Planetary Data System
PDS 2010 Technical Review
Jet Propulsion Laboratory (171-290)
10-11 June 2009

Attendees:

Rafael Alanis	Sean Hardman	Mike Martin
Keith Bennett	John Ho	Paul Ramirez
Mike Cayanan	Lyle Huber	Anne Raugh
Richard Chen	Steve Hughes	Elizabeth Rye
Dan Crichton	Chris Isbell	Boris Semenov
John Dielh	Joni Johnson	Dick Simpson
Josh Ganderson	Steve Joy	Susie Slavney
Patty Garcia	Ron Joyner	Alice Stanboli
Mitch Gordon	Emily Law	Bob Sucharski
Ed Guinness	Hyun Lee	Betty Sword

Introduction:

Called to order by Crichton at 9:08 AM. PPI not present yet. Lunch in cafeteria. Dinner at Cafe Bizou (91 N. Raymond) in Pasadena at 6:45 PM.

Overview:

There will be presentations, but Dan wants discussion. A feedback form is available. Comments will be sorted as data design, software design, or project management; results will be included in the report to August MC. Some comments from non-presenters are highlighted here in red.

Schedule:

November 2008 MC approved broad PDS 2010 objectives with early work being done by working groups on data model and distributed service infrastructure.

Preliminary schedule circulated to MC. Overguide funding has been requested with increments to cover various parts of the PDS 2010 project; this allows incremental implementation.

Goals:

1. Simplified but rigorous archive standards that are consistent, easy to learn, and easy to use.

2. Adaptable tools for designing archives, preparing data, and delivering results efficiently to PDS

3. On-line services should allow users to access and transform data quickly from anywhere in the system.

4. A highly reliable, scalable computing infrastructure that protects data, links nodes, and provides best service to both providers and users.

Move toward a software services model which allows for packaging and reuse of software as services over the net. Identify opportunities to leverage newer standards.

Data Architecture Concepts:

The information model is expressed as classes having data elements (defined by DD). Each is formulated through label schema which are used to create products and labels, which are validated against the original definitions.

A 'software service' is an on-line callable service that performs a function and returns a result; it can be invoked over the net and shared. Three major functions are needed by PDS: delivering data to PDS (standards, tools, and transformations), managing (including preserving) data within PDS, and distributing data (including transformations) from PDS.

Design Considerations:

Design should allow for phased improvements over time.

Design should provide clear boundaries separating archive, user, and data providers.

Discussion topics include options for data and system model, transformations, grammar, generic and specific services, tools, and data dictionary. Also to be discussed are impacts, tradeoffs, migration of existing holdings, and transition to the new system.

Data Architecture (Hughes):

After this meeting tech staff across PDS should understand the model better, they should be able to provide feedback on how to make the model better, and they should be able to identify key information that should be forwarded to management.

Goals of PDS4 include a stable and usable long-term archive, more efficient archive preparation for data providers, and services for consumers to find the data they need and provide the formats they require.

Design principles include a data model that is independent of language and implementation and definition of a few fundamental and stable structures, which can be extended to handle more complex formats. The *archive* formats should be independent of provider and consumer formats. The architecture should include a standard data dictionary model.

Deliverables include a PDS4 information model, a data dictionary model, a standards reference, and grammar options.

There are four basic data structures (Figure 1): a homogeneous N-dimensional array of scalars (array_base), a heterogeneous repeating record structure of scalars (table_base), an unencoded byte stream, and an encoded byte stream. Homogeneous data have no interleaving. Unencoded byte streams can be parsed simply; encoded byte streams cannot and may require computation to recover the data. Encoded byte streams will be tracked so they can be recalled and migrated to more 'fashionable' codes later (if necessary). A hierarchy is built on these basic structures (Figure 2).

The master model constrains data design and defines validation; it and the data dictionary are tightly coupled. The 'document writer' translates modeling information into target languages based on grammar. Updates to the master model are quickly reflected in the documents (standards reference, label schema, diagrams, queries, etc.) because everything is linked and automated.

