Lessons Learned Regarding WCS Server Design and Implementation

Status of this RFC

This RFC Technical Note provides information on lessons learned from an Open Geospatial Consortium (OGC) Web Coverage Service (WCS) server design and implementation. This RFC does not specify an Earth Science Data Systems (ESDS) standard. Distribution of this memo is unlimited.

Change Explanation

This document is not a revision to an earlier version.

Copyright Notice

Abstract

This document provides lessons learned regarding a WCS server implementation for the Coordinated Energy and Water Cycle Observations Project (CEOP) Satellite Data Server, a NASA ACCESS project that provides a gateway between the Open-source Project for a Network Data Access Protocol (OPeNDAP) and WCS protocols. The gateway allows a user to use an OPeNDAP-enabled client to access satellite data held at a WCS server with services such as subsetting, reprojection, mosaicking and time series support. This project was done as part of the Committee on Earth Observation Satellites (CEOS) Working Group and Systems and Services (WGISS) project activity, the WGISS Test Facility for CEOP. This Technical Note presents several challenges encountered in the design and implementation of the WCS server, mostly arising from user access patterns, as well as the approaches taken to address those challenges.

1 Introduction

This Technical Note describes lessons regarding a Web Coverage Service (WCS) server implementation from a NASA ACCESS project to provide a gateway between the Open-source Project for a Network Data Access Protocol (OPeNDAP) and WCS protocols. The project developed a data handler for the OPeNDAP server that enabled serving of data obtained from a WCS server. The aim was to allow access to the data by analysis clients that have OPeNDAP support but not WCS support. For its part, the WCS server enables the serving of high-resolution satellite swath data (suitably reprojected to the WCS supported Earth spatial reference system) in a gridded form intelligible to the OPeNDAP client. A useful byproduct of the gateway architecture also allows third parties to provide OPeNDAP interfaces to data they do not actually hold but simply access remotely through WCS.

The project was user-driven, by the Coordinated Enhanced Observing Period (CEOP) community, which afforded us the opportunity to see how the WCS server implementation served (or did not serve) the analysis clients employed by those users. This project was done as part of the Committee on Earth Observation Satellites (CEOS) Working Group on Information Systems and Services (WGISS) project activity, the WGISS Test Facility for CEOP.

The user input for the project revealed several challenges not normally faced by WCS servers. A key revelation was the expectation that many of the users in question were likely to want to analyze long time series of data for very small geographic areas. This contrasts with most WCS access patterns, which tend to concern short time periods (or even neglect time entirely.) This in turn raised a number of challenges that rippled throughout many aspects of the project, including the WCS server design and implementation. In addition, the projected end-user clients, such as the Gridded Analysis and Display System (GrADS) provided additional constraints and challenges. Finally, user questions about the content of what they were actually seeing pointed out the need to consider quality screening and provenance in the WCS server.

The key lessons learned from attempting to serve this time-oriented community with WCS technologies are relevant to archives contemplating the use of WCS to serve data covering long time periods (*i.e.*, years) to the science research community. Although some of these lessons may have implications as to how the OGC WCS protocol might be extended to better serve the temporal dimension, our primary goal is document our experiences for the benefit of WCS server implementers .

2 Design of CEOP Satellite Data Server

The CEOP Satellite Data Server was initially designed to include a gateway (middleware) between a WCS server for NASA satellite data and the OPeNDAP server. However, the rearchitecture of the new version of the OPeNDAP server, Hyrax, allowed the key gateway functionality to be implemented instead as a “format handler” in the Hyrax’s Back-end Server (BES). Figure 1 shows the eventual high level design of the CEOP Satellite Data Server. The handler can be configured during the installation of the Hyrax server. In addition, significant customization of the WCS server implementation and configuration was necessary as well and forms the main basis for the lessons documented in this Technical Note.

The basic process of fulfilling a request is:

1) The client submits an OPeNDAP request to the CEOP Satellite Data Server, which is handled by its front end, the OPeNDAP Light Front End Service (OLFS).
2) the OLFS interacts with local catalog to identify the data source as WCS;
3) the OLFS instructs its BES to set container type to WCS and passes identifying information about the data to be retrieved;
4) BES formulates a WCS request to the WCS server;
5) BES stores the WCS response to local cache;
6) BES uses the NetCDF format handler to process cached file to satisfy the Data Access Protocol (DAP) request; and
7) Subsequent DAP requests operate against local cache.

