PDS3 to PDS4 Jumpstart Guide

DRAFT

Planetary Data System
National Aeronautics and Space Administration

Data Design Working Group

March 2011

Version 0.11.03.24

CHANGE LOG

Revision	Date	Description	Author
0.11.03.08	2011-03-08	Adapted from DPH Ch2 as it existed in Oct 2010	Simpson
0.11.03.11	2011-03-11	Changed “Today” to “Most recently” in 2.1.1	Simpson
0.11.03.11	2011-03-11	Changed last 2 lines in first paragraph of 2.4.1	Simpson
0.11.03.19	2011-03-19	Revised wording previously describing primary and secondary collections	Simpson
0.11.03.19	2011-03-19	Removed references to NSSDC	Simpson
0.11.03.24	2011-03-24	Made corrections requested by Ed Guinness	Simpson
01.11.03.24	2011-03-24	Distinguished bundle from archive bundle in 2.1.2	Simpson

LIEN LOG

Revision	Date	Description
0.11.03.11	2011-03-11	Remove section 2.1.3 — are data set collections common enough that we want to discuss them?
0.11.03.11	2011-03-11	Have the statements in 2.7.2 been accepted by DDWG?

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Purpose, Audience, and Scope

The purpose of this document is to introduce PDS4 to people with a working knowledge of PDS3. It describes similarities and differences between the two systems in terms that should be familiar to users of PDS3.

1.2 Document Overview

1.2.1 Document Outline

After this introduction, Section 2 reviews similarities and differences between PDS3 and PDS4. Section 3 is a summary chart.

1.3 Applicable Documents

[1] Planetary Data System (PDS) PDS4 Information Model Specification, Version 0.2.0.0.Beta, 21 January 2011.
[2] Planetary Data System Standards Reference, Version 4.0.2, 16 December 2010.
[3] Planetary Data System Standards Reference, Version 3.8, 16 February 2009.
[4] Planetary Science Data Dictionary Document, JPL D-7116 Rev. E, 28 August 2002.

2.0 KEY PDS4 CONCEPTS/IMPLEMENTATIONS

This document describes key PDS4 concepts and implementations in terms that should be familiar to users of PDS3. Since PDS4 represents a fundamental change from earlier versions of PDS, the parallels are limited and the document should not be viewed as a 'how-to' manual. Nonetheless, there may be advantages in describing some of the key concepts in PDS4 in familiar terms to experienced users of the planetary archive.

2.1 PDS3 Data Organization

2.1.1 PDS3 Implementation

In PDS3 an important component of each archive is the 'volume' — the unit of media onto which archival data are written and then delivered. During its early years, PDS preferred delivery on magnetic tape; later, the preference changed to CDs and then to DVDs. Most recently a large fraction of data is being delivered electronically; the physical volume has become a 'logical' volume, which is no longer tightly coupled to 'media'.

PDS3 specifies how data are to be organized onto volumes (Figure 2-1), the names of the top level files and directories, and how many copies of each volume are to be delivered — so that there can be both primary and backup copies. When physical media are used, several data sets are occasionally written to a single volume; more commonly, it takes many volumes to capture a single data set.