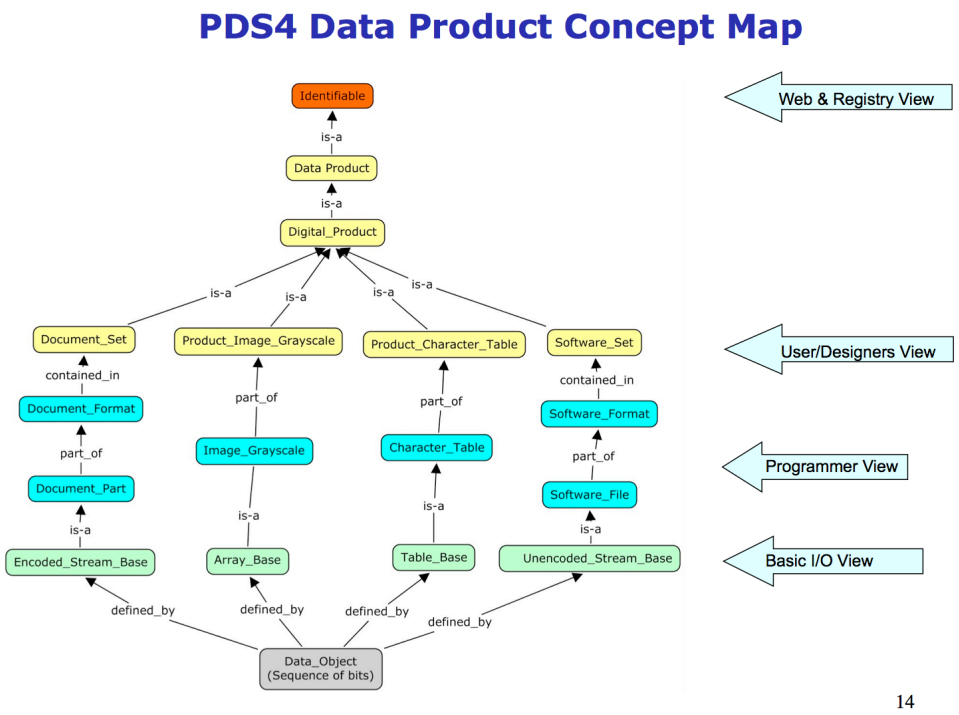

Figure 1- 'Sequence of bits' is the most primitive state in the data hierarchy. Once identified as a fundamental data structure, it can be read or written. With descriptive information, its bits take on meaning and can be interpreted. The 'data product' can then be registered as an 'identifiable.'

