Attribute	none			
Association	none			
Inherited Association	none			
Referenced from	none			

13.7 Resource_Information

Root Class: Operational_Description
Class Description: An entity providing information about a PDS resource.

	Entity	Card	Value/Class	Ind
Hierarchy	Operational_Description . Resource_Information			
Subclass	none			
Attribute	description resource_class resource_id resource_link resource_name resource_status	1 1 1 1 1 1	 DD unique_resource_id URI URL Text Text	
Inherited Attribute	none			
Association	refer_Resource_I	1	Data_Set	O
Inherited Association	none			
Referenced from	Data_Set Data_Set_HouseKeeping			

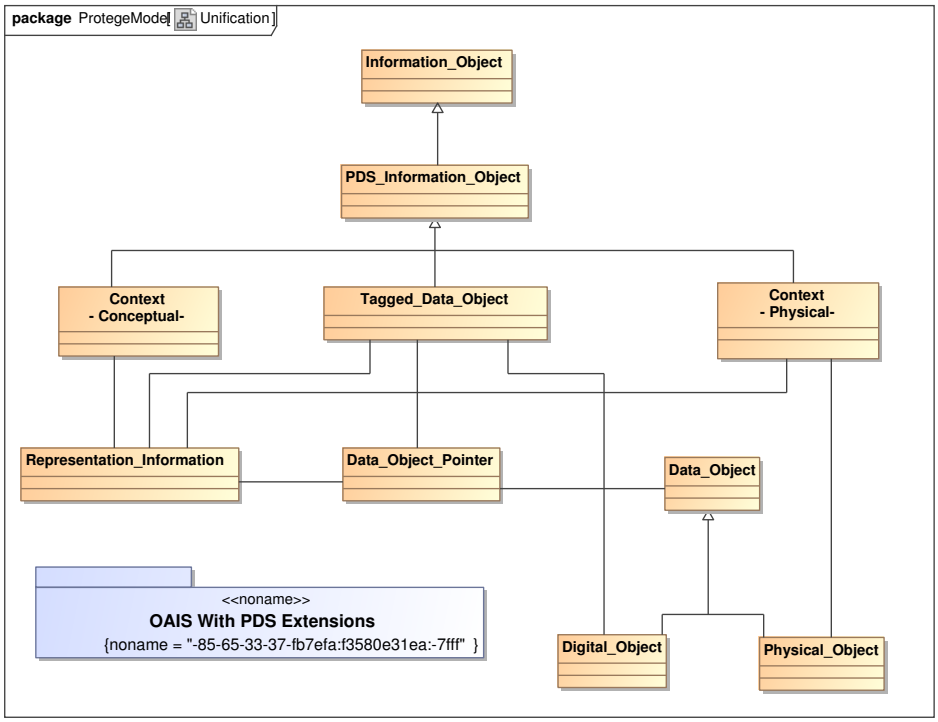

Figure 8: PDS Object Unification Using OAIS Information Object

14 Unification

This section presents the data model for the Information Object, a fundamental component of the Open Archival Information System (OAIS) Reference Model. The Information Object provides a model for the unification of PDS Objects under the PDS defined extensions, the PDS_Information_Object, the Tagged_Data_Object, and two Context classes.

15 Specification Dictionary

The Specification Dictionary provides the definitions of data elements and associations. The data elements are those that are used as class attributes in this specification. They represent a subset of those in the Planetary Science Data Dictionary. The associations are those that are defined and used in this specification.

PSDD The token PSDD implies that any data element in the data dictionary can be used.

SFDU TBD description

URI TBD description

a_axis_radius The a_axis_radius element provides the value of the semimajor axis of the ellipsoid that defines the approximate shape of a target body. 'A' is usually in the equitorial plane.

Type: real

Unit: km

abstract_desc The ABSTRACT_DESC contains an abstract for the product or DATA_SET_INFORMATION object in which it appears. It provides a string that may be used to provide an abstract for the product (data set) in a publication.

Type: character

address_text The address_text data element provides an unlimited-length, formatted mailing address for an individual or institution.

Type: character

alternate_telephone_number The alternate_telephone_number data element provides an alternate telephone number for an individual or node. (Includes the area code.)

Type: character

archive_status The archive_status element provides the status of a data set that has been submitted for inclusion into the PDS archive. If a data set has been partially archived, the archive_status should be ACCUMULATING (e.g., this situation typically occurs when a data set is being produced over a period of time where portions of the data set may be archived, in lien resolution, in peer-review, and under construction). The archive_status_note element is available to describe the archive_status value in finer detail. STANDARD VALUES IN QUEUE - Received at the curation node but no action has been taken by the curation node. Use with caution. PRE PEER REVIEW - Being prepared for peer review under the direction of the curation node. Use with caution IN PEER REVIEW - Under peer review at the curation node but evaluation is not complete. Use with caution IN LIEN RESOLUTION - Peer review completed. Liens are in the process of being resolved. LOCALLY ARCHIVED - Passed peer reviewed with all liens resolved. Considered archived by the curation node but awaiting completion of the standard archiving process. Possible TBD items include

the arrival of the archive volume at NSSDC and ingestion of catalog information into the Data Set Catalog. ARCHIVED - Passed peer review with all liens resolved. Available through the Data Set Catalog and at NSSDC. SUPERSEDED - Superseded by a new version of the data set. This implies that the data set is not to be used unless the requester has specific reasons. When a data set has been superseded the CN will notify NSSDC that their databases need to be updated to advise users of the new status and the location of the replacement data set. SAFED - Received by the PDS with no evaluation. Data will not be formally archived. ACCUMULATING - Portions, but not all, of a data set are in one or more phases of completion (e.g., portions of a data set have been archived while portions remain in lien resolution). Note: If a data set crosses multiple phases of completion, select the highest status level and use the modifier ACCUMULATING. The status is, for example, ARCHIVED-ACCUMULATING, meaning that part of the data set has been archived, but there remains portions of the data set in process. The ARCHIVE_STATUS_NOTE keyword can be used to provide more information. ACCUMULATING value may be used as a modifier to any of the above valid values (e.g., 'ACCUMULATING ARCHIVED', 'ACCUMULATING IN PEER REIVEW').

Type: character

Value: accumulating, archived, in_lien_resolution, in_peer_review, in_queue, locally_archived, pre_peer_review, saved, superseded

axis_length TBD description

axis_name The axis_name element provides the sequence of axis names of a qube or array data object, and identifies the order in which the axes are stored in the object. By default, the first axis name in the sequence identifies the array dimension that varies the slowest, followed by the next slowest, and continuing so the rightmost axis named varies the fastest. The number of names specified must be equal to the value of the axes element. Note: For ISIS qube data objects, the most frequently varying axis is listed first, or leftmost, in the sequence.

Type: character

Value: (band,_sample,_line), (sample,_band,_line), (sample,_line,_band)

axis_scale TBD description

axis_type TBD description

axis_unit The axis_unit element provides the unit(s) of measure of associated axes identified by the axis_name element in an ARRAY data object. For arrays with more than 1 dimension, this element provides a sequence of values corresponding to the number of axes specified. The rightmost item in the sequence corresponds to the most rapidly varying axis, by default.

Type: character

Value: ampere, bits, candela, coulomb, day, degree, farad, gram, gray, henry, hertz, hour, joule, kelvin, kilogram, lumen, lux, meter, minute, mole, n/a, newton, ohm, pascal, pixel, ...

b_axis_radius The b_axis_radius element provides the value of the intermediate axis of the ellipsoid that defines the approximate shape of a target body. 'B' is usually in the equatorial plane.

Type: real

Unit: km

bit_string The bit string.

block_bytes The block_bytes element identifies the number of bytes per physical block used to record data files on magnetic tapes. Note: In the PDS, for portability the block_bytes element should be limited to a maximum value of 32767 for a tape volume.

Type: integer

c_axis_radius The c_axis_radius element provides the value of the semiminor axis of the ellipsoid that defines the approximate shape of a target body. 'C' is normal to the plane defined by 'A' and 'B'.

Type: real

Unit: km

center_latitude The center_latitude element provides a reference latitude for certain map projections. For example, in an Orthographic projection, the center_latitude along with the center_longitude defines the point or tangency between the sphere of the planet and the plane of the projection. The map_scale (or map_resolution) is typically defined at the center_latitude and center_longitude. In unprojected images, center_latitude represents the latitude at the center of the image frame.

Type: real

Unit: deg

center_longitude The center_longitude element provides a reference longitude for certain map projections. For example, in an Orthographic projection, the center_longitude along with the center_latitude defines the point or tangency between the sphere of the planet and the plane of the projection. The map_scale (or map_resolution) is typically defined at the center_latitude and center_longitude. In unprojected images, center_longitude represents the longitude at the center of the image frame.

Type: real

Unit: deg

citation_desc The CITATION_DESC contains a citation for the product or DATA_SET_INFORMATION object in which it appears. It provides a string that may be used to cite the product (data set) in a publication. It should follow the standard citation order as outlined in Appendix B, Section 31.5.5.3.1 of the PDS Standards reference, which in turn follows established practice for scientific journals that cite electronic publications (e.g., AGU Reference citation format). The CITATION_DESC must contain sufficient information to locate the product or data set in the PDS archives. For example, the CITATION_DESC in a DATA_SET_INFORMATION object must contain the DATA_SET_ID; it will also likely contain VOLUME_ID information for the archive volumes, an author list, a release date, and so on as appropriate. Note that if CITATION_DESC is used within any product label within a data set, all product labels within that data set must also have a CITATION_DESC, even if they are only filled with 'N/A'. DATA_SET Example: CITATION_DESC = 'Levin, G.V., P.A. Strat, E.A. Guinness, P.G. Valko, J.H. King, and D.R. Williams, VL1/VL2 MARS LCS EXPERIMENT DATA RECORD V1.0, VL1/VL2-M-LCS-2-EDR-V1.0, NASA Planetary Data System, 2000.' Data Product Example: CITATION_DESC = 'Cunningham, C., MINOR PLANET INDEX TO SCIENTIFIC PAPERS, EAR-A-5-DDR-BIBLIOGRAPHY-V1.0:REFS-REFS-199409, NASA Planetary Data System, 1994.'

Type: character

collected_about Associated Target

collected_by Associated instrument

collected_in Associated data sets.

collected_on Associated Instrument Host

confidence_level_note The confidence_level_note element is a text field which characterizes the reliability of data within a data set or the reliability of a particular programming algorithm or software component. Essentially, this note discusses the level of confidence in the accuracy of the data or in the ability of the software to produce accurate results.

