8B.7

The THREDDS Data Repository: for Long Term Data Storage and Access

Anne Wilson, Thomas Baltzer, John Caron
Unidata Program Center, UCAR, Boulder, CO

1 INTRODUCTION

In order to better manage ever increasing volumes of scientific data, infrastructure in the form of intelligent data servers is needed. Large data volumes mean that server-side subsetting and aggregation are essential server requirements. But simply being able to serve data is no longer sufficient. Data must also be discoverable, queriable, and browsable. Thus these servers must also provide a cataloging system and tools to manage and access metadata.

Data generators and providers want to be able to publish both transient and archival data to a reliable and secure location that allows access to a community of data consumers. These providers want protection from time consuming housekeeping tasks such as locating storage, moving the data, and updating metadata catalogs. They also want to be able to easily configure their data space in a meaningful manner.

Similarly data consumers want to be free from issues related to locating and acquiring data, as well as format incompatibility issues. Such chores are often a time consuming distraction from the actual task to be performed. Indeed, such a level of automation is required in order for the repository to be used by an automated data generator or consumer, such as an data processing system that acquires data for inputs and generates data as outputs.

The Unidata THREDDS Data Repository (TDR), currently under development, provides long term data storage with metadata management support in the form of THREDDS catalogs. Integrated with the THREDDS Data Server (TDS), the two together provide automated, high level data and metadata storage and access.

Through use cases provided by two TDR client projects, this paper discusses TDR requirements and the functionality designed to meet them.

2 TDR CLIENT STORIES

There are two projects that are steering design requirements for the TDR.

2.1 The LEAD Project

Linked Environments for Atmospheric Discovery (LEAD) is a multi-institutional Large Information Technology Research (ITR) project funded by the National Science Foundation (NSF). The goal of LEAD is to create a framework based on Grid and web services to support mesoscale meteorology research and education via provision of capabilities such as running a forecast model over a spatial and temporal domain specified by the user, data mining for meteorological phenomena, and dynamic orchestrations that automatically reconfigure themselves in response to changing weather. LEAD accomplishes these tasks via orchestration software that invokes and manages component services. ("Workflow" is another term for "orchestration".)

LEAD presents unique challenges in managing and storing large data volumes from real-time observational systems as well as data that are dynamically created during the execution of adaptive workflows. Furthermore, LEAD users want one stop shopping to visualize data and store them and their associated metadata in a location where they could be made private or be accessed by

*Corresponding author address: Dr. Anne Wilson PO Box 3000 Boulder Colorado, 80307: anne@unidata.ucar.edu

others, both individuals as well as services in the orchestration.

LEAD orchestrations must retrieve data required for their use. They also generate intermediate and final data products to be stored in the user's space. An orchestration can also acquire and generate metadata for these generated products, such as user information (including security tokens such as certificates or encoded passwords), data format, data size, provenance information, etc.

LEAD metadata is organized around individual users, whose space is structured as projects that contain experiments.

2.2 The STORM Case Study Archive

The Unidata STORM Case Study Archive is a repository for case studies. This resource will provide community access to case studies compiled and annotated by case study authors. The archive will store data, metadata, and other resources such as notes, links to other resources, IDV bundles (a collection of data references and their rendering created by the Unidata Integrated Data View or IDV), etc.

Case study authors need to be able to move data into the repository, provide metadata, and ensure that the resources placed in the repository, both data and metadata, are accessible to the community they are intended to serve. They will need to be able to update the metadata. They will want to be able to organize resources in the repository, moving datasets and associated metadata around in the case study hierarchy. For STORM, the metadata is structured around case studies, such as "Katrina" and "Rita". Multiple case study authors may be authorized to add or change information in a case study.

Users of the case study repository will be able to access, visualize and acquire these resources in useable formats.

2.3 Commonalities

Both of these projects require support in the areas of:

- Long term data and metadata storage
- Data movement into and out of the repository
- Metadata generation and management
- High level data access functionality, such as server side subsetting

3 The THREDDS Data Server (TDS)

The Unidata THREDDS Data Server addresses several of these concerns. The TDS is designed to serve a moving window of recent scientific data and currently does so via the protocols http, OPeNDAP, and WCS for gridded data. It can provide the data in the format in which it is stored on the disk, and is working towards generating netCDF files from a variety of data formats through the Unidata Common Data Model (CDM). (The list of CDM supported data formats currently includes: NetCDF, level II and level III radar, Grib, GINI, and HDF5.)

Also, the TDS supports metadata maintained in the form of a hierarchy of catalogs that contain information such as data format (e.g. NetCDF, GRIB, HDF5, etc.), type (e.g. grid, image, point, radial, etc.), data size, spatial and temporal coverages, variables (if applicable), descriptions, creators, publishers, rights, etc.

Although Unidata does maintain some data archives, an important TDS goal is for users to easily implement and maintain their own data server. For this reason, the TDS is an integrated, easy to install package consisting of THREDDS cataloging software, an OPeNDAP server, a WCS server, and other supporting software.

4 The THREDDS Data Repository (TDR)

While the TDS was designed around serving transitory data, the TDR is intended to fill in the long term storage aspect of the TDS. It provides data movement into a long term repository and ensures data access from the repository through integration with a co-residing TDS. In addition to providing storage for the data, it supports incorporation of preexisting metadata and generation of new metadata into a metadata hierarchy that, in later versions, will be editable through a catalog and metadata management interface.

Indeed, the basis of the TDR is metadata management. The handle to the actual data can be viewed as simply another metadata element.

Information needed by the TDR to put data into the repository includes:

- information about a user, used in authentication, authorization, and elsewhere,
- a reference to the data to be stored, either a local file or a URL,
- any preexisting metadata,
- and, a "logical name", such as "Katrina/data/radar/level2", used to position the metadata in the metadata hierarchy.