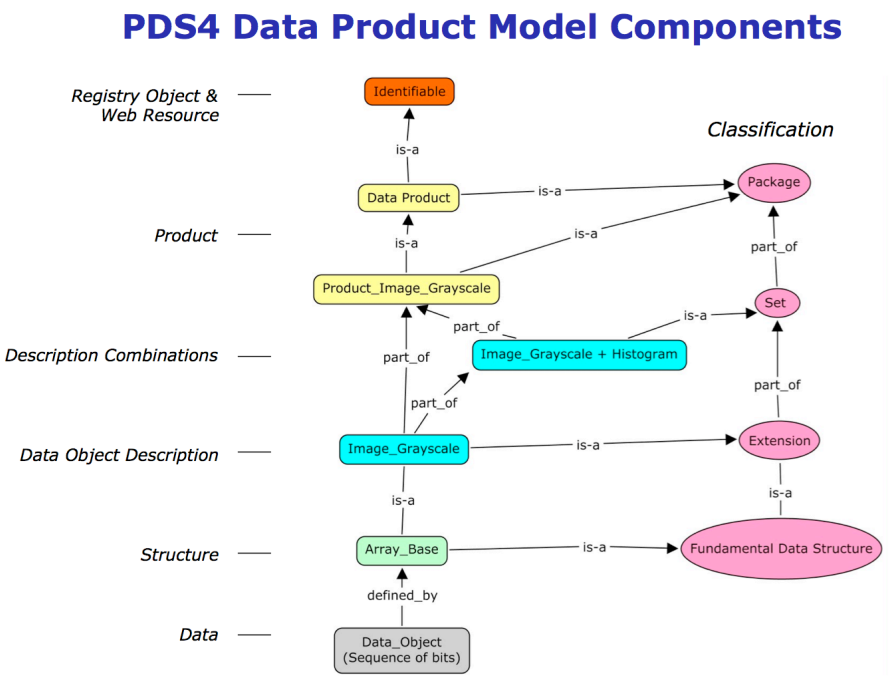

Figure 2 - The progression from 'sequence of bits' (Figure 1) to an identifiable two-dimensional image with associated histogram.

PDS 2010 System Design (Hardman):

Design principles include introducing common software that is extensible; isolating technology from function to minimize tight coupling between components and facilitate future changes; simplifying interfaces to facilitate adoption and use of software; and utilizing standard, open source, and COTS solutions where appropriate.

Design goals include improved ingestion efficiency, facilitation of tracking and improved integrity of the archive, facilitation of searches across nodes, improved delivery to users and deep archive, increased integration of software services across the system, and simplification.

Design constraints include the understanding that DNs govern local data and metadata holdings, that the size of PDS holdings limits the volume of data that can be transferred to perform any given function, and that the PDS 2010 funding profile favors flexible/phased development and deployment.

The Service Oriented Architecture (SOA) is well suited to a distributed system; it can capture some of the best features of other systems (see PDS white paper on registry). The design team has been working toward a lightweight (simple and flexible) SOA solution for PDS. Service-

based functionality will focus on search and retrieval of data. A tool-based approach is still appropriate for data preparation. Gordon suggested asking major data providers what changes would be desirable to improve ingestion. Gordon also suggested that a service be included that checks and updates secondary holdings (about 20 percent of RINGS holdings are primary at another DN).

The proposed design will provide a search capability (subsystem) that can be installed at both EN and DNs; Hardman expects that interfaces (GUI, API, etc.) could be designed and built by users and operators of other data systems. In addition, PDS3 data could be searchable through these interfaces, where performance would depend on the quality of the metadata.

Implementation (Hardman):

A registry provides services for sharing content and metadata, including capturing metadata. Some registries are associated with repositories. A federated registry is distributed. Components can include inventory (mission, instrument, target, data set, data product), dictionary, service, security, search, document (PDS documents, software, schema), and ingestion. The service registry could include services outside PDS. UDDI and ebXML are the two prevailing registry standards in common use. WellGEO RegRep is a COTS product. IVOA and ECHO have built their own.

Data Product Preparation: Tools would include label design, label generation, data transformation, and validation.

Dictionary Maintenance: Maintenance includes creating, reading, updating, and deleting entries. Metadata for registered object/element definitions are stored in a local data base. The service has the potential for stronger enforcement of keyword formation and use; but there are still elements of experience, knowledge, and will.

Data Product Ingestion: A lightweight service would be a 'crawler' that wakes up occasionally to see whether new or modified products have arrived. Metadata are extracted and registered with the inventory service. Detailed tracking can be included and checksum information maintained.

Huber asked whether Hardman could recommend a platform; ATM is expecting to upgrade its system in the near future. The consensus was that the design should be platform neutral; Bennett noted that future data nodes may not be able to adapt and so the system *must* be platform independent to accommodate expansion.

PDS4 Examples (Hughes):

Hughes showed 'first-cut' examples to illustrate what users should see in PDS4 and how the products came about. A generic label schema is translated into a specific label schema to be used with real products (Figure 3). Simpson asked about the order of 'words' in the names (for example, image_grayscale versus grayscale_image). The PDS4 labels include new keywords

which give information that has been assumed previously — for example, where on the display the first pixel should be placed (first_element = TOPLEFT). Bennett noted that the basic data type is assumed and that should be made explicit if the logic is to be consistent.