Type: character

content_type TBD description

coordinate_system_name The coordinate_system_name element provides the full name of the coordinate system to which the state vectors are referenced. PDS has currently defined body-fixed rotating coordinate systems. The Planetocentric system has an origin at the center of mass of the body. The planetocentric latitude is the angle between the equatorial plane and a vector connecting the point of interest and the origin of the coordinate system. Latitudes are defined to be positive in the northern hemisphere of the body, where north is in the direction of Earth's angular momentum vector, i.e., pointing toward the hemisphere north of the solar system invariant plane. Longitudes increase toward the east, making the Planetocentric system right-handed. The Planetographic system has an origin at the center of mass of the body. The planetographic latitude is the angle between the equatorial plane and a vector through the point of interest, where the vector is normal to a biaxial ellipsoid reference surface. Planetographic longitude is defined to increase with time to an observer fixed in space above the object of interest. Thus, for prograde rotators (rotating counter clockwise as seen from a fixed observer located in the hemisphere to the north of the solar system invariant plane), planetographic longitude increases toward the west. For a retrograde rotator, planetographic longitude increases toward the east. Note: If this data element is not present in the PDS Image Map Projection Object (for pre-V3.1 PDS Standards), the default coordinate system is assumed to body-fixed rotating Planetographic.

Type: character

Value: apxs_frame, body_fixed_spherical_coords,
earth-sun_line_cartes_coords, ecliptic_inertial_cart_coords,

ecliptic_inertl_sphercl_coords, equatorial_inert_sphrcl_coords, equatorial_inertial_cart_coord, jupiter_minus_system_iii, mast_frame, mb_frame, mean_inertial_hg_1950, mi_frame, neptune_west_longitude_system, non-rotating_spin_coordinates, planet_centered_cylindrical, planetocentric, planetographic, pvo_inertial_spacecraft_coords, pvo_spinning_spacecraft_coords, rat_frame, rover_frame, saturn_minus_longitude_system, sc_centered_ecliptic_coords, uranus_minus_longitude_system, uranus_west_longitude_system, ...

coordinate_system_type There are three basic types of coordinate systems: body-fixed rotating, body-fixed non-rotating and inertial. A body-fixed coordinate system is one associated with a body (e.g., planetary body or satellite). In contrast to inertial coordinate systems, a body-fixed coordinate system is centered on the body and rotates with the body (unless it is a non-rotating type). For the inertial coordinate system type, the coordinate system is fixed at some point in space. Note: If this data element is not present in the PDS Image Map Projection Object (for pre-V3.1 PDS Standards), the default coordinate system is assumed to be body-fixed rotating Planetographic.

Type: character

Value: body-fixed_non-rotating, body-fixed_rotating, inertial

copies The copies element provides the inventory software with the number of copies of an order that a node is willing to ship using a particular order.

Type: integer

curated_by The curated_by slot provides the names of the node that curates the data set.

curates Associated data set.

curating_node_id The curating_node_id element provides the id of the node currently maintaining the data set or volume and is responsible for maintaining catalog information.

Type: character

Value: atmos, esa, geoscience, imaging, imaging-jpl, n/a, naif, nssdc, ppi-ucla, rad, rings, rs, sbn

da_contact_pds_users_id The da_contact slot indicates the person responsible for data administration.

data_format The data_format element supplies the name of the data format or language that was used to archive the science data that this software accesses.

Type: identifier

Value: compressed, fits, gif, hdf, jpeg, pds, pict, spice, vicar

data_location TBD description

data_object Composition association for Data Object.

data_object_description Composition association for the Data Object Description.

data_object_type The data_object_type element identifies the data object type of a given set of data. Example values: IMAGE, MAP, SPECTRUM Note: Within the PDS, data object types are assigned according to the standards outlined in the PDS Standards Reference. Note: within AMMOS and only for the Magellan catalog, this element is used as an alias for data_set_id. The use of data_object_type as such provides backward compatibility with earlier AMMOS conventions. The use of this element as an alias for data_set_id is not recommended for any new tables. See data_set_id.

Type: identifier

Value: array, array,_table, bit_column, collection, column, container, cube, element, file, fits_label, header, histogram, image, image_map_projection, index_table, map, n/a, occultation_profile, palette, qube, series, spectral_qube, spectrum, spice_kernel, spice_kernel, ...

data_provider_name The data_provider_name element provides the name of the individual responsible for providing the release object and data.

Type: character

data_set_collection_desc The data_set_collection_desc element describes the content and type of the related data sets contained in the collection.

Type: character

data_set_collection_id The data_set_collection_id element is a unique alphanumeric identifier for a collection of related data sets or data products. The data set collection is treated as a single unit, whose components are selected according to a specific scientific purpose. Components are related by observation type, discipline, target, time, or other classifications. Example value: PREMGN-E/L/H/M/V-4/5-RAD/GRAV-V1.0 Note: In the PDS, data set collection ids are constructed according to PDS nomenclature standards outlined in the in the Standards Reference.

Type: identifier

Value: grsfe-e-2/3/4/5-rdr-v1.0, ihw-c-2/3-chron-data-v1.0, ihw-c-2/3/4/5-spacecraft-data-v1.0, ihw-c-3-archive-addenda-select-data-v1.0, ihw-c-lc-2/3-v1.0, mgn-v-rss-5-occ-profiles-v1.0, model-m-ames-gcm-5-1977-4-seasons-v1.0, premgn-e/l/h/m/v-4/5-rad/grav-v1.0, sbnsc-ida/gaspra-7-v1.0, sl9-j/c-3-impact-events-select-data-v1.0, vg1/vg2-sr/ur/nr-1/2/4-occ-v1.0, vg1/vg2-sr/ur/nr-2/4-occ-v1.0

data_set_collection_member_flg The data_set_collection_member_flg element indicates whether or not a data set is a member of a data set collection.

Type: character

Value: n, y

data_set_collection_name The data_set_collection_name element provides the full name given to a collection of related data sets or data products. The data set collection is treated as a single unit, whose components are selected according to a specific scientific purpose. Components are related by observation type, discipline, target, time, or other classifications. Example value: PRE-MAGELLAN E/L/H/M/V 4/5 RADAR/GRAVITY DATA V1.0 Note: In the PDS, the data set collection name is constructed according to nomenclature standards outlined in the PDS Standards Reference.

Type: character

Value: ames_mars_general_circulation_model_5_1977_4_seasons_v1.0, geologic_remote_sensing_field_experiment_e_2/3/4/5_rdr_v1.0,

ihw_comet_halley_chronological_data_v1.0,
ihw_comet_lc_2/3_chronological_data_v1.0,
international-halley-watch-archive-addenda-select-data-v1.0,
international_halley_watch_spacecraft_cometary_data_v1.0,
magellan_v_rss_5_occultation_profiles_v1.0,
pre-magellan_e/l/h/m/v_4/5_radar/gravity_data_v1.0,
shoemaker-levy-9-jupiter-impact-events-select-data-v1.0,
special_collection_of_ida_&_gaspra_data_v1.0,
special_collection_of_ida_&_gaspra_data_v1.0,
vg1/vg2_sr/ur/nr_edited/resampled_ring_occultation_v1.0,
vg1/vg2_sr/ur/nr_raw/edited/resampled_ring_occultation_v1.0

data_set_collection_release_dt The data_set_collection_release_dt element provides the date when the data set collection was released for use. Formation rule: YYYY-MM-DD

Type: date

data_set_collection_usage_desc The data_set_collection_usage_desc element provides information required to use the data.

Type: character

data_set_desc The data_set_desc element describes the content and type of a data set and provides information required to use the data (such as binning information).

Type: character

data_set_id The data_set_id element is a unique alphanumeric identifier for a data set or a data product. The data_set_id value for a given data set or product is constructed according to flight project naming conventions. In most cases the data_set_id is an abbreviation of the data_set_name. Example value: MR9/VO1/VO2-M-ISS/VIS-5-CLOUD-V1.0. Note: In the PDS, the values for both data_set_id and data_set_name are constructed according to standards outlined in the Standards Reference.

Type: identifier

Value: a-5-ddr-astermag-v1.0, a-5-ddr-asteroid-spin-vectors-v3.0,
a-5-ddr-astnames-v1.0, a-5-ddr-pole-position-ref-v1.0,
a-5-ddr-pole-position-v1.0, a-5-ddr-taxonomy-v1.0,
arcb-l-rtls-3-70cm-v1.0, arcb-l-rtls-4-70cm-v1.0,

arcb-l-rtls-5-12.6cm-v1.0, arcb-v-rtls-4-12.6cm-v1.0,
arcb/gssr-m-rtls-5-model-v1.0, c130-e-asas-3-rdr-image-v1.0,
c130-e-tims-2-edr-image-v1.0, clem1-l-h-5-dim-mosaic-v1.0,
clem1-l-lidar-5-topo-v1.0, clem1-l-lwir-3-rdr-v1.0,
clem1-l-rss-1-bsr-v1.0, clem1-l-rss-5-bsr-v1.0,
clem1-l-rss-5-gravity-v1.0, clem1-l-spice-6-v1.0,
clem1-l-u-5-dim-basemap-v1.0, clem1-l-u-5-dim-uvvis-v1.0,
clem1-l/e/y-a/b/u/h/l/n-2-edr-v1.0, co-d-cda-3/4/5-dust-v1.0,
co-e/j/s/sw-caps-2-uncalibrated-v1.0, ...

data_set_name The data_set_name element provides the full name given to a data set or a data product. The data_set_name typically identifies the instrument that acquired the data, the target of that instrument, and the processing level of the data. Example value: MR9/VO1/VO2 MARS IMAGING SCIENCE SUBSYSTEM/VIS 5 CLOUD V1.0. See also: data_set_id. Note: In PDS, the data_set_name is constructed according to standards outlined in the Standards Reference. Note: This element is defined in the AMMOS Magellan catalog as an alias for file_name to provide backward compatibility

Type: character

Value: 120-color_lunar_nir_spectrophotometry_data_v1.0,
2001_mars_odyssey_radio_science_raw_data_set_-_ext_v1.0,
2001_mars_odyssey_radio_science_raw_data_set_-_v1.0,
24-color_asteroid_survey, 2mass_asteroid_and_comet_survey_v1.0,
52-color_asteroid_survey, 52_color_asteroid_survey_v1.0,
52_color_asteroid_survey_v2.0,
ames_mars_general_circulation_model_5_lat_lon_variables_v1.0,
ames_mars_general_circulation_model_5_lat_pres_variable_v1.0,
ames_mars_general_circulation_model_5_lat_time_variable_v1.0,
ames_mars_general_circulation_model_5_lat_variables_v1.0,
ames_mars_general_circulation_model_5_time_variables_v1.0,
ames_mars_general_circulation_model_5_topography_v1.0,
anglo-australian_observatory_data_from_sl9_impacts,
arcb/gssr_m_radio_telesc_derived_radar_model_unit_map_v1.0,
arecibo_moon_radio_telesc_resampled_70_cm_radar_mosaic_v1.0,
arecibo_moon_radio_telescope_calibrated_70_cm_radar_v1.0,
arecibo_moon_radio_telescope_derived_12.6_cm_radar_v1.0,
arecibo_venus_radio_telescope_resampled_12.6_cm_radar_v1.0,
array_of_ici_counts_for_stepped_m/q,v, asteroid_3-micron_survey_v1.0,
asteroid_absolute_magnitudes_and_slopes_v1.0,
asteroid_absolute_magnitudes_v10.0,
asteroid_absolute_magnitudes_v2.0, ...

data_set_release_date The data_set_release_date element provides the date when a data set is released by the data producer for archive or publication. In many systems this represents the end of a proprietary or validation period. Formation rule: YYYY-MM-DD Note: In AMMOS, the data_set_release_date element is used to identify the date at which a product may be released to the general public from proprietary access. AMMOS-related systems should apply this element only to proprietary data.