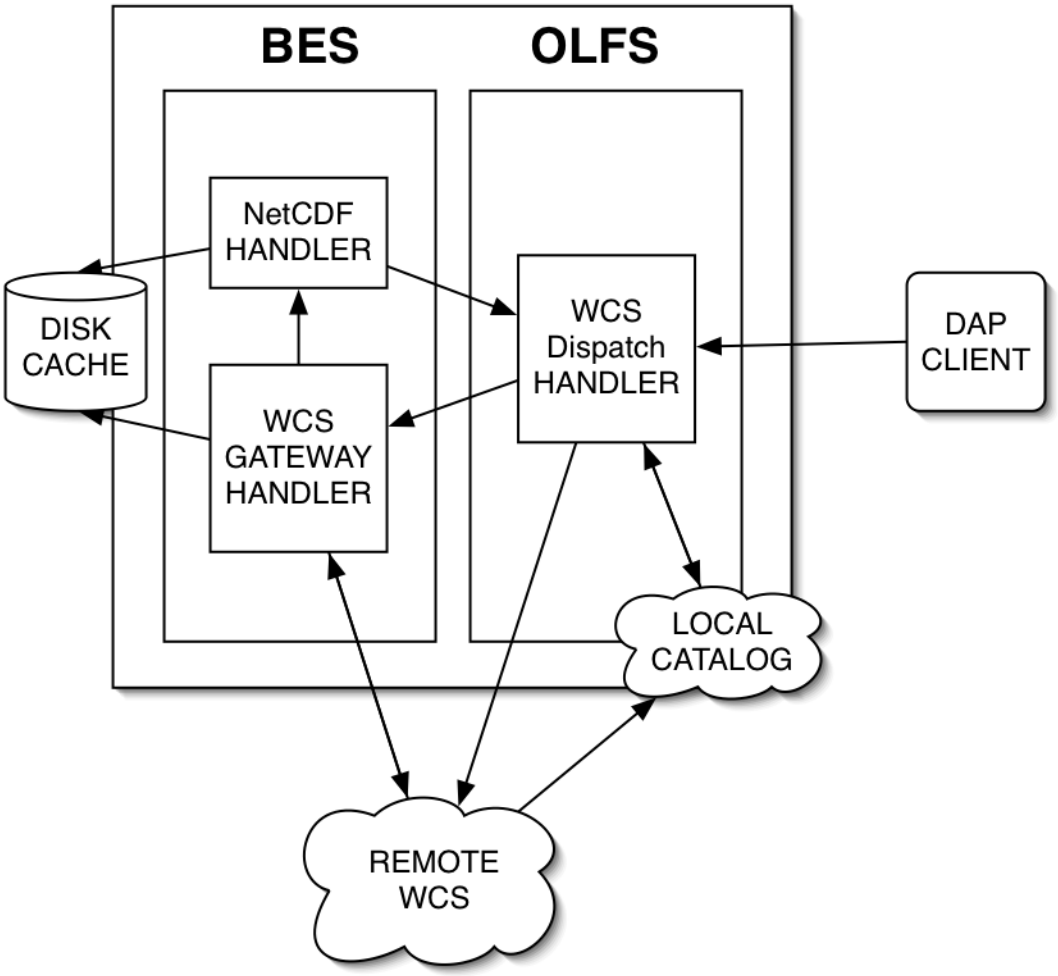

Figure 1. High-level Conceptual Design of the OPeNDAP / WCS Server.

This document focuses on the design and implementation of the WCS component interfaced to the CEOP Satellite Data Server. In fact, wo separate WCS servers were implemented, though sharing a substantial codebase. A WCS server was implemented to serve daily global coverages of an Atmospheric Infrared Sounder (AIRS) Level 2 Standard Retrieval product (aka AIRX2RET) with actual CEOP requests focused on CEOP observation sites, essentially 250x250 km squares around CEOP reference sites. Although in theory, this server could respond to general, arbitrary user WCS requests, its primary purpose was to act as the back end to the CEOP Satellite Data Server, performing essential reprojection and mosaicking of AIRS Level 2 data, Thus, the 250x250 km CEOP reference site squares might be thought of as virtual products, generated on the fly. The global daily coverages presented to the client are actually the virtual products, as they are not archived in the Goddard Earth Sciences Data and Information Services Center (GES DISC). The WCS server generates these coverages from individual 6-minute granule files physically stored in the GES DISC's operational archive. The second WCS server was implemented within George Mason University's GeoBrain environment to serve MODIS data. In this implementation, individual granules were mapped to coverages.