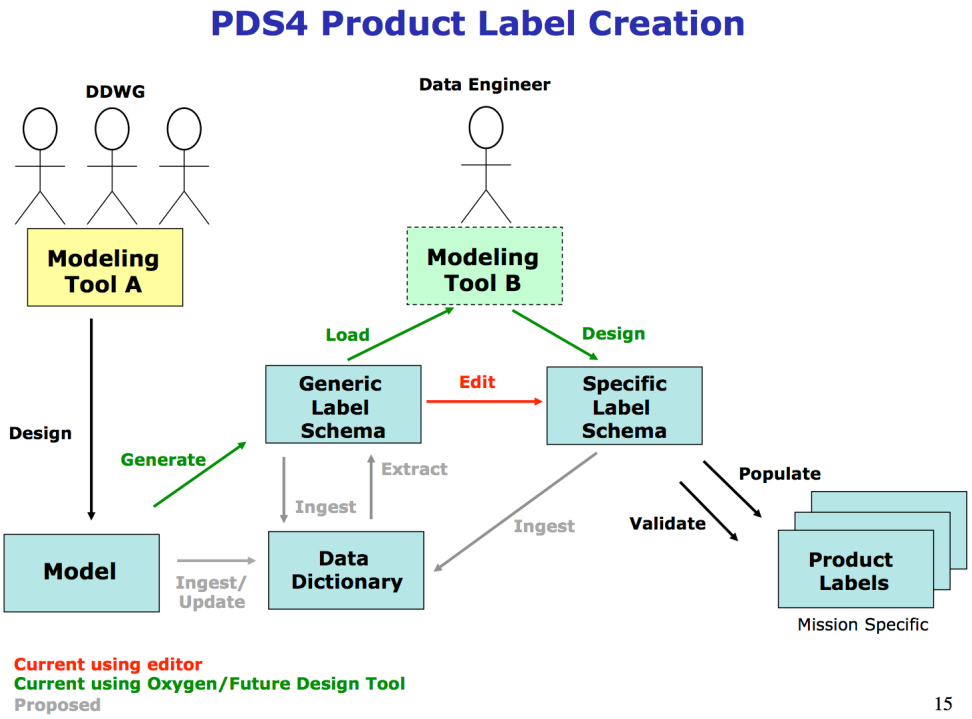

Figure 3. The information model constrains generic labels. When the generic label schema is applied to real data, a specific label schema becomes the template from which many product labels are created. The model, through the data dictionary, is then used to validate the real labels and guide their ingestion.

Products are 'identifiables' and all have an OBJECT = Identification_Section label component. The referencing section includes pointers to mission, instrument, target, etc. and can include values that are specific to the product, such as instrument modes. A question was raised about whether counts start at 0 or 1. Isbell asked whether labels are assumed to be attached, detached, or other; if checksums are to be included, detached is the only practical choice, but then how do you carry the checksum of the label? Some keywords have multiple values; this allows a single keyword to contain names of multiple axes or to contain a file name and offset; Semenov said he preferred explicit, single-valued keywords even though this makes the label longer. Some elements were originally modeled as classes and appear in the label as keywords; he also objected to this simplification.

Label sections include identification, description, circumstances of observation, data set, mission, instrument host, instrument, node, target, set definition, and the file itself.

Simple Table (Huber): The simple table is an ASCII file with numeric and/or character valued columns. Huber's example data were originally in the form of a single Excel file; this has been converted into a 10 column table. The use of quotes and semi-colons on keyword-value lines

depends on the grammar chosen; use of include files would also be governed by the grammar. There needs to be a way to include 'optional' keywords, but that hasn't been discussed; descriptive classes (groups of related keywords) may be possible, but probably not new structures (objects).