Type: date

data_set_terse_desc A brief description of the data set

Type: character

data_sets The data_sets element identifies the number of data sets contained in a data set collection.

Type: integer

dd_version_id This element identifies the version of a PDS dictionary. Current PDS practice is to identify a data dictionary with the identifier used for the PDS Catalog build in which it resides, e.g., pdscat1r47, pdscat1r48, and so on. This keyword will use the upper case representation of the catalog identifier, e.g., PDSCAT1R47, PDSCAT1R48, etc.

Type: character

Unit: n/a

delimiter_style TBD description

description An account of the resource. Description may include but is not limited to: an abstract, a table of contents, a graphical representation, or a free-text account of the resource. Dublin Core

detailed_catalog_flag The detailed_catalog_flag element is a yes-or-no flag which indicates whether additional information is available for this data set in a detailed-level catalog.

Type: character

Value: n, y

discipline_desc The discipline_desc element describes the discipline identified by the discipline_name element.

Type: character

discipline_name The discipline_name element identifies the major academic or scientific domain or specialty of interest to an individual or to a PDS Node.

Type: character

Value: atmospheres, geosciences, image_processing, imaging_spectroscopy, navigation_ancillary_information_facility, plasma_interactions, radiometry, rings, small_bodies

distributed_by The Nodes distributing the data set.

distributes The distributes slot provides the names of the data sets that this node distributes to the community

distribution_type The DISTRIBUTION_TYPE element identifies the type or category of a data product within a data set release.

Type: character

eastern_most_longitude TBD description

electronic_mail_id The electronic_mail_id element provides an individual's mailbox name on the electronic mail system identified by the electronic_mail_type element.

Type: character

electronic_mail_type The electronic_mail_type element identifies an electronic mail system by name. Example values: TELEMAIL, NSI/DECNET.

Type: character

Value: arpanet, bitnet, decnet, e-mail, gsfc, internat, internet, jems, mail_(gte_telenet), n/a, nasamail, nsfnet, nsi/decnet, span/nsi, tcp/ip, telemail, unk

element_bytes TBD description

element_offset TBD description

element_scaling_factor TBD description

element_type TBD description

element_unit TBD description

exposure_time TBD description

external_standard TBD description

facility_name The facility_name element identifies a department, laboratory, or subsystem that exists within an institution.

Type: character

Value: applied_coherent_technology_corporation, applied_physics_lab, atmospheres_node, branch_of_astrogeology, center_for_space_research, department_of_astronomy, department_of_atmospheric_sciences, earth_and_planetary_remote_sensing_laboratory, geophysics_and_planetary_physics, herzberg_institute_of_astrophysics, kosmochemie, laboratory_for_terrestrial_physics, lunar_and_planetary_laboratory, mars_space_flight_facility, mgs_rs_remote_mission_support_area, multimission_image_processing_subsystem, navigation_ancillary_information_facility, pds_data_distribution_laboratory, pds_geosciences_node, planetary_data_system, radio_science_systems_group, space_science_laboratory, tes_operations_facility, the_blackett_laboratory

fax_number The fax_number data element provides the area code and telephone number needed to transmit data to an individual or a node via facsimile machine.

Type: character

field_format TBD description

field_location TBD description

field_name TBD description

field_type TBD description

field_unit TBD description

file_name The file_name element provides the location independent name of a file. It excludes node or volume location, directory path names, and version specification. To promote portability across multiple platforms, PDS requires the file_name to be limited to an 27-character basename, a full stop (. period), and a 3-character extension. Valid characters include capital letters A - Z, numerals 0 - 9, and the underscore character (_).

Type: character

files The files element identifies the total number of files. Note: As an example in the PDS, the keyword files within the Directory Object indentifies the total number of files in the directory. Within the Volume Object the keyword files identifies the number of files within the volume.

Type: integer

filter_parameters TBD description

first_standard_parallel The first_standard_parallel element is used in Conic projections. If a Conic projection has a single standard parallel, then the first_standard_parallel is the point of tangency between the sphere of the planet and the cone of the projection. If there are two standard parallels (first_standard_parallel, second_standard_parallel), these parallel are the intersection lines between the sphere of the planet and the cone of the projection. The map_scale is defined at the standard parallels.

Type: real

Unit: deg

frequency_field TBD description

frequency_unit TBD description

full_name The full_name element provides the complete name or identifier for a person or object. For an individual, full name includes the name as well as titles and suffixes. For an object, full name provides the spelled-out name that in some cases corresponds to an 'id'.

Type: character

hardware_model_id The hardware_model_id element identifies the computer hardware on which a data product was produced. (e.g. VAX 11/780, MACINTOSH II).

Type: identifier

Value: macintosh, macintosh_ii, pc, sun_3, sun_4, sun_sparc_station, tdds, vax_11/750, vax_11/780

has_DCS Composition association for classes that describing the product.

has_DDE Composition association for Descriptive Data Elements. Used to added arbitrary data elements for describing a data product.

has_Electronic_Mail Composition association for Electronic Mail

has_IDE Composition association for Identification Elements

has_Inventory_Data_Set_Info Composite association for Inventory Data Set Information.

has_LSI Composition association for Label Standards Identifiers

has_Node_Inventory Composite association for Node

has_Software_Online Composite association for software_online

has_TFE_TDO Composition association for Tagged_File_Explicit

has_TFI_TDO Composition association for Tagged File Implicit. Tagged File Implicit in turn has an association relationship with other tagged_data_objects. This modeling approach makes the Implicit File symmetric to the Explicit File.

has_inventory_Node_Media_Info Composite association for Inventory_Node_Media_Information.

horizontal_framelet_offset The horizontal_framelet_offset provides the row number of a framelet within a tiled image. In the PDS, offsets are counted from one.

Type: real

image_id The image_id element is used to identify an image and typically consists of a sequence of characters representing 1) a routinely occurring measure, such as revolution number, 2) a letter identifying the spacecraft, target, or camera, and 3) a representation of a count within

the measure, such as picture number within a given revolution. Example: Mariner 9 - Levanthal Identifier - (orbit, camera, pic #, total # of pics in orbit) Viking Orbiter - (orbit #, sc, pic # (FSC/16)), Viking Lander - (sc, camera, mars doy, diode (filter), pic # for that day), Voyager - (pic # for encounter, FDS for cruise) Note: For Mars Pathfinder, this uniquely identified the observation parameters of an image. The most significant four digits identified the command sequence that contained the imaging command. The middle two digits indicated the version of the command sequence, and the right four digits identified the image within a single imaging sequence. If the image_id was even and non-zero, it was a left frame image. If the image_id was one greater than the left frame image_id (and therefore odd), it was the right frame of a stereo image. Note that during operations, a small number of image_ids were re-used with difference command parameters. This eliminated the uniqueness of the image_id for those images. The tlm_cmd_discrepancy_flag may be useful in identifying the images that had this problem.

Type: character

instrument_desc The instrument_desc element describes a given instrument.

Type: character

instrument_host_desc The instrument_host_desc data element describes the spacecraft or earthbase from which particular instrument measurements were taken. For spacecraft, this description addresses the complement of instruments carried, the on-board communications and data processing equipment, the method of stabilization, the source of power and the capabilities or limitations of the spacecraft design which are related to data-taking activities. The description may be a synopsis of available mission documentation.

Type: character

instrument_host_id The instrument_host_id element provides a unique identifier for the host where an instrument is located. This host can be either a spacecraft or an earth base (e.g., and observatory or laboratory on the earth). Thus, the instrument_host_id element can contain values which are either spacecraft_id values or earth_base_id values.

Type: identifier

Value: 24col, aao, amon, arcb, astr, austc14, c130, c154, clem1, co, ctio, ctio15, ctio15m, ctioppt, dif, dii, ds1, ecas, er-2, eso, eso1m, eso22m, fexp, gdscc, gio, ...

instrument_host_name The instrument_host_name element provides the full name of the host on which an instrument is based. This host can be either a spacecraft or an earth base. Thus, the instrument_host_name element can contain values which are either spacecraft_name values or earth_base_name values.

Type: character

Value: 2001_mars_odyssey, 24-color_survey, ames_mars_general_circulation_model, apache_point_observatory_2.5-m_sdss_ritchey-chretien_altazimuth_reflector, apache_pt_obs._2.5m_sdss_ritchey-chretien_altazimuth_refl, arecibo_observatory, arecibo_observatory_305-m_fixed_spherical_reflecting_antenna, cassini_orbiter, cerro_tololo_inter-american_observatory_1-m_boller_&_chivens_ritchey-chretien_reflector, cerro_tololo_inter-american_observatory_1.5-m_ritchey-chretien_cassegrain_reflector, cerro_tololo_inter-american_observatory_1.5_meter, cerro_tololo_inter-american_observatory_2mass_1.3m_telescope, cerro_tololo_interamerican_observatory, clementine_1, ctio_1.5m_telescope, ctio_planetary_patrol_telescope, deep_impact_flyby_spacecraft, deep_impact_impactor_spacecraft, deep_space_1, eight_color_asteroid_survey, el_leoncito_astronomical_complex_2.15-m_boller_&_chivens_reflector, european_southern_observatory, european_southern_observatory_1-m_telescope, european_southern_observatory_1.52-m_spectrographic_cassegrain/coude_reflector, european_southern_observatory_2.2-m_telescope, ...

instrument_host_type The instrument_host_type element provides the type of host on which an instrument is based. For example, if the instrument is located on a spacecraft, the instrument_host_type element would have the value SPACECRAFT.

Type: character

Value: data_base, earth_based, n/a, rover, spacecraft, unk

instrument_id_new The instrument_id element provides an abbreviated name or acronym which identifies an instrument. The proposed new instrument id conceptually combines the old instrument_id with the instrument_host_id.

instrument_name The instrument_name element provides the full name of an instrument. Note: that the associated instrument_id element provides an abbreviated name or acronym for the instrument. Example values: FLUXGATE MAGNETOMETER, NEAR_INFRARED MAPPING SPECTROMETER.

