Intro & Recent Advances
Remote Data Access via OPeNDAP Web Services

For the ESIP Summer-2016 OPeNDAP Workshop
Wednesday, July 20th, 2016, 13:00-17:00

Excerpted from a 2015 presentation to the
CEOS Working Group on Information Systems & Services (WGISS)

Dave Fulker, President of OPeNDAP, Inc.
subcontractor to Raytheon for NASA/ESDIS

Presentation supported by NASA/GSFC under
Raytheon Co. contract number NNG15HZ39C
Part I

Introduction to OPeNDAP* Web Services

*OPeNDAP is an organization and an acronym:

“Open-source Project for a Network Data Access Protocol”
OPeNDAP Concepts
originally from Distributed Ocean Data System (DODS) circa 1994

- URL $\approx$ dataset*  |  URL with constraint $\approx$ subset
- Retrieve
- Retrieval protocol built in to multiple libraries
- flexible data typing
- many, diverse clients

* dataset $\approx$ granule

- dataset descriptions (metadata)
- dataset content (typed/structured)
- arrays (~coverages)
- tables (~features)
URL ≈ Granule*
per OpenDAP’s Data Access Protocol (DAP)

http://laboratory.edu/device/experiment/granule.dmr

<table>
<thead>
<tr>
<th>Domain name often is an organization’s web server.</th>
<th>Servers often have hierarchical collections.</th>
<th>Each URL references a distinct DAP “dataset.”</th>
<th>Suffixes specify return types.</th>
</tr>
</thead>
</table>

Depending on suffix, DAP returns metadata or content, with options for human- or machine-readable forms (XML, NetCDF4,…). Suffix “dmr” → metadata only.

*dataset ≈ granule
OPeNDAP
Datatype Philosophy

- Internal data model has few data types
  - For simplicity…
- Types are domain-neutral but flexible
  - Structures & attributes ➔ rich syntax & semantics
- These types support many domain-specific needs
  - A recent crawl* (23,000 domains in .gov, .edu, .org) found >1400 collections with DAP servers

* By the National Snow & Ice Data Center (for NSF/EarthCube)
OPeNDAP services

Function as Middleware

- Data ingest via encoding-specific adapters
  - Handlers for a growing set of source-data types
- Multiple response encodings
  - Native DAP—useful in Python, Java, C++, Fortran…
  - netCDF (also GeoTIFF where possible)
  - XML (⇒ HTTP via style sheets)
  - Recently added: WMS, W10n (JSON), WCS (beta)
Architectural Overview of Hyrax

a widely-used DAP server

- Data-User App (python, java, native-DAP libraries)
- Data-User App (other standard, OGC-compliant)
- Data User, Browser-Only (XML or JSON)
- Core DAP Services (Hyrax Front-End)
- DAP-Extending Services

Apache Server Framework

Hyrax Back-End Server with Encoding-Specific Handlers

- HDF files
- netCDF files
- text files
- SQL database
- extensible...
URL + Query ➔ Subset & (future) results from other server functions

http://…/granule.nc4?dap4.ce=constraints&dap4.func=functions

<table>
<thead>
<tr>
<th>Dataset identifier as above, except return-type is NetCDF4 (= HDF)</th>
<th>DAP “constraint expressions” yield sub-arrays &amp; other proper subsets</th>
<th>DAP4 “function expressions” enable extensions</th>
</tr>
</thead>
</table>

Constraints specify subsets by variable names, by array indices & (for tables) by content. Likely extensions include statistics, UGRID subsetting, feature extraction…

The query form &dap4.func=… enables DAP extensions ⇒ new server functions
DAP-based
Subset Selection (from arrays | tables)

- Select variables by name
  - For tabular data, this means selecting columns
- Select rows of a table via column-specific value constraints
  - Allows both domain-based & range-based subsetting
- Select sub-arrays by constraining their indices
(array-style)

Index-Constrained Subsetting

Source Array ➔ Sub-Array (response)
Index-Based Subsetting

Excellent if desired subset is a bounding box parallel to source array (indices ↔ coordinates)

