STREAMLINING PDS3 TO PDS4 CONVERSION: LEVERAGING THE PLANETARY DATA READER FOR IMPROVED EFFICIENCY: A CASE STUDY WITH VIKING DATA. M. A. St. Clair[1], S. V. Brown[2], C. C. Million[2], S. A. Curtis[2], [1]Million Concepts LLC (mstclair@millionconcepts.com) [2]Million Concepts LLC

Introduction: The Planetary Data System is in the process of converting data from the Version 3 standards (PDS3) to the Version 4 standards (PDS4). We recently migrated the PDS Imaging Node (IMG)'s Viking Orbiter (VO) holdings from PDS3 to PDS4. The products are currently in review at IMG and will likely be publicly available in Q3 or Q4 2023. Here we describe some details of this migration process, including file conversion, label creation and validation, documentation, and software development.

The Legacy PDS3 Problem: PDS4 has a strict information model [1] that permits formal verification of product validity. This helps guarantee that products are, and will remain, analysis-ready. PDS3 is a looser standard with no such validation mechanism, and, furthermore, data providers did not consistently follow its standards. Many PDS3 products have inconsistent metadata and/or bespoke file formats. It is common to require specialized (and usually abandoned) software simply to *open* a PDS3 product, and this software is often not compatible with modern operating systems.

Our general solution. We have developed an open-source tool called *pdr* (short for Planetary Data Reader). It can currently read most PDS data products and will eventually be able to read almost all. It is intended in part to make PDS4 migration easier. So far, we have used software backed by *pdr* to migrate products from the Chandrayaan-1 (CH-1), Clementine, and Viking missions to PDS4.

A Primer on the VO Imaging Data: Viking orbiters 1 and 2 (VO-1 and VO-2) launched in 1975 and entered Mars orbit in 1976. They remained in operation until 1980 (VO-1) and 1978 (VO-2). IMG holds products derived from data taken by the VO Visual Imaging Subsystems. They include 'unprocessed' Experiment Data Records (EDRs) as well as map-projected Digital Image Models (DIMs) and Digital Terrain Models (DTMs).

History. During the mission, the EDR data were stored on tape and archived with JPL or USGS. JPL's 1983-1984 Planetary Image Conversion Task (PICT) produced VICAR-readable versions of these data. In 1990, personnel from Washington University, USGS, and JPL used the PICT products (and some additional recovered data) to produce a set of CD-ROMs containing the EDRs. USGS and JPL personnel map-projected these images and released them as four CD-ROM sets in 1991 (low-res single-band maps), 1992 (multispectrals and DTMs), and 1999 (high-res single-band maps). IMG's current web holdings are essentially identical to the contents of these CD-ROMs and retain their volume organization.

PDS3 EDR format. The data preparers losslessly compressed the EDRs with an algorithm designed at Flagstaff. Each EDR is a monolithically-compressed file containing a PDS3 label, a histogram, two binary tables, and a raster image. The decompression software is not compatible with modern operating systems.

PDS3 map-projected format. The DIMs and DTMs are PDS IMG files with attached PDS3 labels and histograms. The single-band tiles are derived from all-time data and thus optimized for use as basemaps; the multispectrals are derived from data collected during single orbits (one grid square may have more than one tile) and thus optimized for viewing change over time. Each multispectral tile is 1-4 products (one product per band per tile).

Conversion Process: Simply converting these data sets to PDS4 standard posed a number of challenges. The EDRs were not immediately readable due to their custom compression format, the map projection metadata were not PDS4-compliant, the embedded table data in the EDRs was difficult to interpret, we needed to produce valid XML labels for ~100000 products in heterogeneous formats, and we wanted to improve usability as well as ensure formal correctness.

Fetching and preprocessing. We began by scraping IMG's VO web holdings and checking the results against manifests provided by IMG and indices in the data sets to ensure completeness. The map-projected tiles were in a fairly simple raster format and required no preprocessing: *pdr* could read them 'out of the box'. The custom compression format of the EDRs, however, presented problems. The decompression binaries were only compatible with classic MacOS, DOS, and VAX/VMS; similarly, the Fortran/C sources used legacy language versions. Fortunately, we were able to execute the DOS binary in DOSBox [2], a DOS emulator. DOSBox was designed by historical video game enthusiasts and does not support DOS loop structures, so it was impossible to write batch files to decompress the full data set. We automated decompression by injecting simpler batch files into individual instances of DOSBox executed in parallel. On decompression, we further discovered that the embedded EDR tables were labeled in unusual ways that required us to implement special handling in *pdr*.