Type: character

Value: 2_channel_photometer, 2mass_camera_-_north, 2mass_camera_-_south, 8_color_photometric_system, a_star_tracker_camera, accelerometer, adv._solid-state_array_spectroradiometer, advance_camera_for_surveys, aerosol_collector_pyrolyser, airborne_visible/ir_imaging_spectrometer, airsar, alpha_particle_spectrometer, alpha_particle_x-ray_spectrometer, alpha_proton_x-ray_spectrometer, amateur_photography, amateur_spectrographs, amateur_visual_observations, aperture_photometer, arecibo_radar_data, atmospheric_structure_instrument, atmospheric_structure_instrument_/_meteorology_package, auxiliary_port_imager, b_star_tracker_camera, beckman_dk2a_ratio_recording_spectroreflectometer, boller_&_chivens_spectrograph, ...

instrument_type The instrument_type element identifies the type of an instrument. Example values: POLARIMETER, RADIOMETER, REFLECTANCE SPECTROMETER, VIDICON CAMERA.

Type: character

Value: 3-color_pushbroom_imager, abrader, accelerometer, acoustic_sensor, anemometer, antennae, atmospheric_profiler, attitude_control_system, barometer, beta_detector, camera, ccd, ccd/spectrograph, ccd_camera, charged_particle_analyzer, charged_particle_telescope, computation, cosmic_dust_analyzer, cosmic_ray_detector, detector_array, dosimeter, drill, dust_detector, dust_impact_detector, dust_sample_collector, ...

inventory_special_order_note The inventory_special_order_note element is a text field that provides information on special orders that can be placed for a given data set collection or data set.

Type: character

is_affiliated_with The are_affiliated_with slot provides the names of personnel associated with a node.

kernel_type_new TBD description

label_revision_note TBD description

last_name The last_name element provides the last name (surname) of an individual.

Type: character

line_first_pixel The line_first_pixel element provides the line index for the first pixel that was physically recorded at the beginning of the image array. Note: In the PDS, for a fuller explanation on the use of this data element in the Image Map Projection Object, please refer to the PDS Standards Reference.

Type: integer

line_last_pixel The line_last_pixel element provides the line index for the last pixel that was physically recorded at the end of the image array. Note: In the PDS, for a fuller explanation on the use of this data element in the Image Map Projection Object, please refer to the PDS Standards Reference.

Type: integer

line_projection_offset The line_projection_offset element provides the line offset value of the map projection origin position from the line and sample 1,1 (line and sample 1,1 is considered the upper left corner of the digital array). Note: that the positive direction is to the right and down.

Type: real

Unit: pixel

logical_volume_path_name The logical_volume_path_name element is a character string or set of character strings giving the root directory path for each logical volume. If missing, the volume begins in the root directory as usual.

Type: character

logical_volumes The logical_volumes element is an integer indicating the number of logical volumes in the given volume. If it is missing, it has a default value of 1.

Type: integer

map_projection_desc The map_projection_desc element describes the map_projection_type unambiguously. It shall contain the mathematical expressions (it may even contain the source code or pseudo code, with comments) and any assumptions (e.g. the planet is assumed spherical). Additionally it shall describe the planet eccentricity, the treatment of the a_axis_radius, b_axis_radius, and c_axis_radius when the projection was created, and where the map_scale (or map_resolution) is defined.

Type: character

map_projection_rotation The map_projection_rotation element provides the clockwise rotation, in degrees, of the line and sample coordinates with respect to the map projection origin (line_projection_offset, line_projection_offset) This parameter is used to indicate where 'up' is in the projection. For example, in a polar stereographic projection does the zero meridian go center to bottom, center to top, center to left, or center to right? The polar projection is defined such that the zero meridian goes center to bottom. However, by rotating the map projection, the zero meridian can go in any direction. Note: 180 degrees is at the top of the North Pole and 0 degrees is at the top of the South Pole. For example, if 0 degrees is at the top of the North Pole than the map_projection_rotation would be 180 degrees.

Type: real

Unit: deg

map_projection_type The map_projection_type element identifies the type of projection characteristic of a given map. Example value: ORTHOGRAPHIC.

Type: character

Value: aitoff, albers, bonne, briesemeister, cylindrical_equal_area, equidistant, equirectangular, gnomonic, hammer, hendu, lambert_azimuthal_equal_area, lambert_conformal, mercator, mollweide, oblique_cylindrical, orthographic, polar_stereographic, simple_cylindrical, sinusoidal, stereographic, transverse_mercator, van_der_grinten, werner

map_resolution The map_resolution element identifies the scale of a given map. Please refer to the definition for map_scale for a more complete definition. Note: map_resolution and map_scale both define the scale of a map except that they are expressed in different units: map_resolution is in PIXEL/DEGREE and map_scale is in KM/PIXEL.

Type: real

Unit: pix/deg

map_scale The map_scale element identifies the scale of a given map. The scale is defined as the ratio of the actual distance between two points on the surface of the target body to the distance between the corresponding points on the map. The map_scale references the scale of a map at a certain reference point or line. Certain map projections vary in scale throughout the map. For example, in a Mercator projection, the map_scale refers to the scale of the map at the equator. For Conic projections, the map_scale refers to the scale at the standard parallels. For an Orthographic point, the map_scale refers to the scale at the center latitude and longitude. The relationship between map_scale and the map_resolution element is that they both define the scale of a given map, except they are expressed in different units: map_scale is in KM/PIXEL and map_resolution is in PIXEL/DEGREE. Also note that one is inversely proportional to the other and that kilometers and degrees can be related given the radius of the planet: 1 degree = (2 * RADIUS * PI) / 360 kilometers.

Type: real

Unit: km/pix

maximum The maximum element indicates the largest value occurring in a given instance of the data object. Note: For PDS and Mars Observer

applications – because of the unconventional data type of this data element, the element should appear in labels only within an explicit object, i.e. anywhere between an 'OBJECT =' and an 'END_OBJECT'.

Type: context_dependent

maximum_latitude The maximum_latitude element specifies the northernmost latitude of a spatial area, such as a map, mosaic, bin, feature, or region. See latitude.

Type: real

Unit: deg

maximum_record_length TBD description

median The median element provides the median value (middle value) occurring in a given instance of the data object. Because of the unconventional data type of this data element, the element should appear in labels only within an explicit object, i.e. anywhere between an 'OBJECT =' and an 'END OBJECT'. Note: For the Mars Pathfinder IMP camera, this was the median value of only those pixels within the valid DN range of 0 to 4095. Note: For Mars Pathfinder, refers specifically to the median DN value in the image array.

Type: real

medium_desc The medium_desc element provides the textual description for the medium used in the distribution of an ordered data set.

Type: character

medium_format The medium_format element identifies the unformatted recording capacity or recording density of a given medium.

Type: identifier

Value: 1.0_mb, 1.6_mb, 150_mb, 1600_bpi, 1_gb, 2.0_mb, 2_gb, 30_mb, 360_kb, 5_gb, 60_mb, 6250_bpi, 650_mb, 800_bpi

medium_type The medium_type element identifies the physical storage medium for a data volume. Examples: CD-ROM, CARTRIDGE TAPE.

Type: character

Value: 12-in_worm_disk, 14-in_worm_disk, 19-mm_helical_scan_tape, 3.5-in._floppy_disk, 3.5-in_magneto-optic_disk, 4-mm_helical_scan_tape, 5.25-in_floppy_disk, 5.25-in_magneto-optic_disk, 5.25-in_worm_disk, 7-track_mag_tape, 8-mm_helical_scan_tape, 9-track_mag_tape, cartridge_tape, cd-rom, cd-wo, dvd-r, dvd-rom, electronic, mag_tape, magnetic_tape, n/a, null, photo, tape

minimum The minimum element indicates the smallest value occurring in a given instance of the data object. Note: For PDS and Mars Observer applications – because of the unconventional data type of this data element, the element should appear in labels only within an explicit object, i.e. anywhere between an 'OBJECT =' and an 'END_OBJECT'.

Type: context_dependent

minimum_latitude The minimum_latitude element specifies the southernmost latitude of a spatial area, such as a map, mosaic, bin, feature, or region. See latitude.

Type: real

Unit: deg

mission_alias_name The mission_alias_name element provides an official name of a mission used during the initial design, implementation, or prelaunch phases. Example values: mission_name:MAGELLAN, mission_alias_name:VENUS RADAR MAPPER. The mission_alias_name element accepts set notation for multiple values.

Type: character

Value: cassini, clementine_1, comet_impact_94, di, galileo_europa_mission_(gem), galileo_millennium_mission_(gmm), hubble_space_telescope, huygens, international_solar_polar_mission, international_sun-earth_explor, international_uv_explorer, iras, jupiter_orbiter-probe_(jop), mariner_10, mariner_6_&_7, mariner_9, mars_environmental_survey, mars_environmental_survey_(mesur_pathfinder), mgs, mjs77, mro, ms-t5, msx, n/a, near, ...

mission_desc The mission_desc element summarizes major aspects of a planetary mission or project, including the number and type of spacecraft, the target body or bodies and major accomplishments.

Type: character

mission_name The mission_name element identifies a major planetary mission or project. A given planetary mission may be associated with one or more spacecraft.

Type: character

Value: 2001_mars_odyssey, asteroid_observations, cassini-huygens, cassini-huygens_mission_to_saturn_and_titan, comet_sl9/jupiter_collision, deep_impact, deep_space_1, deep_space_program_science_experiment, galileo, geologic_remote_sensing_field_experiment, giotto, ground_based_atmospheric_observations, hst, ihw, infrared_astronomical_satellite, international_cometary_explorer, international_halley_watch, international_ultraviolet_explorer, iue, lunar_prospector, magellan, mariner69, mariner71, mariner_10, mars_environmental_survey_(mesur_pathfinder), ...

mission_objectives_summary The mission_objectives_summary element describes the major scientific objectives of a planetary mission or project.

Type: character

mission_start_date The mission_start_date element provides the date of the beginning of a mission in UTC system format. Formation rule: YYYY-MM-DDThh:mm:ss[.fff]

Type: date

mission_stop_date The mission_stop_date element provides the date of the end of a mission in UTC system format. Formation rule: YYYY-MM-DDThh:mm:ss[.fff]

Type: date

name The name data element indicates a literal value representing the common term used to identify an element or object. See also: 'id'. Note: In the PDS data dictionary, if the name identifier is prepended

with a namespace identifier (e.g., CASSINI:TARGET_NAME), then the name identifier is restricted to 61 characters where the name identifier and the namespace identifiers are each restricted to 30 characters and are separated by a colon (for a total maximum length of 61 characters). The name identifier and its component parts must conform to PDS nomenclature standards. If the name identifier is used without a namespace identifier (e.g., TARGET_NAME), then the name identifer is restricted to 30 characters, and must conform to PDS nomenclature standards.