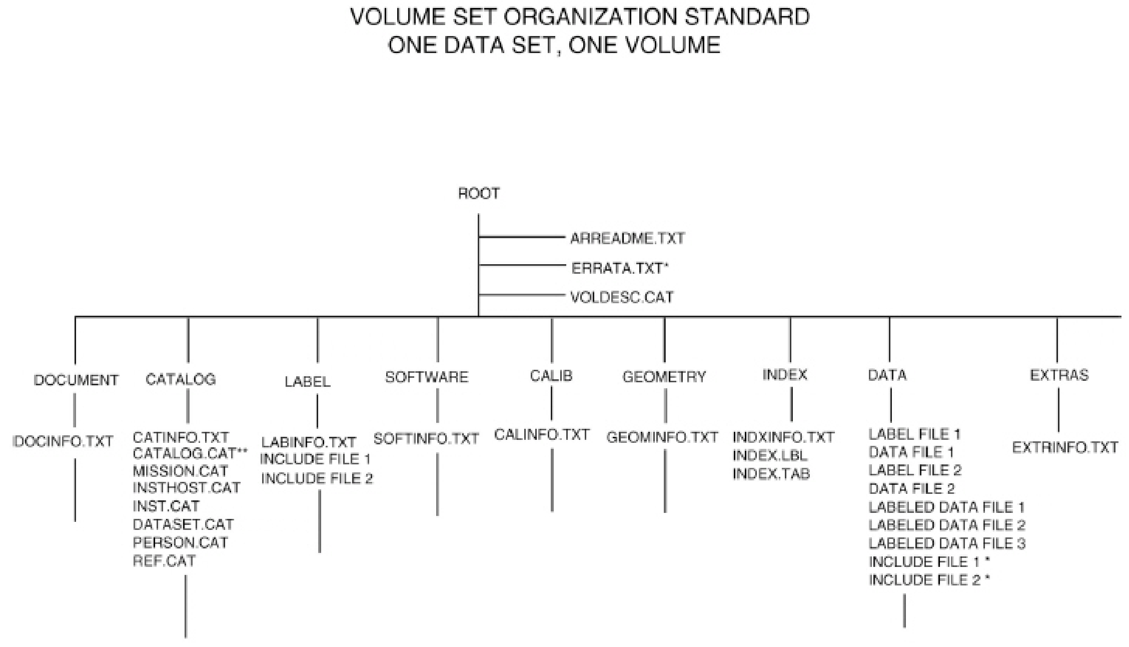


Figure 2-1. One data set, one volume [3]

Whereas the directory structure shown in Figure 2-1 is usually replicated on each physical volume (tape or disc), the ‘elasticity’ of an electronic, logical volume means that new data can simply be added to an existing structure. There is only one logical volume in these modern data sets, which expand (almost) without limit. Files such as INDEX.TAB, AAREADME.TXT, and ERRATA.TXT are updated as needed to reflect changes in the holdings.

The directories in the volume structure (Figure 2-1) are intended to have contents with a common theme. For example, the CALIB directory is expected to contain information that will be needed by future users to calibrate the data, which can be found in the DATA directory (or its subdirectories). The DOCUMENT directory contains documents, which a user might need to read before being able to use and interpret the data; but calibration documents might be found in the CALIB directory instead. There are no fixed rules on these questions, and sometimes community common practices guide the assignments.

Within directories and subdirectories, users find products — each comprising one or more files plus a descriptive ‘label’. The label explains both the structure and content (meaning) of the product, often in a format that is readable by both machines and humans. Products can be data, documentation, software, etc. — either part of the science being archived or the ancillary material needed to understand and use it.

2.1.2 PDS4 Implementation

In PDS4 it is still the product that is at the base of the data tree; but, in PDS4, products are addressable system-wide, whereas in PDS3 it was the data set that was the smallest data unit that could be easily located from the entry portal. Each product has a unique ‘identifier’ that allows it to be found not only in PDS but also within a federation of data systems of which PDS is one member. In addition to the obvious advantages for search and retrieval, the product identifier has important implications for tracking and inventory control.

The unique identifier is one of several important components of PDS4 labels. As in PDS3, it is the label plus one or more files that constitute a product; but the PDS4 label includes other components which can link a product to its source instrument, a mission, and/or an observing campaign (if applicable) The expanded, more structured label means that more can be done with PDS4 products since their context is better documented.

Products are grouped into ‘collections’ — which can be loosely viewed as equivalent to PDS3 directories. In fact, the collection itself is merely a list of names and addresses; the products themselves can be anywhere, locatable by their unique identifiers (and the system’s cross-referencing capabilities). When a product is first delivered to PDS, it is associated with one (and only one) collection; it becomes a permanent ‘primary member’ of this collection. After this initial association, it can be a ‘secondary member’ of an arbitrary number of other collections.

