10.6

Unidata's THREDDS Data Server

John Caron, Ethan Davis *, Yuan Ho, Robb Kambic
Unidata Program Center / UCAR

1. Abstract

The THREDDS Data Server combines catalog services with integrated data-serving capabilities, including OPeNDAP, HTTP file serving, and OGC Web Coverage Server (WCS). The OPeNDAP and WCS data-serving capabilities are built on the netCDF-Java library, which combines the netCDF-3, OPeNDAP-2, and HDF5 data models, into what is called the netCDF-4 Common Data Model. The THREDDS Data Server is 100% Java, open source, and runs as a Tomcat web server application. This paper details its capabilities and implementation status.

2. THREDDS

Unidata's THREDDS project provides middleware services to bridge the gap between data providers and data consumers. Originally funded through NSF's National Science Digital Library (NSDL) initiative, ongoing work is now funded under Unidata's core funds budget.

THREDDS c*atalogs* are XML documents that allow a data provider to list available on-line datasets. The catalog creator groups datasets into a simple hierarchical classification scheme, which makes a catalog a "logical data directory". Catalogs can be static XML files, or dynamically generated by Web servers to track continuously changing datasets. At a minimum, a catalog must specify a "human readable" name and an access URL for each dataset.

THREDDS catalogs also define a modest set of specific metadata elements that we recommend data providers add to their catalogs, making them into *enhanced* catalogs. These enhanced metadata elements were chosen to make the datasets more useable by THREDDS clients such as visualization and analysis programs like Unidata's IDV. They were also chosen to allow automatic creation of digital library records as well as their insertion into discovery centers like GCMD and NSDL.

Catalogs can also include arbitrary metadata using any XML elements from another namespace. This allows THREDDS catalogs to contain (or point to) other metadata, FGDC, Dublin Core, and DIF for example. THREDDS catalogs are thus general containers of metadata about any type of dataset, in fact about any resource that can be accessed with a URL.

3. The Common Data Model

While THREDDS catalogs are, in general, ignorant of the meaning and content of their contained datasets, the THREDDS Data Server is intended for datasets that can be read and understood using the Common Data Model.

The *Common Data Model* (CDM) is an abstract data model used by the netCDF-Java (version 2.2) library and the netCDF-4 project, which is a NASA funded project to develop a new version of netCDF using HDF5 as the file format. The CDM combines the netCDF, OPeNDAP, and HDF5 data models.

The netCDF-Java library currently reads netCDF, OPeNDAP, and HDF5 datasets into an implementation of the CDM, as well as other binary formats such as GRIB-1, GRIB-2, GINI, NEXRAD level 2 and 3, and DORADE radar files.

Beyond the low level data model, the CDM also adds two higher levels. First, the *Georeferencing Coordinate System* level builds on the lower level to provide an abstract model for coordinate and georeferencing information. Next, the *Scientific Data Type* level provides abstractions for domain specific data types, e.g., gridded data, station data, and imagery. These two levels provide the semantics needed to convert datasets to other protocols and formats such as those required by the WCS.

As with the low level data model, the netCDF-Java library provides a standard API for both the Georeferencing Coordinate System level and

* *Corresponding author address*: Ethan Davis, UCAR/Unidata, PO Box 3000, Boulder, CO 80307; email: edavis@ucar.edu

the Scientific Data Type level.

The form and availability of coordinate system and scientific data type information depends on the format of the data being considered. For general purpose formats and protocols such as netCDF, HDF5, and OPeNDAP, the library adds this information by parsing well known *attribute conventions* such as CF-1.0, and by using THREDDS metadata to add missing coordinate system information and other metadata. For file formats such as GRIB and NEXRAD, the library extracts the information by understanding these specialized file formats. The CDM and its implementation in the netCDF-Java library allow this information to be accessed, regardless of how the information is obtained, from a standard set of APIs.

4. THREDDS Data Server

The THREDDS Data Server (TDS) extends previous implementations of THREDDS Catalog Server by adding integrated data serving capabilities. It is implemented in 100% Java, and is contained in a single WAR file, which allows very easy installation into the open-source Tomcat web server. Configuration is made as simple and as automatic as possible, and we have made the server as secure as possible, for example by using SSL to access sensitive information such as logs.

The TDS has an integrated OPeNDAP server that provides access to any dataset that can be read through the netCDF-Java library into the Common Data Model. This is essentially an adapter of the CDM data model into OPeNDAP using the Java-DODS library.

The TDS provides bulk file access through the HTTP protocol. Using THREDDS *compound services*, it is easy to mark the same dataset as being available through both OPeNDAP and HTTP without having to duplicate any other information.