Complex Table (Guinness): SPECTRUM and CONTAINER are PDS3 objects that require more than the simple TABLE object. Guinness' example data come from the Phoenix Atomic Force Microscope (AFM), which maps topography at atomic scales. Output data are 512x512 grids of measurements; not all "z" values have the same meaning (some are height, some are slope). In PDS3 there is a simple header table and four data tables, using CONTAINER objects to group repeating columns.

Image Grayscale (Rye): The simple image starts from the array_base basic structure; an example is from Mars Pathfinder. There can be header, statistics, and special constants associated with the image (forming a set). There was discussion about which keywords have fixed values, what happens if someone changes them, and how the system reacts. Ganderson asked whether the objective was to construct a label that is good enough to be interpreted in 50 years without reference to the model or the dictionary? There was consensus that the model should not be necessary but the dictionary might be. Should the data dictionary (and the standards reference) be included in the archive? They should probably be available, and PDS should keep all versions of both. Martin asked whether labels are necessary in the archive; isn't the information in the registry? Ganderson asked whether default keywords could be added during the ingestion process rather than requiring all labels to include all keywords. Simpson asked whether a gray scale image could be associated with red, green, and blue images to create a de facto color image; yes, but perhaps the DN would object. Browse images could also be packaged with original images.

Software/Document (Raugh): Both usually have several discrete parts, which constitute a whole; there can also be application specific formats (software scripts, source, and binaries; text, graphics, Microsoft, and PDF versions for documents). Software should be accompanied by programmer's and user's manuals; there was a question where the documents should be stored (with the software or with other documents). Software/documents are 'products' on a par with data. Semenov asked how the NAIF Toolkit would fit; NAIF likes to distribute a tar file with an install script. Bennett asked whether browse images would be considered documents; that has been practice in PDS3, but there has been no discussion of browse files for PDS4. Joy asked whether test input/output files should be included with software; Raugh has recommended regression_* keywords, but the model has not been developed to such a point. Joy asked whether PDS should continue to pursue copyright permissions to include 'instrument' papers; this is a different question, but the 'doi' is the permanent identifier which should be included in citations.

Format Translation (Raugh): Raugh has looked at FITS and VICAR files to see what would be involved in translation to PDS4. The FITS to PDS4 image translation took about 4 hours to code and test. She then spent about an hour adding a FITS header to a PDS4 ARRAY object. Coding and testing the VICAR conversion prototype took about 8 hours. Raugh has not seen any VICAR files with floating point pixel values, which simplifies matters; VAX numbers have

bytes swapped, and VAX floats have a different exponent bias, which can be converted with a PDS SCALING_FACTOR.

Calibration (Garcia):

How do users know when new calibration data or software have been released? Do DNs actually know when new calibration data or software have been released? Slavney believes DNs usually know, but there is no formal process for notification. The 'registry' in PDS4 could be used to track which products need to be calibrated, where to find the necessary files, and who needs (or wants) to be notified. What kinds of calibration are needed, expected, desired, ... and what degrees of calibration are available (accuracy, precision, etc.)? Rye is working on types of calibration, Hardman is considering how to track. Joyner noted that some data are accompanied by 'quality' indicators, which may change.

PDS has historically instructed users to check the latest volume for calibration and errata. But this isn't so simple with large, accumulating electronic volumes. Are AAREADME.TXT and ERRATA.TXT always up to date? Simpson said that it might be time to recommend structure in ERRATA.TXT; large ERRATA.TXT files can be very difficult to read because the content is organized chronologically at best. EN doesn't like frequent updates of DATASET.CAT with new coverage, calibration, or quality information because the ingestion process is so cumbersome. Joyner recommended a revision_history.txt file, which would be separate. Can the subscription service help? Can we have pop-up windows that advise of new calibrations or offer subscription notification of future revisions? Gordon recommended general notices in *INFO.TXT files pointing to calibration files and processing history.