Each collection has its own theme — for example, raw measurements from the Venus Express magnetometer. When *derived* products are delivered, they would presumably go into a different collection than the raw data (although that is not required). It is the choice of the data provider and PDS discipline node to decide what limits are placed on the 'theme'.

An aggregation of collections is called a 'bundle.' If requirements are placed on the bundle to make it suitable for archive (such as scope and quality of documentation), it may be called an 'archive bundle' (Figure 2-2). Like the collection, the bundle is only a list; if the bundle is to be transferred, the list will be read and the collections will be copied (with each collection list in turn being interpreted as file names and locations). In fact, something a little more tangible is needed for real transfers of data; PDS4 has the 'package' and 'container', which are useful for this purpose.

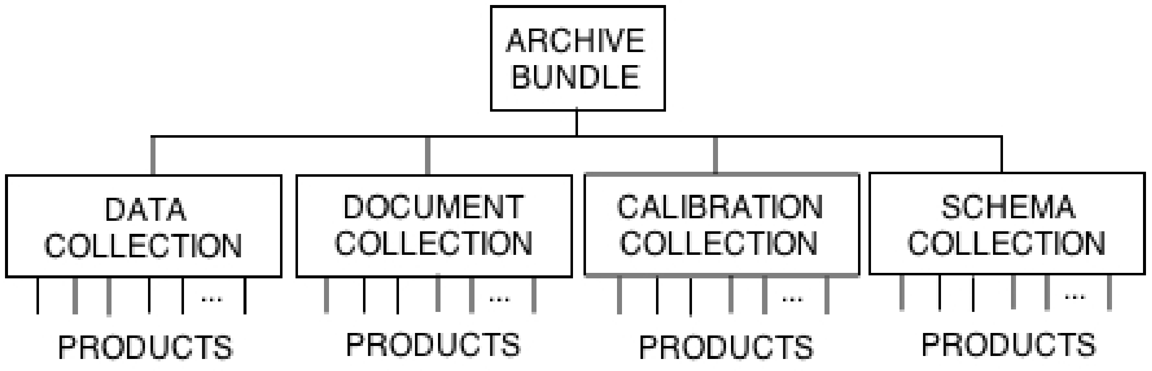


Figure 2-2. Archive bundles are lists of collections, while collections are lists of products. When there is to be a data transfer, the lists and all of the members are wrapped together into a delivery package, which can bear some similarity to a PDS3 volume (Figure 2-1).

Collections with different themes can be assembled into a single bundle. In this way data, documentation, calibrations, and other information can be wrapped together in a way that resembles the PDS3 data set — except that each of the bundle components is a PDS4 collection rather than a PDS3 directory.

Under PDS4, some of the files and directories that were part of the PDS3 volume structure have been deprecated. For example, there is no role for VOLDESC.CAT or the information it carries in a PDS4 collection or bundle. PDS4 has also added some new directories — for example, the SCHEMA directory (Figure 2-2), which contains the XML schemata used to construct PDS4 product labels.

2.1.3 Data Set Collection

A PDS3 data set is an aggregation of products with a common origin, history, or application. A typical data set includes primary (observational) data plus the ancillary data, software, and documentation needed to understand and use the observations. Files in a data set share a unique data set name, share a unique data set identifier, and are described by a single DATASET.CAT catalog object (or equivalent).

A PDS3 data set collection is an aggregation of several data sets that are related by observation type, discipline, target, or time and which can be treated as a unit. Data sets in a data set collection share a unique data set collection name, share a unique data set collection identifier, and are described by a single DATA_SET_COLLECTION object (or equivalent).

One of the primary considerations in creating a data set collection is that the collection provides more utility than the sum of the utilities of the individual data sets.

Whereas data set and archive bundle share several organizational characteristics, there is no parallel to data set collection in PDS4. A bundle can be defined as aggregating collections from several related instruments (or several product types from a single instrument); but there would then be no bundle equivalent to the single data set.