Type: character

node_desc The node_desc element describes a PDS Node.

Type: character

node_id The node_id element provides the node id assigned to a science community node.

Type: character

Value: atmos, en, esa, geoscience, hq, imaging, imaging-jpl, n/a, naif, nssdc, ppi-ucla, rad, rings, rs, sbn

node_institution_name The node_institution_name element identifies a university, research center, NASA center or other institution associated with a PDS node.

Type: character

Value: european_space_agency, goddard_space_flight_center, hq, jet_propulsion_laboratory, johns_hopkins_university_applied_physics_laboratory, massachusetts_institute_of_technology, n/a, nasa/ames_research_center, new_mexico_state_university, seti_institute, stanford_university, united_states_geological_survey, university_of_california,_los_angeles, university_of_hawaii, university_of_iowa, university_of_maryland, washington_university

node_manager_pds_users_id The node_manager slot indicates which person manages the node.

node_name The node_name element provides the officially recognized name of a PDS Node.

Type: character

Value: central, engineering, european_space_agency, geosciences, hq, imaging, n/a, national_space_science_data_center, navigation_ancillary_information_facility, planetary_atmospheres, planetary_plasma_interactions, planetary_plasma_interactions_-_ucla, planetary_rings, radio_science, radiometry, small_bodies

note The note element is a text field which provides miscellaneous notes or comments (for example, concerning a given data set or a given data processing program).

Type: character

number_of_axes TBD description

number_of_fields TBD description

number_of_records TBD description

offset The offset element indicates a shift or displacement of a data value. See also: scaling_factor. Note: Expressed as an equation: true value = offset value + (scaling factor x stored value).

Type: context_dependent

on_line_identification The on_line_identification element is a unique identifier for product resources which are on-line. It may be a URL to a home page, an e-mail address, an ftp site or a jukebox. An on_line_identification element may be associated with a data set, data set collection, mission, instrument, host, target or volume.

Type: character

on_line_name The on_line_name element is a unique name which corresponds to a given on_line_identification element. It is used to create HTML links to appropriate home pages.

Type: character

operating_system_id The operating_system_id element identifies the computer operating system and version of the operating system on which data were manipulated, (e.g., VMS 4.6,UNIX SYSTEM 5, DOS 4.0, MAC).

Type: character

Value: dos_3.3, dos_4.0, mac, os/2, unix_4.2_bsd, unix_system_5, vms_4.6

operations_contact_pds_users_id The operations_contact slot indicates the person responsible for node operations.

orbit_direction The orbit_direction element provides the direction of movement along the orbit about the primary as seen from the north pole of the 'invariable plane of the solar system', which is the plane passing through the center of mass of the solar system and perpendicular to the angular momentum vect or of the solar system orbit motion. PROGRADE for positive rotation according to the right-hand rule, RETROGRADE for ne gative rotation. See also: orbital_inclination

Type: character

Value: n/a, prograde, retrograde, unk, unknown

pds_address_book_flag The pds_address_book_flag data element indicates whether or not a registered PDS user will have an entry in the PDS telephone directory.

Type: character

Value: n, null, y

pds_affiliation The pds_affiliation data element describes the type of relationship an individual has with a PDS node. (e.g., staff, advisory group, etc..)

Type: character

pds_user_id The pds_user_id element provides a unique identifier for each individual who is allowed access to the PDS. The system manager at the Central Node assigns this identifier at the time of user registration.

Type: character

pds_version_id The PDS_version_id data element represents the version number of the PDS standards documents that is valid when a data product label is created. Values for the PDS_version_id are formed by appending the integer for the latest version number to the letters 'PDS'. Examples: PDS3, PDS4.

Type: identifier

Value: pds3, pds4

platform The platform element describes the available platforms which the software supports.

Type: identifier

Value: ibm/dos, mac/osx, multiple, sun/sunos, sun_10/solaris, sun_2/sunos, vax/vms

positive_longitude_direction The positive_longitude_direction element identifies the direction of longitude (e.g. EAST, WEST) for a planet. The IAU definition for direction of positive longitude is adopted. Typically, for planets with prograde rotations, positive longitude direction is to the WEST. For planets with retrograde rotations, positive longitude direction is to the EAST. Note: The positive_longitude_direction keyword should be used for planetographic systems, but not for planetocentric.

Type: identifier

Value: east, west

primary_body_name The primary_body_name element identifies the primary body with which a given target body is associated as a secondary body.

Type: character

Value: ceres, comet, earth, galaxy, halley, jupiter, mars, n/a, neptune, p/grigg_skjellerup, pluto, saturn, sl9, solar_system_barycenter, sun, unk, uranus

producer_full_name The producer_full_name element provides the full_name of the individual mainly responsible for the production of a data set. See also: full_name. Note: This individual does not have to be registered with the PDS.

Type: character

product_creation_time The product_creation_time element defines the UTC system format time when a product was created. Formation rule: YYYY-MM-DDThh:mm:ss[.fff]

Type: time

product_data_set_id The product_data_set_id element provides the data_set_id of a cataloged data set that resulted from the application of the processing software to the source data sets. The data set name associated with the product data set is provided by the data_set_name element.

Type: character

product_id The product_id data element represents a permanent, unique identifier assigned to a data product by its producer. See also: source_product_id. Note: In the PDS, the value assigned to product_id must be unique within its data set. Additional note: The product_id can describe the lowest-level data object that has a PDS label.

Type: character

product_type The PRODUCT_TYPE data element identifies the type or category of a product within a data set. Examples: EDR, DOCUMENT, CALIBRATION_IMAGE, SPICE_SP_KERNEL, TRAJECTORY.

Type: identifier

Value: aedr, agk, amd, ancillary, annotated_tiff, apxs_edr, apxs_xrc, asp, astrometry_table, averaged_hend_data, averaged_neutron_data, bck, bro, browse, bsp, btr, c1-midr, c2-midr, c3-midr, cahv_lin_rdr, calibrated_1d_spectrograph, calibrated_image, calibrated_quality_mask, calibration, calibration_model, ...

property_map Association to Property_Map

protocol_type The protocol_type element identifies the protocol type for the on_line_identification element. Example value: URL, FTP, E-MAIL.

Type: character

Value: ftp, url

publication_date The publication_date element provides the date when a published item, such as a document or a compact disc, was issued. Formation rule: YYYY-MM-DD

Type: date

record_bytes The record_bytes element indicates the number of bytes in a physical file record, including record terminators and separators. When RECORD_BYTES describes a file with RECORD_TYPE = STREAM (e.g. a SPREADSHEET), its value is set to the length of the longest record in the file. Note: In the PDS, the use of record_bytes, along with other file-related data elements is fully described in the Standards Reference.

Type: integer

record_type The record_type element indicates the record format of a file. Note: In the PDS, when record_type is used in a detached label file it always describes its corresponding detached data file, not the label file itself. The use of record_type along with other file-related data elements is fully described in the PDS Standards Reference.

Type: identifier

Value: fixed_length, stream, undefined, variable_length

refer_Catalog Associated Catalog.

refer_Catalog_Data_Set Associated Data Sets. The relationship is implemented using a catalog pointer.

refer_Catalog_Data_Set_Collection Associated Data Set Collection. The relationship is implemented using a catalog pointer.

refer_Catalog_Instrument Associated Instrument. The relationship is implemented using a catalog pointer.

refer_Catalog_Instrument_Host Associated Instrument Host. The relationship is implemented using a catalog pointer.

refer_Catalog_Mission Associated Mission. The relationship is implemented using a catalog pointer.

refer_Catalog_Personnel Associated Personnel. The relationship is implemented using a catalog pointer.

refer_Catalog_References Associated Reference. The relationship is implemented using a catalog pointer.

refer_Catalog_Target Associated Target. The relationship is implemented using a catalog pointer.

refer_Data_Producer Associated Data Producer

refer_Data_Set Associated Data Sets

refer_Data_Set_Inventory Associated Data Set

refer_Data_Set_Map_Projection Associated Data Set Map Projection

refer_Data_Set_Projection Associated Data Set

refer_Data_Supplier Associated Data Supplier

refer_Directory Associated Directories. Tree structure allowed.

refer_Discipline Associated Discipline

refer_File Associated File.

refer_Host The associated instrument host. Implemented as a many-to-many relationship using Instrument_Host_Id, Instrument_Id and and Data_Set_Id. This one many-to-many relationship is used for both Instrument_Host and Instrument relationships with Data_Set.

refer_Host_I The associated data sets. Implemented as a many-to-many relationship using Instrument_Host_Id, Instrument_Id and and Data_Set_Id. This one many-to-many relationship is used for both Instrument_Host and Instrument relationships with Data_Set.

refer_Host_Instrument The associated_host slot provides the names of the instrument_host on which the instrument was mounted.

refer_Host_Instrument_I The mounted_instrument slot provides the names of the instruments associated with an instrument host. Implemented as a many-to-many relationship using Instrument_Host_Id and Instrument_Id. Instrument_Host_Id has two roles, unique identifier for Instrument_Host and part of the unique compound identifier for Instrument.

refer_Instrument The associated instrument. Implemented as a many-to-many relationship using Instrument_Host_Id, Instrument_Id and and Data_Set_Id. This one many-to-many relationship is used for both Instrument_Host and Instrument relationships with Data_Set.

refer_Instrument_I Associated data set. Implemented as a many-to-many relationship using Instrument_Host_Id, Instrument_Id and and Data_Set_Id. This one many-to-many relationship is used for both Instrument_Host and Instrument relationships with Data_Set.

refer_Mission The associate mission. Implemented as a many-to-many relationship using Mission_Name and Data_Set_Id.

refer_Mission_Host Implemented as a many-to-many relationship using Mission_Name and Instrument_Host_Id.

refer_Mission_I The associated data set. Implemented as a many-to-many relationship using Mission_Name and Data_Set_Id.

refer_Product_Implicit Associated Data Products. Used only in catalog implementation and derived from inverse relation.

refer_Reference Associated documents implemented via bibliographic citation.

refer_Reference_Projection Associated Reference

refer_Resource The associated resource. Implemented as a many-to-many relationship using Resource_Id and Data_Set_Id.

refer_Resource_I The associated data set.

refer_Software Associated Software

refer_Target The associated target body. Implemented as a many-to-many relationship using Target_Name and Data_Set_Id.

refer_Target_I The associated data set. Implemented as a many-to-many relationship using Target_Name and Data_Set_Id.

refer_Volume The associated volume. Implemented as a many-to-many relationship using Volume_Id and Data_Set_Id.

refer_Volume_I Associated data set.

reference_desc The reference_desc element provides a complete bibliographic citation for a published work. The format for such citations is that employed by the Journal of Geophysical Research (JGR). This format is described in the JGR, Volume 98, No. A5, Pages 7849-7850, May 1, 1993 under 'References'. Data suppliers may also refer to recent issues of the Journal for examples of citations. Elements of a complete bibliographic citation must include, wherever applicable, author(s) or editor(s), title, journal name, volume number, page range and publication date (for journal article citations), or page range, publisher, place of publication, and publication date (for book citations).