Data Dictionary (Hughes):

The current model is simplistic, limited by ODL grammar, and poorly maintained. There is no constraint language ('recommended' and 'optional' appear frequently; neither is helpful). There are three main recommendations for PDS4 (see below).

Concepts: The DD contains a set of defined data elements; each has a set of attributes. The DD expresses the information model's structural and descriptive metadata. The DD has a registration authority. Data elements can be grouped and classified, allowing hierarchies.

Recommendation 1: The ISO/IEC 11179 metadata registry specification is recommended. It is the standard model for data dictionaries and provides a standard description of data and a common understanding of data across organizational boundaries. It addresses semantics, representation of data, and registration of descriptions. There are six volumes in the specification; volume 3 addresses schema and volume 6 is on best practice in forming descriptions. PDS should adopt volume 3, but the others have more to do with governance and PDS may want to develop its own equivalents.

The schema includes sections for data element, class, and values. The data element is defined by name, submitter, steward, definition, namespace, source of definition, change log, version, concept (a higher level definition), alternate names or aliases, definition in other natural languages, classification (a grouping for some purpose), unit of measure, and effective dates. Class includes attributes. Value includes value, submitter, steward, definition (mostly for character strings), cardinality (how many values are allowed), source of definition, change log, version, concept, character set, representation (what types of values are allowed), minimum and maximum values, minimum and maximum length, alternate encodings (not known whether arithmetic expressions are allowed), and effective dates. The details of how each is used (including whether used) will be determined by PDS. Conditional keywords (keywords which depend on other keywords) may be possible in the next version of the ISO/IEC standard. Schematron is a tool which will check XML files that values for different keywords are consistent.

Recommendation 2: Classify data elements in the dictionary into namespaces; use namespace to minimize changes to the dictionary at the global level and to coordinate among levels. There are expected to be three types of namespace: PDS common, discipline/agency/organization, and mission/LDD. All PDS common data elements will be in the PDS DD. Any others used in the PDS archive will at least be registered with PDS, and ideally will be incorporated into the archive where it is used.

Recommendation 3: Set up a data dictionary service. It would export the DD in a variety of formats, manage and audit DD changes, and enable on-line validation against the DD.

Secondary Recommendations: Adopt the concept of 'steward' as an organization that manages an administered item." The set of classes, relationships, and attributes grouped by a namespace will have the same steward and registration authority. Design a generalized namespace that can be implemented in multiple grammars. Allow attribute and class references to the current dictionary and any dictionary higher in a hierarchy. Do not allow lateral references between local data dictionaries. All dictionaries will be accessible.

The plan for the next few months is to build a DD using the standard data elements around a namespace standard, set up an on-line DD, and develop tools for managing the DD.

PDS Grammar in ODL, PVL, and XML (Ramirez):

Grammar provides a syntax for capturing PDS labels, but there are implications beyond syntax. Selection of the grammar should be standards-based and widely adopted. The choice should provide files which are both human and machine readable, expressing all aspects of PDS data object descriptions. Ability to use the grammar in pipeline processing is important, and support by libraries in many languages yielding consistent results is highly desirable.

ODL is governed by PDS. There are attribute, object, and group statements, all boiling down to keyword = value. PVL is like ODL except it uses semi-colons and is governed by CCSDS. XML looks like HTML (but is not). It uses tags with nesting; users define tags. There are many

standards, which address special needs, supporting XML. W3C (World Wide Web Consortium) governs XML.

There was extended discussion on which grammar provides files that are easier to read, which can be transformed to other formats most easily, what third-party software is available, who maintains applications and tools, and what the future holds for each choice. There are a lot of third-party tools available, which help with label design and prototyping. XML can be generated by software in batch mode.

Recommendation 1: Upgrade or replace PDS grammar in PDS4. Recommendation 2: Perform a PDS tech group survey to determine whether there is a 'near consensus' on how to proceed. EN believes XML will improve PDS software development long term (cost, functionality, and consistency). The major question is impact and usability. Ultimately, EN believes ODL, PVL, or XML will work.