2.2 Labels

A PDS label describes the structure and content (meaning) of its associated product.

2.2.1 PDS3 Implementation

PDS3 labels are written in the Object Description Language (ODL). Although ODL was designed for broader applications, its use has been limited almost exclusively to PDS, which oversees the ODL standard (see Chapter 12 in [3]).

PDS3 labels are constructed in one of three ways:

- Attached: prepended to the associated data file
- Detached: a separate file in the same directory as the data file
- Combined Detached: a separate file in the same directory which logically wraps two or more data files

PDS3 labels typically have line lengths of no more than 80 characters, including an ASCII carriage-return line-feed delimiter.

A PDS3 label includes one or more object definitions, constructed using keyword-value pairs. For example, a column in a table could be defined as follows:

```
OBJECT           = COLUMN
  NAME             = “DETECTOR TEMPERATURE”
  START_BYTE       = 27
  BYTES            = 5
  DATA_TYPE        = ASCII_REAL
  FORMAT           = “F5.1”
  UNIT             = “KELVIN”
  MISSING_CONSTANT = 999.9
  DESCRIPTION      = “Temperature of the detector.”
END_OBJECT       = COLUMN
```

2.2.2 PDS4 Implementation

In PDS4, ODL has been replaced by the eXtensible Markup Language (XML), a set of rules for encoding documents in machine-readable form but which is equally important in web services. XML is widely recognized and supported in the 'open source' environment. Since it accommodates Unicode characters, XML is well-suited for use with international data sets.

Everything in PDS4 is called an 'information object', which is the pairing of a description with a 'data object'. In the PDS3 sense, every label is thus 'detached'. However, physical and conceptual 'data objects' — for example, the planet Saturn or a musical tune — cannot be incorporated into the archive (they are not digital); in these cases, only the label will be found in the archive. Note that an *image* of Saturn and a musical *score* can be digital objects, but the planet and the tune are not.

It is possible to embed data within an XML label. Although this is rare, it is the default in the case of bundle inventory lists (see above). The number of collections included in a bundle is expected to be small, so the list is built into the label rather than appearing as a separate file.

In PDS4 the label is an XML document; its syntax is governed by a 'schema' against which production labels can be validated. There are no restrictions on line length or characters except those imposed by XML.

A PDS4 label includes one or more object definitions, constructed using XML statements. For example, a column in a table (a PDS4 field) could be defined as follows:

```
<Table_Character_Field>
  <field_name>WIND DIRECTION</field_name>
  <field_number>7</field_number>
  <field_data_type>ASCII_REAL</field_data_type>
  <field_location>40</field_location>
  <field_length>10</field_length>
  <field_format>F10.6</field_format>
  <field_min_logical>0.</field_min_logical>
  <field_max_logical>360.</field_max_logical>
  <field_unit>deg</field_unit>
  <field_description>Wind direction in degrees given in
    meteorological convention (0 = from N, 90 = from E,
    180 = from S, 270 = from W) </field_description>
</Table_Character_Field>
```

2.3 Pointers

2.3.1 PDS3 Implementation

Pointers may be used within PDS3 labels to indicate the relative locations of objects in the same file or to reference external files. There are three types of pointers in PDS3:

- Data Location Pointer: identifies the start location of data — for example, by file name and byte number
- Include Pointer: identifies another file which should be substituted for the pointer statement to complete a label
- Related Information Pointer: provides a file name with additional human-readable information — typically used for detailed descriptions found in the PDS3 DOCUMENT directory.

In PDS3 the referenced file is assumed to be in the same directory as the label containing the pointer. Exceptions include the PDS3 LABEL directory for include pointers, the DOCUMENT directory for related information pointers, and the CATALOG directory for related information pointers used in a catalog context.