3 Client-driven Issues

In this project, the end goal was to make the data transparently usable within analysis clients, such as GrADS. As it happened, this imposed some basic constraints not faced by a WCS server designed to serve more generic WCS clients, such as commercial ones.

3.1 Length of WCS Request URL

Coverages provided by WCS servers that serve high data volumes are generally based on the physical data files in server databases or file systems. That is, a server offers one coverage for one physical file. Our MODIS server is implemented in this approach. Because one MODIS file usually contains more than one scientific parameter, additional parameter identification is needed after a physical file name, which itself can be quite long due to inclusion of date and time. This results in a coverage name dozens of characters long, e.g.:
MOD05_L2.A2004209.0230.005.2007025064844.hdf:Grid:mod05:Water_Vapor_Near_Infrared

The long coverage name, together with other parameters in the WCS *GetCoverage* request, can make the request URL too long to be handled by some analysis client software. Note that this problem is not specific to the WCS protocol, but can become significant when WCS is used as a component of a distributed workflow.

3.2 Encoding of Special Characters in URL

In addition to the length of URL, special characters included in coverage identification may also cause problem in application software, for example, characters '&', ':', and '%' in coverage identification may cause failure of the application software. Again, this becomes a potential factor when WCS URLs are embedded in distributed workflows for pass-through.

3.3 URL Transparency

A popular feature in GrADS is the ability to script analysis on large assemblages of files using "templating". This requires access handles (filenames or URLs) that are both meaningful and predictable.

There were actually two servers fielded for this project, which had very different properties with respect to the transparency of the URLs. The server for MODIS data presents each data file separately, without mosaicking, while the server for AIRS data uses mosaicking to present each variable as a global coverage. One issue in the per-file based MODIS server is that the physical file path and physical file name are exposed in the offered coverage name to the users. This name includes information which is confusing and opaque to users, such as the inclusion of both product observation and generation times in such file names as *MOD05_L2.A2002274.0115.005.2006332104856.hdf*. Worse, the data observation time requires an orbit model to predict for a given location, and the data generation time is completely non-deterministic, thwarting users attempting to script access to multiple coverages. Our users commented that they prefer service URLs with indicative information such as variable type and spatial and temporal extents. For the AIRS server providing virtual coverage, the coverage is simply named using variable name, such as "TSurfAir" for surface air temperature. The URLs viewed by the users are constructed in the Hyrax server with indicative information included, such as:
http://test.opendap.org:8080/opendap/wcs/CEOP/BRA/ceopL2AIRSTimeOffsetLineage/TSurfAir/2002-10-02.html

in which the CEOP reference station name (BRA), variable, and time are clearly presented.

3.3.1 Lesson learned

It is relatively easy and straightforward for the Hyrax server to construct informative URLs for end users when the WCS server supports virtual coverages. On the other hand, the per-file based WCS implementation with the physical file name exposed is less than ideal. The near term solution to this can be to use a logical versus physical file name lookup table to map physical file name to a more concise and informative logical name and the server can expose a physical file with its associated logical name. For example, the MODIS file name mentioned in the previous paragraph can be replaced using a more informative and predictable name such as MODIS_WaterVapor_20021001.

3.4 Choice of Profile

Most important was the ability of the client to understand the contents of the results. This was the prime motive for using the NetCDF/CF1 profile in the WCS server: the GrADS clients used by many CEOP users can open and read NetCDF/CF1 with a single call, as can other clients like Ferret and Panoply.

4 Coordinate Reference System (CRS) Transformation

4.1 The Necessity of CRS Transformation

Many of the data products served are swath data whose values are geometrically aligned with satellite swath coordinate system, while data users usually need data in earth coordinate, geographic or projected reference system. CRS transformation (CRST) is therefore necessary in the gateway. Allowing users to request CRST in a standard way is one of the key advantages that the OGC WCS protocol provides. In the broad sense, CRST includes any change of the CRS associated with a dataset, including transformation from satellite swath CRS to geographic/projected CRS and various reprojections among projected CRSs. Our user community (i.e., the CEOP scientists) is mostly interested in using the satellite data in geographic latitude/longitude coordinate reference system. Thus, the CRST implemented in our server is mainly swath to geographic CRS transformation.