Metadata design. Some PDS3 parameters have obvious translations into PDS4 attributes or classes. Many do not. We attempted to design labels that were expressive, idiomatic, and erred on the side of

including more rather than less metadata. *pdr's* fast metadata parsing features were helpful; they permitted us to build tables of all parameter values across the data corpus in order to understand the "types" implied by each parameter. Map projection definitions posed a special challenge, as we describe in more detail below.

Label templating. To help enable programmatic generation of labels, we reused a markup syntax we developed for our CH-1 M3 [3] and Clementine [4] projects. It consists of simple hooks for our processing code inserted into stub XML files. It is *not* a template language (e.g. XSLT). Embedding executable code in a template is a powerful technique, but we consider it unnecessarily complex for 'offline' processes—more appropriate for web applications than archival pipelines.

Data design. While embedding binary tables and a raster image in the same file is PDS4-compliant, we consider it poorly usable. For this reason, we 'unpacked' each EDR file into one raster and two table files. We discarded all embedded histograms; they were designed for CRT display optimization and irrelevant to modern users. We reformatted the 'split' multispectral tiles into multiband files (we believe the data producers intended them to be analyzed this way, and split them out of sensitivity to early-90s memory constraints). We wrote all raster files in FITS format.

Browse products. We generated PNG browse images for all observational products. We created two PNGs for each EDR and map-projected observational product: a simple contrast-stretched version and a version that masks missing data in cyan (the default mask color for *pdr*'s `dump_browse` method). We also created a third 'filtered' browse image for each EDR by implementing an algorithm described in the source documentation. We believe these filtered images best express what the data providers considered 'good' representations of the EDRs.

Bundle design. We did not retain the 68-volume structure of the PDS3 data corpus, which was ideal for CD-ROM but not the modern Internet. We organized all products into a single PDS4 bundle with document, browse, and data collections. The data collection has distinct subtrees for the single-band DIM, DTM, EDR, and multispectral products. The map-projected products then have subtrees for tilesets by resolution and latitude bins within those tilesets; the EDRs have subdirectories by orbit number. We also modified the EDR filename format, which was designed to fit in the 8 + 3 character limits of the era and so contained limited identifying information.

Georeferencing. PDS4's map projection standards are very different (and considerably stricter) than PDS3's. PDS3 volumes contain one DSMAP.CAT file per projection type; observational product labels contain parameters for these projections. By contrast, PDS4 labels must completely describe the projection. PDS4 also requires that labels explicitly specify the latitude and longitude of the upper-left pixel, which PDS3 labels usually only give implicitly via bounding coordinates, scale, and offset from projection center. With the help of the *pyproj* library, we were able to derive compliant map projection metadata from the labels of the sinusoidal products (most of the tiles) as well as the "special" multispectral orthographic tiles. The polar tiles, however, did not contain adequate metadata to specify compliant map projections.

Actually converting the products. *pdr* is under active development, so to ensure traceability, we created a "frozen" fork of *pdr*. Then, building on techniques we developed for the M3 and Clementine data, we wrote a library that leverages *pdr's* easy access to product data and metadata to populate our XML templates and execute file conversion. The library is centered on the *PDSVersionConverter* class. Instances of this class wrap one or more *pdr.Data* objects and associate them with sets of conversion rules, which can be defined in either class or module scope (to share them between subclasses). These rules provide procedures for converting files and populating labels. Most label conversion rules work by defining relationships between XML markup tags and PDS3 parameters, although some markup tags must be populated more dynamically (e.g., tags related to file size). *PDSVersionConverter* also defines a *.from_converter* method that can be used to define subclasses for producing secondary (e.g. browse) products. This provided us with a modular, highly-performant system that permitted us to quickly address bugs, insert special cases, and add functionality to the pipeline [5]. In our experience, every large legacy dataset has dozens of unusual features that only appear in a small number of products and cannot be detected until actually processing everything and running Validate Tool, so it is very important to keep your code flexible.

Acknowledgments: Support for Viking VIS data conversion provided by USGS 140G0322P0253. Support for development of PDR provided by NASA 80NSSC21K0885.

References: [1] PDS4 Information Model Specification Team, (2022) *PDS4 Information Model*, *v.1.19.0.0*

[2] DOSBox: https://www.dosbox.com/

[3] M3 conversion software github repo: https://github.com/MillionConcepts/m3-conversion

[4] Clementine conversion software github repo: https://github.com/MillionConcepts/clementine-conversion

[5] Viking orbiter conversion software github repo: https://github.com/MillionConcepts/viking-orbiter-conversion