2.3.2 PDS4 Implementation

PDS4 does not have pointers; instead all file locations are given explicitly — either as an absolute URI to a location somewhere within PDS or its federated partners or by a relative (but explicit) path from the label file. PDS4 has no implicit rules for locating files.

2.4 Record Types

The choice of a record type for a data product is influenced by several factors.

2.4.1 PDS3 Implementation

PDS3 recommends a record format of FIXED_LENGTH or STREAM whenever possible; but it also allows VARIABLE_LENGTH and UNDEFINED. VARIABLE_LENGTH records, although popular in some binary contexts twenty years ago, are used less frequently now, often supplanted by ASCII records in STREAM format. The UNDEFINED record type allows data providers to create PDS3 products having irregular record structure — such as records with no delimiters or which do not begin and end on even byte boundaries.

An additional problem is that UNDEFINED is the only choice when a single label describes two or more data objects with different record types. For example, if a pair of files comprising a single product have FIXED_LENGTH and STREAM formats, respectively, neither value can be used with RECORD_TYPE, which must be set to UNDEFINED.

2.4.2 PDS4 Implementation

Under PDS4, the record type of each digital object is defined explicitly; there is no potential for conflicting record type definitions in a PDS4 label.

In addition PDS4 requires data objects to have a homogenous record structure — they may not have a record structure in which rows or records are interleaved across types of objects (Figure 2-3).

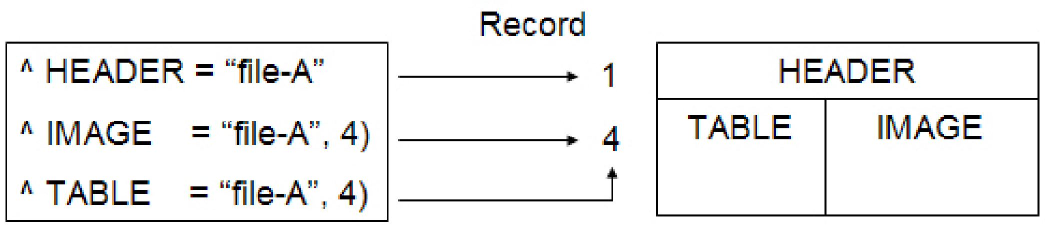
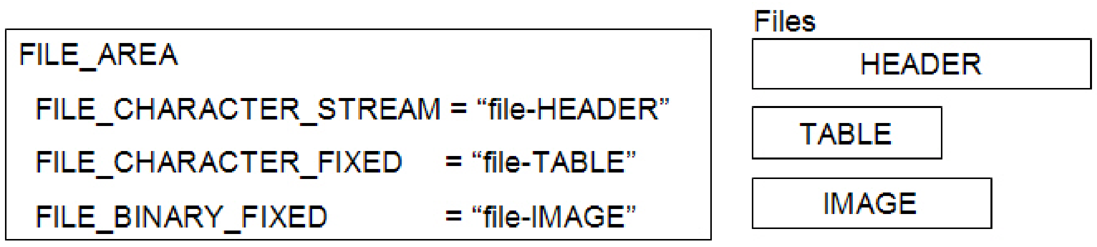
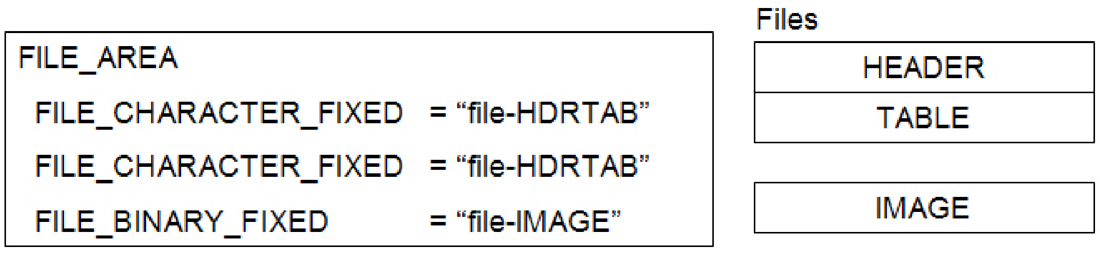


Figure 2-3: It is possible in PDS3 to have interleaved data objects. In this figure, the first part of Record 4 is a table row and the last part of the record is an image row.