4.2 The WCS Approach to Describing CRST

The WCS specification provides a very simple standard parameter in defining how a CRST should be done, i.e., CRS. A server lists all the CRSs it can offer in an XML element named *SupportedCRS* and a client specifies "CRS=crsValue", where *crsValue* must be one of the values listed in the SupportedCRS element, to specify a desired CRS. Although the CRST itself is theoretically a geometrical processing of data, it is actually influenced by the nature, or scientific meaning, of a data product being served, because it is essentially deriving new data values from input dataset values through interpolation and/or resampling. WCS version 1.0.0 allows clients to use the *InterpolationMethod* parameter to specify a method to be used in deriving new values in desired CRS from input values in input CRS. Three classical digital image processing interpolation methods, namely nearest neighbor, bilinear, and cubical convolution, are prescribed in WCS1.0.0. A server may offer one or more of these methods and a client can select any one of the offered methods.

4.3 Implementation Options

While the WCS protocol defines CRST and related parameters, different servers may select different approaches to implement a CRST. There are usually two ways to perform satellite swath to geographic CRST. One is a forward method in which the data points in swath CRS are mapped to the output geographic CRS and interpolation is performed in the output coordinate system. The other is a backward method in which the data point coordinate values in the output grid are mapped backwards to the input swath coordinate values and interpolation is performed in the swath CRS. In our implementation, we used forward mapping and nearest neighbor interpolation during the forward mapping. The reason to use forward mapping is that there is no straightforward way to find swath coordinate values from geographic coordinate values. In our earlier implementation, we used a polynomial fitting function to derive swath coordinate values from geographic coordinate values. The polynomial fitting function, usually with an order not exceeding three, performs faster than point-to-point forward mapping but lacks accuracy at the edge of a swath, unless multiple fitting functions are used, which will result in additional mosaicking processing at the boundaries of the valid regions of different functions.

In addition to the coordinate transformation and interpolation, the output grid point value may also involve further aggregation, as opposed to resampling, when the request grid cell size is larger than the input swath cell size. The algorithm is product specific, depending both on the physical parameter measured and its representation (e.g., numerical vs. categorical). The most common approach is to average the multiple input data points falling into one output grid cell, but there are also others such as picking the majority (mode), maximum, or minimum value. Currently all of our offered coverages are numerical data types and we used an averaging method to perform aggregation. Further implementation will include the majority rule for nominal (categorical) data.

4.3.1 Lessons learned

CRST, or georectification/reproject, must be product specific and parameter-specific. Parameters in the WCS interface protocol are not sufficient to define the difference among products.

5 Mosaicking of Multiple File Granules

Most currently available WCS servers offer coverages based on physical files, i.e., each physical file is offered as one coverage. While this is the most straightforward implementation and reflects a clear one-to-one relationship between a physical file and a coverage, it may not be desirable when a user's area of interest exceeds the extent of one single physical file, in which case multiple requests must be used to get enough data for the interested area and the multiple returned coverages need to be mosaicked at the client side. The alternative is for a server to offer a virtual coverage that is not tied with any one single physical file in the server's file system. The server dynamically determines which physical files are associated with each client request and performs mosaicking, when necessary, of data from multiple files. The advantage of this implementation is that the mosaicking is transparent to clients and the offered coverage can be described in a very general way. The disadvantage is that it may take much longer for a server to respond due to the more complicated processing steps involved in the server side. The client side mosaicking is usually much simpler because the coverages to be mosaicked are already in the desired geographic CRS. In addition to more complicated processing steps, a server needs to estimate the memory and speed (i.e., space and time) of the server machine in determining how general a virtual coverage can be. Limits on spatial and temporal extents may need to be imposed to make sure the generalization does not exceed server's capability.

As mentioned earlier, the AIRS WCS server offers virtual coverages where each variable is offered as a daily coverage whose spatial extents covers the entire globe. Such virtual "global" coverage may involve many physical files (240 files per days in our AIRS Level 2 product case). This was refactored from the original implementation of separate, spatially constrained coverages for each CEOP reference site (there are about 30), which produced scalability problems in the gateway due to the combinatorially large number of resultant coverages (stations * parameters * days).