Joy said the people building SPASE with XML are not terribly thrilled with how it is being used by this project. But they are starting from poorly defined PVL. SPASE has no data; it is the standards organization. NVO is tasked to use the SPASE system; the four VXOs are the only groups putting data into the system. Joy writes ODL and has a converter, which translates it to XML.

Gordon expressed concern that data providers and data users will not be conversant enough with XML to jump into this quickly. Ramirez suggested googling on "xml open source editor" to get information on available tools; Bennett thinks the transition to XML may not be as difficult as it seems. Rye suggested that major data providers may already be eager to convert to XML.

PDS4 Standards Reference (Rye):

The current standards are plagued with inconsistencies; maintenance is time consuming; the document is bulky and difficult to use; and there are requirements, recommendations, and suggestions. In addition to the SR, there are "guides" and policy statements approved by MC.

The intent in PDS4 is to start with the PDS4 information model, to which the DD and SR will be subsidiary. There would also be a guide for novice users, a data format standard, and XML standards. The inconsistency problem should be solved by having the information model at the core, the target audience will be narrowed to experienced users and programmers (except the novice handbook), and there will only be 'requirements.'

The SR will have an introduction, a summary of archive organization and nomenclature, sections on archive components (browsers, calibration, catalog files, data, dictionaries, gazetteers, geometry, indices, labels, and software), and appendices (grammar, data format standard, and user classes). But there was sentiment for including the basic data types rather than user classes.

A possibility is to change the SR name to *Archive Standards Reference*, emphasizing what goes into the archive; but there was concern that users also need a reference document to understand what has been downloaded from PDS.

Gordon recommended that there be better control over what goes into catalog, document, and calibration directories of the archive. There should be more consistency across data sets in terms of what is found and where. There was also consideration of whether there should be more emphasis on pointing new users toward AAREADME.TXT and/or providing a 'site map' of the archive.

Simpson was concerned about having multiple versions of SR, each of which needs to be maintained. Rye is open to having a single document with multiple levels of explanation or to modular documents; but she suspects that the document versions we have been using (full reference, novice, proposer, etc.) may still be requested.

Transformations (Gordon):

These are the underlying PDS4 philosophies: a model-based, integrated system; a model that is rigorously defined, explicit, and internally consistent; and base formats that have been optimized for archiving.

Huber noted that base formats optimized for archiving are not usually what instrument teams provide; data which have been converted to base formats may receive very little inspection prior to ingestion.

Considerations: Expect to support PDS3 and PDS4 in parallel for the next ~10 years. Expecting that PDS4 rolls out in 2010, what mission is the best mission to be the beta test? Since PDS4 will likely be released in stages, what are the functions to implement first? MAVEN, GRAIL, and LADEE are possibilities, each with advantages; timing is tricky. Perhaps a legacy mission (CASSINI?) will be the mission which provides most of the testing.

The initial release will have minimal capabilities and tools; but it must be sufficiently well developed to support archive planning. Is the first expansion available before the first mission submits? What will be in the planned expansions?

Separately, PDS must plan for migration of old data (legacy formats, labels, and metadata); when does this start and how long does it take? What are the high-priority data sets? There will be decisions needed on byte order; should we retain original data formats and convert on the fly, or convert to archive base formats and abandon originals? Do we want to stay with 7-bit ASCII or change to uni-code (this has international implications)? How do we deal with the migration to 64-bit processors? Do we want to specify axis order, delimiters on text files, data type in binary data, mapping of black-white to pixel value, ...?

There are format conversion questions (FITS, ISIS2, ISIS3, CRISM, VICAR, etc.). Raugh has already written a FITS to PDS4 converter. There is consensus that ISIS3 should not be ingested

into PDS; ISIS2 is TBD. VICAR could be approached three ways: providers make PDS compliant files, allow exceptions to the model, or convert to a base format as part of ingestion. Although conversion is possible, PDS should be aware that VICAR applications require certain metadata which may not be available in FITS images (for example).