Less useful when
- Subset selection not based on domain coordinates
- Source is not organized as coordinate-mapped arrays
- Desired subset is polygonal or is skewed (relative to source-array orientation)
Part II

Recent Enhancements of OPeNDAP Web Services

*With Demonstrations*
This part of the presentation is drawn primarily from a project report on:
NASA Data Interoperability

An EOSDIS Presentation & Demo
Originally given March 27, 2015

Original Presenters: James Gallagher & Nathan Potter (OpenDAP)
main NASA motivations for

**OPeN DAP Enhancements**

- Easier software builds & better documentation
- Authentication of data users
- More response encodings
  - Open Geospatial Consortium (OGC) Web Services (WMS, WCS…)
  - JavaScript Object Notation (JSON) for Webification (w10n)
- Requesting DAP ops on many granules at once
  - Response = concatenated CSV (arrays ➔ tables) or
  - Response = zipped files
progress on simplifying OPeNDAP Server Installation

Context

- Hyrax-install complexity was once a barrier to use

Key Accomplishments

- Adding modules does **not** increase the package count
- Source build: now just 3 distinct packages
  - Previously 18 packages
- Binary install: now just 2 RPMs + 1 WAR
  - Previously 15 RPMs + 1 WAR
progress enhancing OPeNDAP’s Website & Documentation

Key Accomplishments

- Various Website repairs
- 760 fixed links (from automated before/after crawls)
- Five documents added
  - Client configuration for authorization
  - Server configuration for authorization
  - Source-code build how-to
  - Summary of Winter-2015 ESI-P-panel on Web-services performance
  - Server configuration for WMS provision
progress on

User Authentication
(via EarthData Login at NASA EOSDIS)

Context/Things to Notice

- Fine-grained access control for individual directories
- Demo is Web-only, but cURL tests work as well
  - cURL—like most client applications—is built around libcurl, thus serving as a lowest common denominator
  - EarthData credentials are simply stored in a user’s .netrc file

Live Demo…
prior context for enhancing

**Multi-Granule Aggregation**

- Many servers have allowed DAP providers to form (virtual) aggregations of (similar) granules
- But until now, end users could not choose
  - Granules to be aggregated
  - **Forms** of aggregation
- Furthermore, array- & table-style subsetting could not be **mixed** (with or without aggregation)
Multi-Granule Aggregation

**Context/Things to Notice**

- Request data from 1,000s of files with one operation
  
  *N.B.* Necessitates use of HTTP POST (to avoid huge URLs)

- Two forms of aggregation response
  - Zipped netCDF files
  - Concatenated tables (CSV)

  *N.B.* Arrays may be aggregated as concatenated tables!

**Live Demo…**
DAP Output-Encoding Extensions

DAP(2|4), netCDF, XML, GeoTIFF, ...
OGC Protocol: WMS Web Mapping Service

- WMS (Web Mapping Service)
  - Great for 2-dim geospatial data on ‘maps’ (but not for higher-dimensional data types)
  - A bridge to display tools, notably, Google Earth

Live Demo…
DAP Interoperability Leverage

Hyrax Back-End Server with Encoding-Specific Handlers

Apache Server Framework

Core DAP Services (Hyrax Front-End)

DAP-Extending Services

Other Web Services

Data User, Browser-Only
XML or JSON

Data User App (other standard)
OGC-Compliant

Data-User App (python, java, Native-DAP Libraries

NetCDF Libraries

Data-User App (netCDF-based)

DAP(2|4), netCDF, XML, GeoTIFF, WMS, JSON…

Other Web Services

HDF files
netCDF files
text files
SQL database
extensible…
relevance:

**OPeNDAP & Interoperability**

- We demonstrated
  - NASA (HDF5) files ➔ OpenDAP ➔ WMS ➔ Google Earth

- Notably, it seems unlikely that either
  - Google Earth engineers anticipated reading HDF5 or
  - NASA engineers planned to display data on Google Earth!

- This suggests* a definition for interoperability:
  “supporting unanticipated uses”

*Paraphrasing John Orcutt
This presentation, and the recent work described, were supported by NASA/GSFC under Raytheon Co. contract number NNG15HZ39C