Type: character

reference_key_id The reference_key_id element provides the catalog with an identifier for a reference document. Additionally, it may be used in various catalog descriptions, for example in data_set_desc, as a shorthand notation of a document reference. The reference_key_id element is composed according to the following guidelines: 1. if there is an author for the publication, the general rule is: REFERENCE_KEY_ID = <author's last name><year><letter>, where <author's last name> is a maximum of 15 characters, and may need to be truncated. <year> is 4 characters for the year published. <letter> is optional but consists of one character used to distinguish multiple papers by the same author(s) in the same year. The following variations apply: a. If there is one author: <author's last name><year> Example value: SCARF1980 b. If there are two authors: <first author's last name>&<second author's last name> <year> Example value: SCARF&GURNETT1977 c . If there are three or more authors: <first author's last name>ETAL<year> Example value: GURNETTETAL1979 d. If one author has the same last name as another: <author's last name>,<author's first initial> <year published> Example value: FREUD,A1935 e. If the same author(s) published more than one paper in the same year: <author's last name><year><letter> or <first author's last name>&<second author's last name> <year><letter> or <first author's last name>ETAL<year><letter> Example values: SCARF1980A SCARF&GURNETT1977B f. In cases where an initial reference has been catalogued and published on an Archive medium and subsequent references for the same author and same year are needed at a later date, the following rule applies: Leave the original reference as is, and add a letter to the subsequent references starting with the letter 'B' since the original reference will now be assumed to have an implicit 'A'. For example: PFORD1991, PFORD1991B. Note that if the initial reference has only been catalogued and not yet published, then it can be modified such that the 'A' is explicit, i.e. PFORD1991A. 2. If there is no author for the publication, the general rule is: REFERENCE_KEY_ID = <journal name><document identification> where <journal name> is a maximum of 10 characters, and may need to be abbreviated <document identification> is a maximum of 10 characters. This id may consist of a volume number, and/or document or issue number, and/or year of publication. Example values: SCIENCEV215N4532 JGRV88 JPLD-2468

Type: character

reference_latitude The reference_latitude element provides the new zero latitude in a rotated spherical coordinate system that was used in a given map_projection_type.

Type: real

Unit: deg

reference_longitude The reference_longitude element defines the zero longitude in a rotated spherical coordinate system that was used in a given map_projection_type.

Type: real

Unit: deg

registration_date The registration_date element provides the date as of which an individual is registered as an authorized user of the PDS system. Formation rule: YYYY-MM-DD

Type: date

relates_to_data_set Associated data set.

release_date The release_date element provides the date when a data set or portion of a data set is made available for use. Typically this is when the data is on-line and available for access.

Type: date

release_id The RELEASE_ID element identifies the unique identifier associated with a specific release of a data set. All initial releases should use a RELEASE_ID value of '0001'. Subsequent releases should use a value that represents the next increment over the previous RELEASE_ID (e.g., the second release should use a RELEASE_ID of '0002'). Releases are done when an existing data set or portion of a data set becomes available for distribution. Note: The DATA_SET_ID and RELEASE_ID are used as a combined key to ensure all releases are unique.

Type: character

release_medium The release_medium element provides a textual description for the medium used in the distribution of a released data set or portion of a data set. Examples include: CD-ROM, DVD, etc.

Type: character

release_parameter_text The release_parameters_text element provides a list of parameters that identify the data being released. These parameters are formulated so that they can be appended to a data set browser query. The parameters are specific to individual data sets and their associated data set browsers.

Type: character

required_storage_bytes The required_storage_bytes element provides the number of bytes required to store an uncompressed file. This value may be an approximation and is used to ensure enough disk space is available for the resultant file. Note: For Zip file labels, this keyword provides the total size of all the data files in the Zip file after being uncompressed. For the software inventory template, this is often the size of the uncompressed distribution tar file.

Type: character

resource Associated Resources

resource_class The RESOURCE_CLASS element indicates the type of resource associated with the dataset. For the primary browser, the value should always be set to: application.dataSetBrowserP

Type: character

Value: application.catalog, application.datasetbrowser, application.datasetbrowserc, application.datasetbrowserp, application.datasetbrowserx, application.interface, application.targetbrowser, application.website, data.volume, data.volumefuture, data.volumeoffline, data.volumeremote, data.volumesuperceded

resource_id The resource_id element provides an unique indentifier for the resource.

Type: character

resource_link The RESOURCE_LINK element provides the url of a data set browser that allows searching for particular data products or other ancillary files.

Type: character

resource_name The Resource_Name element provides the descriptive name of a resource url as it should appear in the Data Set Search results page.

Type: character

resource_status The RESOURCE_STATUS element indicates the operational status of the resource associated with the dataset. In most cases the value would be UP to indicate an operational data set browser, etc.

Type: character

rotation_direction The rotation_direction element provides the direction of rotation as viewed from the north pole of the 'invariable plane of the solar system', which is the plane passing through the center of mass of the solar system and perpendicular to the angular momentum vector of the solar system. The value for this element is PROGRADE for counter -clockwise rotation, RETROGRADE for clockwise rotation and SYNCHRONOUS for satellites which are tidally locked with the primary. Sidereal_rotation_period and rotation_directio n_type are unknown for a number of satellites, and are not applicable (N/A) for satellites which are tumbling.

Type: identifier

Value: n/a, prograde, retrograde, synchronous, unk, unknown

rotational_element_desc The rotational_element_desc element describes the standard used for the definition of a planet's pole orientation and prime meridian. The description defines the right ascension and the declination values used to define the planet pole, and the spin angle value of the planet referenced to a standard time (typically EME1950 or J2000 time is used). Periodically, the right ascension, declination, and spin values of the planets are updated by the IAU/IAG/COOSPAR Working Group On Cartographic Coordinates and Rotational Elements because an unambiguous definition of a planet's coordinate system requires these values.

Type: character

sample_first_pixel The sample_first_pixel element provides the sample index for the first pixel that was physically recorded at the beginning of the image array. Note: In the PDS, for a fuller explanation on the use of this data element in the Image Map Projection Object, please refer to the PDS Standards Reference.

Type: integer

sample_last_pixel The sample_last_pixel element provides the sample index for the last pixel that was physically recorded at the end of the image array. Note: In the PDS, for a fuller explanation on the use of this data element in the Image Map Projection Object, please refer to the PDS Standards Reference.

Type: integer

sample_projection_offset The sample_projection_offset element provides the sample offset value of the map projection origin position from line and sample 1,1 (line and sample 1,1 is considered the upper left corner of the digital array). Note: that the positive direction is to the right and down.

Type: real

Unit: pixel

second_standard_parallel Please refer to the definition for first_standard_parallel element to see how second_standard_parallel is defined.

Type: real

Unit: deg

sequence_number The sequence_number element indicates a number designating the place occupied by an item in an ordered sequence.

Type: integer

software_desc The software_desc element describes the functions performed by the data processing software. If the subject software is a program library, this element may provide a list of the contents of the library.

Type: character

software_id The software_id element is a short-hand notation for the software name, typically sixteen characters in length or less (e.g., tbtool,lablib3).

Type: character

software_license_type The software_license_type element indicates the licensing category under which this software falls.

Type: identifier

Value: commercial, public_domain, shareware

software_name The software_name element identifies data processing software such as a program or a program library.

Type: character

software_purpose The software_purpose element describes the intended use of the software.

Type: identifier

Value: analysis, browse, copy, data_modeling, development, display, documentation, inventory, management, mathematics, modification, processing, production, reformatting, subsetting, theory, transformation, verification

software_version_id The software_version_id element indicates the version (development level) of a program or a program library.

Type: character

spacecraft_clock_start_count The spacecraft_clock_start_count element provides the value of the spacecraft clock at the beginning of a time period of interest. Note: In the PDS, sclk_start_counts have been represented in the following ways: Voyager - Flight Data Subsystem (FDS) clock count (floating point 7.2) Mariner 9 - Data Automation Subsystem, Mariner 10 - FDS - spacecraft_clock Mars Pathfinder - spacecraft clock

Type: character

spacecraft_clock_stop_count The spacecraft_clock_stop_count element provides the value of the spacecraft clock at the end of a time period of interest.

Type: character

standard_deviation The standard_deviation element provides the standard deviation of the DN values in the image array. Note: For the Mars Pathfinder image data, the standard deviation was calculated using only those pixels within the valid DN range of 0 to 4095.

Type: real

start_time The start_time element provides the date and time of the beginning of an event or observation (whether it be a spacecraft, ground-based, or system event) in UTC. Formation rule: YYYY-MM-DDThh:mm:ss[.fff].

Type: time

stop_time The stop_time element provides the date and time of the end of an observation or event (whether it be a spacecraft, ground-based, or system event) in UTC. Formation rule: YYYY-MM-DDThh:mm:ss[.fff].

Type: time

stream_type TBD description

target_desc The target_desc element describes the characteristics of a particular target.

Type: character

target_name The target_name element identifies a target. The target may be a planet, satellite,ring,region, feature, asteroid or comet. See target_type.

Type: character

Value: 1000_piazzia, 1001_gaussia, 1003_lilofee, 1004_beloposkya, 1005_arago, 1006_lagrangea, 1007_pawlowia, 10094_eijikato, 100_hekate, 1011_laodamia, 1012_sarema, 1013_tombecka, 1014_semphyra, 1015_christa, 1016_anitra, 1017_jacqueline, 1018_arnolda, 10199_chariklo, 1019_strackea, 101_helena, 1020_arcadia, 1021_flammario, 102_miriam, 10_hygiea, 1_ceres, ...

target_type The target_type element identifies the type of a named target. Example values: PLANET, SATELLITE, RING, REGION, FEATURE, ASTEROID, COMET.

Type: identifier

Value: asteroid, calibration, comet, dust, galaxy, globular_cluster, meteorite, meteoroid_stream, meteoroid_stream, n/a, nebula, open_cluster, planet, planetary_nebula, planetary_system, planetary_system, plasma_cloud, reference, ring, satellite, star, star_cluster, sun, terrestrial_sample, trans-neptunian_obj, ...

technical_support_type The technical_support_type element indicates the type of support provided for a piece of software. SOURCE_NAME = PDS CN/S. Hughes.