Transition, Migration, and Impact (Crichton):

Migration is the process of moving archived data from PDS3 to PDS4 (this does not include changes needed to implement PDS4). Is there a distinction between migrating data and migrating metadata?

Transition is the process of moving from a functional, prime PDS3 system through the PDS 2010 project to decommissioning of the PDS3 system (or not).

Impact starts with understanding what the major design decisions and options are. What are the impacts to PDS, providers, and users of these decisions? Components of the impact analysis include resources, schedule, training, usability, and efficiency. Quantitative estimates need to include both cost and the number/types of data.

Design decisions include selection of the data/information model, selection of the grammar, and selection of the data dictionary. Impacts may be most severe on data providers. Although some may appreciate the shift to a more logical information model and use of XML, they may find the constraints on file formats to be objectionable; this has been a problem in the past, and it's not likely to vanish with appearance of PDS4. Raugh believes the learning curve for new users will be easier in PDS4 than it has been in PDS3. Crichton thinks that the new data production tools will leave humans with fewer data handling responsibilities, and the stress of creating archives will be lower.

Migration and transition decisions include how to select the data to be migrated. 'No migration' is probably not an option; a phased and incremental migration based on user demand is probable. Gordon said this forces a decision on disposition of superseded data; but Simpson broadened the question to deciding what versioning means. Crichton needs to come up with a rigorous procedure for defining ARCHIVE_STATUS levels and assigning the values. There is a question of whether NSSDC is properly configured to serve as the PDS deep archive if 10's of TB are beyond its capacity to ingest.

The transition proposal is to move to PDS4 with a succession of increasingly functional builds, continue support of PDS3 ingestion and distribution, and migrate the PDS3 catalog to PDS4 registry services. The principal impact will be migrating the central catalog and on building PDS3/PDS4 compatible tools. There should be relatively less impact on data providers, DNs, and users.

Quantitative analysis includes cost (FTE support for upgrading software, development of PDS4 services, change to XML, and granularity at which some of the changes are made) and data (what percentage of PDS3 products would be accepted in PDS4, and the anomalous formats such

as QUBE, VICAR, FITS, ...). MIWG will be enlisted to spread the word that currently accepted formats may be in jeopardy; DNs will have to advise new missions and data providers that there may be restrictions.

Wrap Up (Crichton):

Moving forward to the Management Council meeting in August ...

Working groups need to continue, drafting specifications on initial capabilities and developing plans to engage data providers and users. Identify the first missions. Develop a schedule for tool development and a schedule for other phased development.

Add warnings to web pages advising of changes ahead. Solicit input from data providers and users.

What are the options for distributed development? Are there things that can be done at DNs, using their special skills, which make the federation more meaningful and spread the load? The Senior Review objected to perceived duplication of effort in DN software development.

What to prepare and present? Should the Tech Group meet a day early to prepare for the MC presentations? Identify benefits of new design, compare to other science disciplines. Node examples and presentations might be useful. What are the design trade-offs and recommendations. Impact matrix. Build/release plan. Next steps. Prepare a list of known limitations and problems that can reasonably be expected with a fully deployed PDS4.

Simpson's List of Loose Ends and Nagging Issues (Editorial)

1. Need to be developing registry requirements *now*. Successful operation of registries is critical to most PDS improvements. Gaps in registry coverage will make desirable PDS4 functions impossible. Registries need to be overly capable rather than less to optimize success of future enhancements.

2. Where are the implementation boundaries if PDS3 is to be 'converted' to PDS4? What are the options for partial (as opposed to phased) implementation?

3. Versioning (part of registry). What represents a different version, how do you keep track of it, how do users understand the differences among versions?

4. Extensibility/scalability has been a goal; what do we know about the subject?

Original: 2009-06-11
Changed Duxbury to Rye; minor edits and corrections: 2009-06-18
Incorporated suggestions from Crichton: 2009-06-24