Under PDS4, the above example would have to be decomposed into two or more separate files, each having a homogenous record structure.


FILE_AREA
FILE_CHARACTER_STREAM = "file-HEADER"
FILE_CHARACTER_FIXED = "file-TABLE"
FILE_BINARY_FIXED = "file-IMAGE"
Files
HEADER
TABLE
IMAGE

A single file (or pair of files) in which record lengths are identical throughout the file is also acceptable in PDS4. For example the HEADER and TABLE objects could be combined if both objects have identical record lengths.


FILE_AREA
FILE_CHARACTER_FIXED = "file-HDRTAB"
FILE_CHARACTER_FIXED = "file-HDRTAB"
FILE_BINARY_FIXED = "file-IMAGE"
Files
HEADER
TABLE
IMAGE

Since data objects are defined explicitly, it would also be possible to concatenate the three objects above into a single file *with different record lengths* by carefully specifying offsets.

2.5 Data Object Types

PDS4 data object types are a subset of PDS3 types. A goal of PDS4 has been to constrain types to a few fully functional and easily understood formats. This simplifies management and interpretation of data products; more importantly, it facilitates development of PDS software to manipulate and display products once they have been ingested.

2.5.1 PDS3 Implementation

In PDS3 there are 32 data object definitions with both overlap and functional differences. SERIES and SPECTRUM are simply variations of TABLE. DATA_PRODUCER and DATA_SUPPLIER provide ‘fingerprints’ (identify sources) while TABLE defines a true data object.

ALIAS	DATA_PRODUCER	HEADER	SPECTRAL_QUBE
ARRAY	DATA_SUPPLIER	HISTOGRAM	SPECTRUM
BIT_COLUMN	DIRECTORY	HISTORY	SPICE_KERNEL
BIT_ELEMENT	DOCUMENT	IMAGE	SPREADSHEET
CATALOG	ELEMENT	INDEX_TABLE	TABLE
COLLECTION	FIELD	PALETTE	TEXT
COLUMN	FILE	QUBE	VOLUME
CONTAINER	GAZETTEER_TABLE	SERIES	WINDOW

2.5.2 PDS4 Implementation

Under PDS4, there are four fundamental data structures. All data products delivered to the PDS must be constructed from one or more of these structures:

- Table Base (repeating heterogeneous records of scalars),
- Array Base (an N-dimensional, homogeneous array of scalars),
- Parsable Byte Stream (a stream which can be interpreted byte-by-byte)
- Encoded Stream Base (a stream which can be interpreted only after computation)

PDS4 data objects will actually be defined using extensions of these four fundamental structures. For example, there are Product_Table_Character and Product_Table_Binary extensions of Table Base, each with its own set of required and optional attributes, associations, and child classes. But every table in PDS4 can ultimately be traced back to the Table Base definition.

Equivalents to *most* PDS3 data objects can be identified within the tree of PDS4 definitions; but there are some important exceptions. The PDS3 QUBE, which allows complex interleaving of object types, is not possible within PDS4.

Encoded Stream Base is not a catch-all for data which do not fit within the other three categories; there must be supporting software that is widely available, easily used, and maintainable on time scales of decades before a specific encoding scheme is acceptable in PDS4.

2.6 Data Type Representations

Under PDS3, data files contain data in either ASCII or binary formats, of which there are several variations. ASCII formats allow for a more human-readable interpretation but at the cost of substantially more storage space — for example, an 8-bit signed pixel value in a binary image file would require a 4-byte field if stored in ASCII.