Type: identifier

Value: full, one_time, prototype

telephone_number The telephone_number element provides the area code, telephone number and extension (if any) of an individual or node. See also: fts_number.

Type: character

transfer_command_text The transfer_command_text element represents the complete command used to create a data volume, such as COPY or BACKUP for tape volumes. It should also include special flags that were used to perform the command (eg. tar -xvf).

Type: character

vertical_framelet_offset The vertical_framelet_offset element provides the column number of a framelet within a tiled image. In the PDS, offsets are counted from one.

Type: real

volume_format The volume_format element identifies the logical format used in writing a data volume, such as ANSI, TAR, or BACKUP for tape volumes and ISO-9660, HIGH-SIERRA, for CD-ROM volumes.

Type: identifier

Value: ansi, high-sierra, iso-9660, iso-9660_level1, iso-9660_level2, none, tar, udf_iso-9660_bridge, vax-backup

volume_id The volume_id element provides a unique identifier for a data volume. Example: MG_1001.

Type: identifier

volume_insert_text The volume_insert_text element provides a text field to be included on the volume insert. The text field should identify the data products or data sets included on the volume. The text field should consist of 8 or fewer lines of text where each line is no more than 60 characters wide.

Type: character

volume_name The volume_name element contains the name of a data volume. In most cases the volume_name is more specific than the volume_set_name. For example, the volume_name for the first volume in the VOYAGER IMAGES OF URANUS volume set is: Volume 1: Compressed Images 24476.54 - 26439.58

Type: character

volume_series_name The volume_series_name element provides a full, formal name that describes a broad categorization of data products or data sets related to a planetary body or a research campaign (e.g. International Halley Watch). A volume series consists of one or more volume sets that represent data from one or more missions or campaigns. For example, the volume series MISSION TO VENUS consists of the following three volume sets: MAGELLAN: THE MOSAIC IMAGE DATA RECORD MAGELLAN: THE ALTIMETRY AND RADIOMETRY DATA RECORD PRE-MAGELLAN RADAR AND GRAVITY DATA SET COLLECTION

Type: character

Value: ames_mars_general_circulation_model, clementine_mission, deep_impact, deep_impact_support_archive, deep_space_1, deep_space_1_mission, di_ground-based_support_archives, dis_volume_ser_name_aa_0001, ds1_data, earth-based_ring_occultations, giant_planet_satellite_astrometry, giotto_extended_mission_project, ground_based_atmospheric_observations, ihw_archive_addenda,

international_halley_watch, iue_comet_database, mars_exploration_rover, mars_gravity, mars_odyssey, mission_to_earth, mission_to_jupiter, mission_to_mars, mission_to_saturn, mission_to_small_bodies, mission_to_the_moon, ...

volume_set_id The volume_set_id element identifies a data volume or a set of volumes. Volume sets are normally considered as a single orderable entity. Examples: USA_NASA_PDS_MG_1001, USA_NASA_PDS_GR_0001_TO_GR_0009

Type: identifier

Value: eu_esa_dsci_gem_0001, n/a, usa_nasa_ihw_hal, usa_nasa_ihw_hal_0001_to_hal_0023, usa_nasa_ihw_hal_0024, usa_nasa_ihw_hal_0025_to_hal_0026, usa_nasa_jpl_coradr_0001, usa_nasa_jpl_coradr_0042, usa_nasa_jpl_coradr_0043, usa_nasa_jpl_coradr_0045, usa_nasa_jpl_coradr_0046, usa_nasa_jpl_coradr_0047, usa_nasa_jpl_coradr_0048, usa_nasa_jpl_coradr_0050, usa_nasa_jpl_coradr_0051, usa_nasa_jpl_coradr_0053, usa_nasa_jpl_coradr_0054, usa_nasa_jpl_coradr_0055, usa_nasa_jpl_coradr_0058, usa_nasa_jpl_coradr_0059, usa_nasa_jpl_coradr_0060, usa_nasa_jpl_coradr_0061, usa_nasa_jpl_coradr_0062, usa_nasa_jpl_coradr_0063, usa_nasa_jpl_coradr_0064, ...

volume_set_name The volume_set_name element provides the full, formal name of one or more data volumes containing a single data set or a collection of related data sets. Volume sets are normally considered as a single orderable entity. For example, the volume series MISSION TO VENUS consists of the following three volume sets: MAGELLAN: THE MOSAIC IMAGE DATA RECORD MAGELLAN: THE ALTIMETRY AND RADIOMETRY DATA RECORD PRE-MAGELLAN RADAR AND GRAVITY DATA SET COLLECTION In certain cases, the volume_set_name can be the same as the volume_name, such as when the volume_set consists of only one volume.

Type: character

Value: clementine:_basemap_mosaic, clementine:_edr_image_archive, clementine:_intermediate_and_reduced_bistatic_radar_data, clementine:_raw_bistatic_radar_data_archive, clementine_basemap_mosaic, clementine_hires_mosaic,

clementine_uvvis_mosaic, comet_halley_archive,
comets_crommelin_and_giacobini-zinner_archive,
dtm/mdim:_global_coverage,
electron_temperature_probe_processed_data_sets,
fields_and_particles_data_sets,
galileo:_near_infrared_mapping_spectrometer_(nims)_cube_dat,
galileo:_near_infrared_mapping_spectrometer_(nims)_cube_data,
galileo:_near_infrared_mapping_spectrometer_(nims)_edr_data,
galileo:_raw_radio_science_data,
galileo_earth/moon_nims_experiment_data_records_v1.0,
galileo_probe_archive, galileo_solid_state_imaging_orbits_11_-_17,
galileo_solid_state_imaging_raw_edr_images,
galileo_venus_nims_experiment_data_records_v1.0,
geologic_remote_sensing_field_experiment,
giotto_extended_mission_archive,
ground_based_atmospheric_observations,
hst/wfpc2_saturn_images_through_november_1995, ...

volume_version_id The volume_version_id element indentifies the version of a data volume. All original volumes should use a volume_version_id of 'Version 1'. Versions are used when data products are remade due to errors or limitations in the original volumes (test volumes, for example), and the new version makes the previous volume obsolete. Enhancements or revisions to data products which constitute alternate data products should be assigned a unique volume id, not a new version id. Examples: Version 1, Version 2.

Type: character

volumes The volumes element provides the number of physical data volumes contained in a volume set. Note: In the PDS, volumes represents the total number of related data volumes that comprise a single orderable unit, as represented by the volume_set_id. For Example, the volume set VOYAGER IMAGES OF URANUS has the volume_set_id of USA_NASA_PDS_VG_0001_TO_VG_0003 and the value for volumes would be 3.

Type: integer

western_most_longitude TBD description

16 Glossary

The following glossary contains a list of terms used within this specification and the definitions for those terms.

Abstract_Class An "abstract class" is a class that can not be used to create objects.

Association An "association" is a type of defined relationship between classes.

Attribute An "attribute" is a property or characteristic that allows both identification and distinction.

Cardinality "Cardinality" is the number of values allowed to an attribute or association in a single class. Cardinality in general is stated as a range with a minimum and maximum. For example, an attribute that may be multi-valued will have a cardinality of "1..*". When at least one value is required the minimum cardinality must be at least "1". A cardinality where the minimum and maximum are the same is often shown as the single value. For example, an attribute required to have exactly one value will be shown to have a cardinality of "1".

Class A "class" is the set of attributes which identifies a family. A class is a template from which individual members of each family may be constructed.

Class_Hierarchy A "class hierarchy" is a classification of object types, denoting objects as the instantiations of classes.

Data_Elements A "data element" is a discrete unit of data or metadata. It is an elementary piece of information in a data dictionary.

Entity An "entity" is something that has a distinct, separate existence.

Metadata Metadata is data about data.

Model A "model" is a representation or description designed to show an entity and its composition.

Object An "object" is a specific instance of a class.

17 Review Notes

The following is a general list of notes, anomalies, and things to consider during the design of the next version of the PDS standards. Little attempt has been made to rigorously categorize or analyze the items. They have simply been captured during the development and review of this document.

010_080204_001_DataSetProductMap PDS product labels have Data_Set_Id and Product_Id as required data elements. The definition of Product_Id states that its value must be unique within a data set.

The implication is that the a compound key created from Data_Set_Id and Product_id would create a unique identifier for any product across the PDS archive. This also implies that a product can only belong to one data set. However, since there is no requirement or even recommendation that a product belong to only one data set, the current model allows a product to belong to more than one data set. There are examples of this situation occuring at the nodes.

010_080204_002_Bit_Element_NotDefined The Class Bit_Element is not currently defined in the standards reference.

010_080229_003_MissingIDs_LinkClasses Support links between all Identifiable classes. Consider using the W3C Uniform Resource Identifier (URI). In addition consider the equivalent of the Dublin Core (DC) where the standard elements Identifier, Title, and Alternate exist for each Identifiable Class. If a PDS data element plays the role of the DC Identifier (e.g. Data_Set_Id) or the DC Title (e.g. Data_Set_Name) then the PDS data elements are used. The Identifier would be unique within the PDS. The URI would be unique globally and created from the Identifier. Add unique ids for Software and Document.

010_080229_004_Instrument_Id Added new attribute, instrument_new_id as the identifier for instrument. For legacy instruments consider creating a value by concatenating of the values for instrument_host_id and instrument_id.

010_080229_005_Implicit_File Either eliminate or replace implicit File object. Consider one or more explicit File objects, one for each file that comprises a product. Metadata associated with data objects in a file are associated with the explicit File object referencing that file.

010_080229_006_Class_Hierarchies Class hierarchies are implicit in the PDS model. They should be modeled starting with base classes that include only required attributes. E.g. Gazetteer_Table can not currently be modeled as a subclass of Table.

010_080321_007_Data_Element_Data_Types The current approach to typing data elements needs refinement. Issues include character set and case constraints. Consider definition of data types that specify character sets.

010_080321_008_Attribute_Gouping Need to consider a mechanism for the grouping of attributes. (e.g. axis_* or baind_* data element groupings).

010_080321_009_Attribute_Hierarchy The push toward an explicit model and class hierarchies will tend to make more attributes required.

However the existing flexibility that allows the addition of "single use" local attributes must not be impared. Namespaces and control authorities need to be modeled.

010_080410_010_Structural_Models The ODL concept of subobject, as in a Column object is a subobject of a Table Object, is an intuitive ODL semantic. It is used to imply that Colum is a structural component of Table. Since we are modeling descriptions and not structures, this concept can be modeled using a composition (relation) along with a meaningful relationship name and description. Formal structural models will be addressed TBD.

010_080410_011_Volume_Data_Set The relationships and models between Volume and Data_Set are both logical and physical.

010_080410_012_Dependencies Dependencies between classes and attributes are not modeled. For example, a bit column needs the attribute start_byte from column and the table pointer.

