PDS4 in GDAL

PSIDA, April 2018
Trent Hare and Lisa Gaddis

thare@usgs.gov
What Domain?

• PDS4 supports dozens of domains
  • Atmospheres (GCMs)
  • Planetary Interiors
  • Astronomy
  • Astrobiology
  • etc., etc., etc.

GDAL is Geospatial – “tied to a planetary surface”
What is GDAL

**Geospatial Data Abstraction Library**

- GDAL is a “translator library for raster geospatial data formats”
- Open source (and community driven)
- Used in many “geo” applications: QGIS, UDIG, SAGA, ARCMAP, GMT, MapServer (WMS), Google Earth, AMES Stereo-pipeline, SOCET GXP, …
- Handles many image formats for read and slightly less for writing: PDS3, ISIS2/3, VICAR, **FITS** (via CFITSIO), ENVI, GeoTiff, Jpeg2000, PNG, cloud-based, NetCDF, … - over **150 formats**

April 2018

More info
What is GDAL

Geospatial Data Abstraction Library

• Started in 1998 by Frank Warmerdam
• A project of OSGeo since 2008
• MIT/X Open Source license (permissive)
• >1M lines of code for library and utilities
• > 150K lines of tests in Python

From: Even Rouault (link)
Adding PDS4 -- Contract

• Open bid awarded to Hobu Inc. https://hobu.co/ GDAL and LIDAR specialist.
• 3 months allocated. Delivered in 2 months (1 month for testing).

• Overview:
  • Incorporate PDS4 read and write capabilities into existing GDAL C/C++.
  • PDS4 keywords to be accessed programmatically (C++ or Python).
  • Allow for detached PDS4 label that is pointed into a “raw” GeoTiff.
  • All code available and supported under the standard GDAL release.
Beta released in trunk (Sep. 12th, 2017)

• Initial testing finalized on Linux RedHat
• Astro has built a Anaconda environment (for low-level Python API support)
• 18 Unit tests provided with delivery: https://trac.osgeo.org/gdal/browser/trunk/autotest/gdrivers/pds4.py
• Initial help page:
  • http://www.gdal.org/frmt_pds4.html
Reign in expectations

• A full "GDAL" solution relies on a **good PDS4 template**.
  • *Similar how FGDC metadata works in the real world. Metadata is never fully automatable.*

• Only the **image's physical parameters** (lines, samples, bit type) and **map projection** are **automated**.

• But GDAL supports template **variables**, so scripting can support a full EDR label using Python, PERL, etc.
Simple scripting examples on GitHub

• Written In Python, but shows how Bash, PERL, etc. could be used
https://github.com/USGS-Astrogeology/GDAL_scripts/tree/master/PDS4gdal

```
observeID = getkey(from_=inputlbl, keyword='InstrumentId', grp='Archive')

theLine = '-co VAR_OBSERVING_SYSTEM_NAME={}'.format(observeID)

fileConfig.write(theLine)
```
Pros

• Supports writing and **full XML schema validation** (via Xerces).
• Convert from most PDS3, ISIS2/3, VICAR, FITS, GeoTiff, Jpeg2000
• With XML template, GDAL can support fully **compliant** PDS4 label
• Templates can be used from an “http” address
  • e.g. mission website, PDS, github link
• Template **variables** allow for simple scripting
• Full XML access in Python using lxml
Pros (cont.)

• Supports writing a PDS4 label pointing into a GeoTiff (interoperability)
• Array_xD_Type defaults to Array_3D_Image (but can be user defined)
• Builds on all OSs (even on cell phones ;-) )
• PDS4 Driver will eventually show up in GIS apps like QGIS, ArcGIS Pro, SAGA, GMT, GRASS…

April 2018
Cons

- Targeted specifically to support map projected data sets
- No PDS4 table support (yet).
- No support for writing multiple image arrays in one file (called sub-datasets).
- XML Templates requires more design up-front and more input from data provider
  - Leverages existing PDS4 label helpers (PLAID, OLAF, …).
Conversion examples (viewed in PDS4 Viewer)

Dawn, Galileo, Hayabusa (Mercator, Sinusoidal)
Conversion examples (PDS4 Viewer, ArcMap)

Dawn (Equirectangular)
Conversion examples (PDS4_Viewer, ArcMap)

CASSIS
PDS4 XML examples

• Very minimal PDS4 XML template (ships with GDAL binaries)
  • https://www.dropbox.com/s/px97xm57n2q6a83/pds4_template.xml?dl=0

• Dawn output example w/ GDAL-added image section and map projection
  • https://www.dropbox.com/s/b69cejqvum181dl/dawnEqui_pds4.xml?dl=0

• LOLA output example with GDAL added image params and map projection
  • https://www.dropbox.com/s/pn2dwq5mdz702cx/ladem_4_pds4.xml?dl=0
Current testing environments (GDAL 2.3.0 beta1)

1. Build from GDAL trunk: [http://www.gdal.org/daily](http://www.gdal.org/daily)

2. Anaconda (Python)
   • `$ conda install -c usgs-astrogeology gdal-pds4`
   • or install a whole environment with the required packages:
     • `$ conda env install usgs-astrogeology/astrogdal`
Next contract

• Add PDS4 **table** support.
• Write multiple image arrays in one file (called sub-datasets).
• Add units within “image” label section

• **Vector support**
  1. If PDS4 table has Latitude / Longitude field (acts as point)
  2. If PDS4 table has Well-Known-Text (WKT) field then supports points, lines, polygons
- Add PDS4 table support.
- Write multiple image arrays in one file (called sub-datasets).
- Add units within "image" label section.
- Vector support:
  1. If PDS4 table has Latitude / Longitude field (acts as point).
  2. If PDS4 table has Well-Known-Text (WKT) field then supports points, lines, polygons.

Proposed PDS4 ASCII Table example:
Wait - what...

• PDS4 supports Well-Known-Text (WKT)?
  1. Not yet approved
  2. Need feedback

Why WKT
• ISO standard
• Simple but allows for multiple shapes and holes
• Broad use across applications
Please email if you have specific requirements for next contract.

thare@usgs.gov, Trent Hare