When a client requests an output coverage with a large spatial and temporal extent, the server may not be able to fulfill the request. To mitigate this, we put a constraint on the temporal extent of less than one day, from 0-hour to 24-hour. Unfortunately the WCS specification does not

define how to describe such a constraint in a machine-understandable way. Our server simply responds with an exception message when a request crosses the 24-hour boundary. The message, in free text, is informative to a human user but not machine parsable. In addition to that on the temporal extent, constraints on spatial extent can also be applied. We used a temporal constraint because our user's study areas, the CEOP sites, are globally distributed, allowing a global spatial extent enables users to request data for any CEOP site. When the WCS server is integrated with the OPeNDAP Hyrax server, coverages from individual days are assembled into time series so that a user can request coverages for any CEOP site and at any time length within the two year (2002-10-01 to 2004-09-30) CEOP observation period.

5.1 Lesson learned

1) It is desirable for a WCS server to provide a virtual coverage service with server-side mosaicking of multiple physical files on demand. A client need not know the relationship between the offered coverage and the actual physical files in the server. Such a server capability, on the other hand, may be at the expense of either the server's performance or its spatial/temporal extents, depending on factors such as file size, internal storage and transfer capability, and encoding.

2) For servers performing on-the-fly-georectification, limits to spatial and temporal extents often need to be applied due to performance considerations. In our case, this was not so much a function of exactly which CRSTs were supported. Rather it was a function of the two-year time period supported, raising the spectre of a global request potentially using as many as 175,000 input swath granules. One issue with applying temporal limitations is that WCS does not have a machine-parsable way to describe such a limitation to the client.

6 Time Stamps

The WCS specification provides a *Time* keyword with which a user can define desired temporal extent for a data product. The keyword is flexible enough to allow specifying either time points or time periods. It is expected that the output coverage from a server should exactly reflect the time requirement requested by a user. For example, when a requested time is accurate to a minute, the values in the output coverage should reflect the measurement of that specific minute, and when the time is accurate to a day, the value should be a daily measurement. In actual use scenarios, however, things may become more complicated, depending on the data product being served. For pre-gridded products such as hourly, daily, and monthly accumulated precipitation, the server can offer unambiguous time information and a user can obtain coverage with temporal extent matching exactly with what is requested, such as to specific hour(s) and day(s). However, for ungeorectified swath data, approximation and customization in terms of time may be necessary.

With our Level 2 AIRS products, our server offers "daily" coverages allowing a user to request data for a specific day. However, the observation times for different points in the coverage are not the same and thus coverage values do not actually represent a daily measurement. That is, the actual observation times of the swaths that contribute to a 250x250km square vary from day to day and may even span more than one orbit. In addition, there are both day and night crossings in many cases. Therefore it is necessary to discuss with the intended users on how the time

information can be best represented and used. Our discussions with users led us to group the observation into two groups, one representing day time and the other night time. Thus, the output coverage would have a day time component and a night time component, i.e., the coverage data array has a "time" dimension whose size is 2. Initially, we used the average of day time observations and average of night time observations to represent the day and night components of the coverage, respectively. Such average times are not fixed at specific times during a 24-hour period, but rather change with different spatial locations and days. This variation is difficult to handle in the GrADS software when constructing time series data for temporal analyses. After further tests, our users suggested that we use two fixed times, one 09:30 and the other 19:30, to represent day and night components of the output coverage. Since these are not the *actual* observation times, an additional data array is needed to record the exact observation time for each grid point in the returned coverage file. This additional data array may not be easily used by users.

6.1 Lessons learned

1) There are cases where the WCS coverage specification must be understood differently due to special data product characteristics. The temporal and spatial dimensions in WCS are assumed to be perpendicular (or independent). In our swath data product, this assumption is not true. The spatial location of a coverage grid point changes as time changes. This leaves the WCS implementor on the horns of a dilemma: in mosaicking, the observation times can be fixed at a certain granularity (daily in our case), in which case the accuracy of the observation time is sacrificed. Alternatively, the implementor can fall back to "granule-based" coverages, which preserve most of the temporal accuracy, but with some loss of usability as noted in section 5.

2) The server provider should consult with intended users to decide how best to represent the time values in such output coverages. It is possible that different users may have different requirements and thus different versions of server implementations may be needed. This is somewhat at odds with the goal of the WCS interoperability, i.e., WCS servers and clients should be interoperable at machine to machine level without having to make special arrangements between specific servers and clients. Solving this conundrum would require a more nuanced representation of time in the WCS specification that can account for the cases where spatial coverage is covariant with time. At this point, our exercises indicate that a mutual understanding between service providers and service recipients on product specifics is required for a useful and usable exchange for applications (such as much scientific research) that are sensitive to temporal accuracy.