Under PDS4, the data may be represented in binary, ASCII, or UTF-8. The addition of UTF-8 allows PDS to capture and manage data sets from sources where English is not the primary language. This applies to both data and metadata; the fact that XML is compatible with UTF-8 is also important. For example, a native Russian could submit data using characters from the Cyrillic alphabet.

```
а б в г д е ё ж з и й к л м н о п р с т у ф х ц ч ш щ ъ ы ь э ю я
```

2.7 Special Values

It is sometimes necessary, during construction of PDS labels and data objects, to indicate that a value exists but is believed to be incorrect, is unknown, or is inappropriate.

2.7.1 PDS3 Implementation

PDS3 provides the symbolic literals "N/A", "UNK", and "NULL", each of which may be used in a label as follows:

- "N/A" indicates that a keyword and value is not applicable. For example, INSTRUMENT_ID is not applicable in a SPICE SPK file label.
- "UNK" indicates that a value is not known and never will be known. For example, FILTER_NAME could be set to "UNK" if the observing log recording the filter name was lost and the name of the filter is not otherwise recoverable.
- "NULL" is used to flag values that are *temporarily* unknown. It indicates that the data preparer recognizes that a specific value should have been entered but that the true value was not readily available. "NULL" values should be replaced by actual values prior to final archiving.

PDS3 also defines five keywords that set values to indicate special conditions:

- `INVALID_CONSTANT`: value indicates that data were outside the valid range
- `MISSING_CONSTANT`: value indicates that no data were available

- `NOT_APPLICABLE_CONSTANT`: value indicates that data were not applicable
- `NULL_CONSTANT`: value indicates that data are *temporarily* unknown
- `UNKNOWN_CONSTANT`: value indicates that data are permanently unknown

2.7.2 PDS4 Implementation

Under PDS4, the terse symbolic literals “N/A”, “UNK”, and “NULL” have been replaced with the more explicit values “not applicable”, “unknown”, and “temporarily not known”. The circumstances under which the values are used remain unchanged under PDS4. The set of attributes for which these values can be specified is limited — *e.g.,* time values, numeric values.

As in PDS3, there are attributes which can be used to set values indicating special conditions in data objects. These ‘special constants’ are:

- error_constant: value indicates that the original value was in error
- invalid_constant: the value indicates that the original value was invalid
- missing_constant: the value indicates that the original value was missing
- not_applicable_constant: the value indicates that the original valkue was not applicable
- saturated_constant: the value indicates that the original value was saturated
- unknown_constant: the value indicates that the original value was unknown

3.0 SUMMARY

The similarities and differences between PDS3 and PDS4 are summarized in the table below.

PDS3	Rough Equivalent in PDS4
Data object (a set of results from an observation, or information needed to use those results)	Data object (a set of results from an observation, or information needed to use those results)
Object description (description of structure and content of data object)	Description object (description of structure and content of data object)
Label (one or more object descriptions plus implementation overhead)	Label (one or more description objects, including identification and cross references, plus implementation overhead)
Product (one or more related data objects plus their label)	Product (one or more related data objects plus their label; the smallest unit of data which is uniquely identified and locatable in PDS4)
Directory (a named storage area containing one or more related products)	Collection (a list of one or more related products)
Data set (an aggregation of data products with a common origin, history, or application and the files needed to use and interpret them; the smallest unit of data which is uniquely identified and locatable in PDS3)	Bundle or archive bundle (a list of one or more related collections)
Data set collection (an aggregation of related data sets)	N/A
Volume (a unit of named media onto which data are written for transfer and/or storage)	Package and container (vehicles by which data are grouped and transferred)
ODL (PDS3 implementation language) `OBJECT      = COLUMN` `NAME       = “TEMPERATURE”` `START_BYTE = 27` `...` `END_OBJECT = COLUMN`	XML (open source rules for encoding documents and data structures in PDS4) `<Table_Character_Field>` `<field_name>WIND DIRECTION</fie` `<field_number>7</field_number>` `...` `</Table_Character_Field>`
Data types: ASCII, binary	Data types: ASCII, binary, UTF-8