011_080516_013_DS_1_Bit_Column Bit_Column - There is no way to specify dependencies like Bit_Column needing start_byte from Column and the pointer from Table.

011_080516_014_DS_1_Series The inheritance of table_storage_type by spectrum and series is awkward.

011_080516_015_DS_2_Mission_Target The MISSION_TARGET subobject was essentially replace by data_set_mission. However the mission_target_host relation still exists and might contain unique information such as targets visited for which there is no data.

011_080516_016_DS_2_NSSDC_DSID NSSDC_DATA_SET_ID is both an object and a data element name. In any case, the NSSDC interface is changing.

011_080516_017_DS_2_Pallette Pallete - Seems to an anomaly.

011_080516_018_DS_2_Table When the base Table Class (i.e. classes modeled with required keywords only) were eliminated the modeling of the subclasses of table was much more difficult. Note that the description of Index Table states that it is a subclass of table.

011_080516_019_DS_3_Software Software seems to be an anomaly. It is currently used as a catalog object. However, since it describes digital data it would seem to be a data object. Pointer usage is not consistent

011_080516_020_MG_1_SPICE PDS3 SPICE model needs fixing and input from NAIF. Chuck wants a spice data set linked to instrument

however this can not be done since we have not yet modeled data set subclasses

011_080516_021_MG_1_Logical_Volume Model the standard logical Volume organizations and physical layouts.

011_080516_022_MG_1_History HISTORY object describes a section of the data file (pointed to by the HISTORY pointer) that looks like ODL, however it is not actually part of the PDS label. Inside the bytes of the HISTORY object itself, PDS rules do not apply. And note that it is another anomaly that the HISTORY object (and thus the History class) has neither required nor optional elements or sub-objects. It is, actually, just an elaborate pointer to a text file segment.

011_080516_023_MG_1_Index_Table Do dependencies between suggested columns need to be implemented.

011_080516_024_MG_2_Target_Reference_Information TARGET_REFERENCE_INFORMATION is an optional object of TARGET. Specific objects should not allow optional objects.

011_080516_025_MG_2_Volume Is Volume a catalog or a data object.

011_080516_026_AR_1_Uses_Pointer Removed "uses_pointer" from the data description classes. The semantics might be needed since has_pointer suggests containment.

011_080516_027_AR_1_Data_Supplier Data_Supplier is a generic catalog object

011_080516_028_AR_1_Directory Directory is a generic catalog object.

011_080516_029_AR_1_DataSet_DataProduct The relation from data set to data product - The "implicit" relation between data_set and data_product is obvious, might exist in the form of certain index_tables, and can be calculated from the relationship between product and data set. Is this sufficient?

011_080516_030_AR_1_Description_pointer For a description pointer, if the thing pointed to has its own label then there is an implicit undocumented relationship between the two things. Does this need to be explicit.

011_080516_031_AR_1_Multi-valued_pointers Are multi-valued pointers allowed?

011_080516_032_AR_1_Data_Object_Type The data_object_type cardinality is currently one even though many data sets have many data_object_types.

011_080516_033_AR_1_Resources Resource linking to data set. Currently this is done as part of housekeeping. The nodes provide resource_information templates and they are linked by the DE to the data sets.

011_080516_034_AR_1_Projection_Objects
IMAGE_MAP_PROJECTION and DATA_SET_MAP_PROJECTION - There should be an association given the way it is modeled in the specification. A relational (ODL) implementation should use a foreign key, the unique identifier of the map projection.

011_080516_035_AR_1_Data_Supplier The Data_Supplier object has optional data elements. This is allowed since the object was not designed to load a catalog database. This raises a question that will have to be answered in the design phase, after we finish this task. Optional elements and especially elements with cardinalities greater than one have added baggage when a traditional catalog (e.g. relational database) is to be implemented. This has limited us in the past (e.g. data_object_type). It would be nice to consider new implementation options.

011_080516_036_AR_1_Document_Object Also, it is frequently true that in "data" pointers for DOCUMENT objects there is more than one file listed in a single pointer. This is inconsistent with the attributes of the Data_Object_Pointer class in section 9.2 of the IM Spec. SR Chapter 14 ("Pointer Usage") doesn't specifically prohibit this, although it doesn't show any examples of multi-file pointers, but it's a DOCUMENT convention that's been used for years and is included in the sample DOCUMENT object in SR Appendix A. I have also seen cases where sometimes all the files of the same encoding type are grouped together (all the ASCII files, all the PDF files, etc.), and others where all the logical components of a single document are grouped together (the ASCII text and JPEG graphics, e.g.).

011_080516_037_AR_1_has_Product_Implicit This is an anomaly and needs more discussion. There is an explicit association from data product to data set through the use of the foreign key data_set_id in the Identification_Data_Elements class. It could be argued that there is an explicit relationship from data set to data product in an index_table.This was an attempt to model something that seemed obvious.

011_080516_038_AR_1_has_Resource For the implementation of has_Resource, a keyword should exist. It is a housekeeping object and the DE handles the anomaly

011_080516_039_AR_1_Description_Pointer If the thing pointed to by the description pointer has its own label, then there is an association here between products that we've got to document somehow. Even if there isn't another label involved, pointing to an additional file external to both the label and the data in order to provide additional meta-data is a relationship that probably needs to be documented somehow. So I'm thinking maybe one additional pointer class needs to be defined, but I'm not sure you can document its potential associations without pouring cement into the already very muddy waters. - We might want to model the simple case and leave the others as anomalies. Anyway this seems to be a discussion item.

011_080516_040_RJ_1_Bit_Columns BIT_COLUMNS/COLUMNS with ITEMs and the required use of BITS/BYTES.

011_080516_041_RJ_1_PSDD PSDD is currently considered an anomaly and will be readdressed in PDS4.

011_080516_042_RJ_1_Catalog_Pointers Catalog pointers are an anomaly

011_080521_043_MG_3_Label_Revision_Note The requirement that label_revision_note is required within PDS catalog files simply add an attribute without context. The label_revision_note is not an attribute of the class. What is it an attribute of?

011_080605_044_RG_2_Product_Data_Set_Id The data element product_data_set_id used in the inventory node media class should be replaced with data_set_id.

011_080605_045_RG_2_Processing_History Data Set processing history needs to be revisited. At the least, the model needs to support the ability to link an edr data set to its rdr data set and similar cases. Capture of software used, parameters, and other ancillary files needs consideration.

011_080717_046_DS_6_QUBE QUBE - Special requirements are imposed by the ISIS system but the Stds Ref leaves open the question of whether these are also PDS requirements. The text for PDS users should be clarified

011_080717_047_DS_6_Generic_Product The current model for products is generic. It does not have subclasses for Image, Table, etc. Therefore it is difficult to determine the required tagged_data_objects and their associated data objects for a specific product type.

011_080722_048_DS_7_Spec_Qube Spectral_Qube - Conditional keyword anomaly - SUFFIX_BYTES is a required attribute if suffix planes are included

011_080722_049_TK_2_Fig_Constants Figurative constants (e.g. unknown_constant) should be modeled in Column and not Index_Table

011_080722_050_TK_2_TDO_Anomalies All tagged data objects should have one required pointer and one required data_object. In PDS3 Tagged_Text_Object has no required pointer, Document allows more than one data_object, and Implicit_File does not have an explicit pointer.

011_080817_051_CI_1_Sequences The use of sequences as standard values is an anomaly. For example the PSDD defines Band_Sequence with general_data_type = CHARACTER and has a standard value list consisting of ODL sequences, each containing a unique permutation of three values.

011_080817_052_TK_2_2_Data_Types Current data type specifications are not sufficient for defining data elements.

011_080817_053_BS_1_SPICE_PSDD NAIF requests PSDD be removed from the SPICE_Kernel class definition.

011_080817_054_BS_1_SPICE_KERNEL_VV Fix multiple instances of "SPICE_Kernel" in the PSDD valid values list. One exists with blanks the other with underscore.

011_080817_055_EG_1_Attribute_Ordering Attribute Ordering - The implementation requirements on SFDU and PDS_VERSION_ID suggest that the model needs to specify attribute ordering, at least for these two attributes. For the PDS3 model specification this will not be addressed other than to state that the PDS3 standards need to be consulted for the particulars.

011_080817_056_EG_1_Units Units - ¡confirm anomaly exists¿ The PDS3 model specification does not provide "units". The modeling database used to generate the PDS3 model specification includes a copy of the PSDD. Units, min/max values and all other data element definitional information is available. The presentation of units will be taken up as part of PDS4.

011_080817_057_EG_1_Implicit_Assumptions Implicit Assumptions - In general, during the development of the PDS4 data architecture, implicit assumptions are to be made explicit where ever possible. (e.g. Histogram) - "How can the bits be interpreted without knowing if the

values are ascii (unlikely case) or binary? Or how does the model capture assumptions or defaults like this is it assumed that the histogram is binary if interchange format is not listed? The description of histogram in the standards has other assumptions that the model does not capture. For example, if scale and offset are not given, then they are assumed to be 1 and 0.)"

011_080817_058_EG_1_DE_Relationships The PSDD does not capture relationships between data elements such as "has similar meaning" and "has similar valid values".

011_080817_059_EG_1_text_object Text - "Why do we have a text object? Shouldn't it be related to a document? Why does text use note (required) and document use description (optional) as attributes? Why is interchange format optional? Shouldn't text objects always be ascii? Even though EDCDIC is a standard value for interchange_format, would we allow a text object in EDCDIC or a binary text object?"

011_080817_060_EG_1_Compressed_file_objects Compression - Compressed file objects are not handled consistently, especially in regard to legacy data. (e.g. Huffman First Difference) - Consider the subclassing of explicit file for a compressed file class in PDS4. – "There is nothing in the spec about modeling compressed data, whereas the standards ref. has a separate appendix that lists requirements for describing some types of compressed data. The requirements for JPEG2000 and ZIP list two required objects that I don't see in the spec compressed_file and uncompressed_file (and uncompressed_file seems to require the image object). Requirements for other compression methods are not described in the appendix (listed as tbd), but existing usage in PDS does not conform to the way JPEG2000 and ZIP are described."

011_080817_061_EG_1_Browse_Images Browse_Images - "There seems to be an implicit rule in PDS that browse products cannot be described as images because they are compressed and not considered by some to be data. Yet encoding_type is a valid optional attribute for the image object and its standard values includes several compression methods."

011_080817_062_EG_1_Data_Object_Pointer Data_Object_Pointer - In a PDS3 attached label, both file name and offset are not required. Address this issue together with the issue of attached labels.

020_070616_007_Products_Extensions Added Ancillary products comprised of at least one data object and descriptive information about that data object. This subclass of products include software, document, and SPICE products.