The TDS also has an experimental server providing data access through the OpenGIS Consortium (OGC) Web Coverage Service (WCS) protocol. This works for any gridded dataset whose coordinate system information is complete and whose axes are evenly spaced. Users can add missing information to a dataset in the TDS catalog where needed.

The TDS provides automatic catalog generation for files local to the server, which makes setting up a TDS very easy. Enhancing catalogs with appropriate metadata is still mostly done by hand and is the most time consuming aspect of TDS configuration. The common case is to group homogeneous datasets under a single directory, particularly ones that differ only in their time coverage. The entire set of data can then be served and metadata added with a single entry in the TDS catalog.

In general, working with groups of files rather than individual files is essential to manage the sheer volume of data at many data provider sites. THREDDS catalogs are trees of datasets, and metadata can be added to any dataset node and apply to all the datasets contained in the node. A particularly powerful way of managing groups of files is with NcML (NetCDF Markup Language) which allows adding, changing and removing attributes, variables, and even data in the datasets themselves. NcML can be added to TDS catalogs in a way that modifies all the datasets contained in a subtree. This allows data providers to easily correct or augment the metadata contained in their datasets.

It is possible to use the TDS to only generate/ serve catalogs or to only serve the data. However, the integration of catalog services and data serving significantly reduces the burden of server configuration and management. In particular, getting a simple data server up and running is very little additional work beyond deploying a Tomcat web server.

5. THREDDS / IDD Server

Much of the real-time data available over the Unidata Internet Data Distribution (IDD) is available through a THREDDS Data Server hosted at Unidata on motherlode.ucar.edu. This includes a significant portion of the NCEP model runs, the National Digital Forecast Database, both the level 2 and level 3 NEXRAD data, GOES satellite data, and Metar station data. Currently we are maintaining the latest seven days worth of data. We plan to continue to add real-time datasets from the IDD onto the motherlode server, and to add enhanced metadata.

A special version of the TDS with catalogs and corresponding pqact files for the LDM are available to other groups who want to make IDD data available over a THREDDS server.

6. Future Plans

An immediate priority is to make catalog generation as easy and as powerful as possible. To this end we are integrating the functionality of the THREDDS *Catalog Generator* program directly into the TDS. We are also investigating the automatic extraction of metadata from the datasets, and other ways to ease the burden of adding enhanced metadata.

We will also be adding the capability to restrict user access to different catalog subtrees where applicable. The default implementation will use Tomcat user/role authentication, and hooks will be provided to allow others to add their own authentication mechanism at runtime. Note that authentication must be done by the data server, not by the client, so the integration of catalog and data server is particularly useful.

An important and much requested capability is to group multiple files together into a single dataset. This is what the OPeNDAP Aggregation Server (AS) does. We are adding this functionality to the CDM layer so that different data servers can use it, rather than being limited to the OPeNDAP server. The AS is therefore not being developed and new users are advised to wait and use the TDS for this functionality. The exact way that the CDM and TDS will interoperate to provide aggregation is still being determined.

Aggregation capabilities lead to the need for more complex capabilities like the ability to query large datasets. Querying can be thought of as a way to subset a dataset at a higher semantic layer than netCDF array index subsetting. Two major outstanding issues to be resolved are: 1) what queries will be allowed against any particular dataset, and 2) in what form will the result be returned?

Both OPeNDAP and ADDE protocols provide a way to query, and the THREDDS Dataset Query Capability (DQC) documents are a first pass at defining, in a general way, how a query can be formed. In the future we expect the TDS to be good framework to provide at least some query functionality on the server, tying in with CDM scientific data type APIs on the client side.

As server capabilities become more complex, Unidata expects to continue to move towards *web services* to define their interface. Whether this remains simple XML transported over HTTP, as in the current implementation, or changes to use other layers such as SOAP and WSDL remains to be seen. Certainly binary data transport will remain a necessary part of efficient data serving. We will continue to work closely with OPeNDAP, the OGC, and others to help shape the future of remote access for scientific data.

Glossary

http://www.unidata.ucar.edu/publications/acronyms/glossary.html

References

Common Data Model
http://www.unidata.ucar.edu/software/netcdf/CDM/index.html

Dataset Query Capabilities Specification
http://www.unidata.ucar.edu/projects/THREDDS/tech/dqc/DqcSpec.html

NetCDF-4 Project
http://www.unidata.ucar.edu/software/netcdf/netcdf-4/

NetCDF-Java library
http://www.unidata.ucar.edu/software/netcdf-java/

OGC Specifications
http://www.opengeospatial.org/specs/?page=specs

OPeNDAP
http://www.opendap.org

THREDDS Catalog Specification
http://www.unidata.ucar.edu/projects/THREDDS/tech/catalog/InvCatalogSpec.html

Unidata IDD/LDM
http://www.unidata.ucar.edu/software/ldm/