The successful completion of a request to put data into the repository returns a unique ID, a non-semantic sequence of characters that is a handle to the data. Either this ID, or the original user information and logical name can be used to retrieve the data.

The use cases dictate that the TDR provide two types of interfaces. One is via a web form where users can enter information to update the repository. The other is via programmatic calls from one web service to another. The TDR is implemented as a Java servlet, which meets both of these goals.

For the STORM project, case study authors are provided with a web form that allows users to enter a URL or browse for a local file to be a data source. Also, the user can enter metadata such as the data type and format if it is known. Users also provide a project name and logical name for the data being stored via this form. The programmatic interface provides the same information via Java networked I/O.

Once the TDR receives a request either via the web interface or programmatically it performs the following sequence of actions:

1. Locate storage for the data.
2. Generate a unique ID for the data.
3. Copy the data to the storage.
4. Generate additional metadata.
5. Incorporate the metadata into the existing metadata hierarchy.

We will examine each of these tasks individually. These different jobs are each performed by respective specialized task managers: the Storage Locater, the Unique ID Generator, the Data Mover, the Metadata Generator, and the Cataloger.

4.1 Locate Storage

Locating storage for the data entails knowing where there is space available to put the data. Thus the Storage Locater must track space availability. It is the job of the Storage Locater to ensure that the user is authorized to use a space and that by doing so they will not violate any space management policies. Depending on authorizations, space available could include a variety of storage devices, such as space on a mass storage device.

4.2 Generate a Unique ID

This ID will be used as a handle to the data. It must be guaranteed to be unique within the domain in which it exists. For the TDR that is serving both LEAD and the STORM Case Study Archive, the ID must be unique within all the catalogs served by the TDS. (Algorithms exist that generate IDs with such an extremely low likelihood for duplication that they can be considered unique.) The Unique ID Generator performs this task.

4.3 Copy the data to the storage

The Data Mover is responsible for copying the data to the target destination. The Data Mover makes decisions about how to move the data based in part on the protocol portion of the URL of the source file. Http service is assumed to be available at the data source, but if the protocol indicates a better option, such as gridftp, that will be tried.

4.4 Generate Additional Metadata

New metadata to be generated includes recording where the data now physically resides. (This mapping from unique ID to physical location could get complex if replication was involved. In such a case name mapping could be achieved via a separate Name Resolution object or service.)

Perhaps the most important additional metadata generated is the creation of access URLs for the data. In order to create these an attempt must be made to know the data

format. The Metadata Generator is currently using the following logic to determine how the data can be served:

1) All files can be served via http.

2) If there is a gridftp server present, all files can be served via gridftp.

3) If the file can be successfully opened by the Common Data Model file accessor, then it can be served via OPeNDAP.

4) If 3 above is true and the file is known to be a grid, then it can be served via WCS.

4.5 Incorporate the metadata into the Metadata Hierarchy

The TDR updates the appropriate catalogs with the new metadata, which involves the creation of new datasets in the existing catalogs and may also involve the creation of new catalogs. The logical name is used to determine where in the metadata hierarchy to place the new dataset. Reflecting the logical path may involve the generation of nested datasets.

The TDR then signals the TDS to reread the catalogs in order to incorporate the changes. Now the data and metadata can be served via the TDS.

5 FUTURE DIRECTIONS

A prototype implementation of the TDR is available on the Unidata LEAD test bed. It is currently handling input for both clients.

The next important functionality to be added is to support the handling of a collection of files in the form of a compressed zip file. It is assumed that these files are of the same data type so that a single chunk of metadata can be applied to the entire set. Also, general repository "editing" capabilities need to be provided, such as the ability to add and edit metadata and remove data and its associated metadata.

6 ACKNOWLEDGEMENTS

LEAD is a Large Information Technology Research (ITR) Grant funded by the National Science Foundation under the following Cooperative Agreements: ATM-0331594 (University of Oklahoma), ATM-0331591 (Colorado State University), ATM-0331574 (Millersville University), ATM-0331480 (Indiana University), ATM-0331579 (University of Alabama in Huntsville), ATM03-31586 (Howard University), ATM-0331587 (University Corporation for Atmospheric Research), and ATM-0331578 (University of Illinois at Urbana-Champaign, with a sub-contract to the University of North Carolina).

7 REFERENCES

Droegemeier, K.K. and Co-Authors, 2005: Service-oriented environments in research and education for dynamically interacting with mesoscale weather. *Computing in Science and Engineering*, **7**, 12-29.

Droegemeier, K. K. and Co-Authors, 2005: Linked environments for atmospheric discovery (LEAD): Architecture, technology roadmap and deployment strategy. Preprints, *21st Conf. on Interactive Info. Processing Systems for Meteorology, Oceanography, and Hydrology*, San Diego, CA, Amer. Meteor. Soc.

Droegemeier, K.K. and Co-Authors, 2005: Service-oriented environments in research and education for dynamically interacting with mesoscale weather. *Computing in Science and Engineering*, **7**, 12-29.

Murray, D., J. McWhirter, S. Wier, and S. Emmerson, 2003: The Integrated Data Viewer—a web-enabled application for scientific analysis and visualization. Preprints, *19th Conf. On Integrated Information and Processing*, 8-13 February, Long Beach, CA, Amer. Meteor. Soc.

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

OPeNDAP Project http://www.opendap.org/

The Integrated Data Viewer (IDV) http://www.unidata.ucar.edu/software/idv/

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

The THREDDS Data Server (TDS) http://www.unidata.ucar.edu/projects/THREDDS/tech/#TDS