3) Another note about the time dimension in our server is that our current implementation of dividing day and night components is actually not consistent, in strict understanding of WCS protocol, with the offered (and also requested) time description. When a client requests one single time, i.e., a day, our server responds with a coverage having a time dimension of size of 2, each for day time and night time, respectively. This may cause problems for more general clients that do not have prior knowledge of the returned coverage. An alternative is to offer two coverages for each day, a day time coverage and a night time coverage. A client would need to send two requests and receive two separate coverage files for a specific day. However, this would degrade the performance at both server and client sides.

7 Handling Ancillary Information

7.1 Quality Screening

Quality screening is generally product-specific. Though some products have simply good and bad data, others range up to complicated bitmasks defining different quality attributes. Thus, it is difficult to define general parameters within the WCS protocol to drive such screening. An alternative is for the service provider to implement specific screening approaches for the products being offered. Again, this may sometimes involve intended users because a specific user (or user community) may have special requirements with respect to data quality. For example, many of our AIRS variables have different quality levels (e.g., good and best) and different users may have different criteria. At the implementation level, a service provider may also have alternatives. For example, a server may pack the quality control values into a coverage file for users who want to check the qualities themselves. In our implementation, however, we perform the quality screen at the sever side in the interest of usability and do not provide additional quality fields to the client. We included in our output coverage all data points that are not labeled as "do not use" . The criterion can be changed, based on user feedback, in the server configuration file.

7.1.1 Lesson learned

For scientific research, it is critical that the data be properly screened for quality. The WCS specification does not account for quality screening, so our approach instead was to implement server-side screening to eliminate bad quality data. We did not get many comments from our users on how quality screening should be performed and how QC data should be packed into the output coverage, partly because our server-side implementation of screening made the process transparent to them. However, in the long term, the adequacy of this solution needs extensive usage in real applications, which unfortunately exceeds the project cycle. (With extended maintenance of the server at the Goddard Earth Sciences Data and Information Services Center, this user feedback might be obtainable, allowing us to revise our quality approach as necessary.) Should users require more flexible ways of screening or assessing the quality of data served through a WCS server, some additions to the protocol might become necessary.

7.2 Including Ancillary Information in Output Coverages

Aside from the per-gridpoint time and quality assurance in the previous discussion, there are potentially more ancillary fields including viewing and illumination geometry arrays, cloud percentage, standard deviation, and measurement/retrieval error value. Whether such information should be included in the output coverage depends on users' specific needs. A server has different options to expose such information to users, such as in the coverage description and metadata. The simplest, most straightforward way for the server is to offer these ancillary data arrays as coverages in their own right, just as the main variable coverage. For example, a server may offer land surface temperature and its associated cloud fraction as two separate coverages. A user needing cloud fraction to evaluate the land surface temperature may want to acquire both coverages. This implementation can avoid potential difficulties in data encoding because some data encoding formats do not allow multiple data arrays to be packed

into one single file. The drawback to this is that ancillary variables that should be bound to the main coverage are instead separated from it, complicating data management on the user's side.

7.2.1 Lessons learned

We currently do not include ancillary data arrays in output coverage. However, the output coverages in our service are already screened using ancillary data information, particularly the quality control arrays. When required, the WCS server can be revised to also offer ancillary data coverages so that a user will be able to request such coverages. A potential issue for such an implementation is that it will require user to link an ancillary coverage with the actual variable coverage, which may be difficult with many variables and ancillary coverages, especially when the ancillary coverages are not simultaneously requested with the variable coverage. This places a burden on the end user, if human, or even the WCS client developer in matching up the proper ancillary coverage with the variable coverages. In contrast, it is easier for the end user if all related ancillary data are packed with the variable data in one output physical coverage file. Even in this case, a number of additional issues arise, such as where in the payload to include the information where it can be discovered and accessed, as well as how to translate into standard or conventional units.

7.3 Provenance Information

Early on, it became apparent that service chaining risked the obscuration of the source data and the processing applied to that data by the services. In our case, for instance, Level 2 swath data were first quality screened and then reprojected to a geographic grid. These are significant operations: should anything look "funny" in the data, a user is likely to want to go back to the original source data for comparison. Indeed, the user is generally not even aware that these operations have been done. The WCS protocol provides no explicit location for putting this information. In our demonstration of provenance chaining, we chose to embed the provenance information in a NetCDF/CF1 attribute (following an ISO 19115 type of structure). While not the most general solution, it has the virtue that the provenance stays with the data. This approach is described in more detail in RFC-15.

8 Scaling and Performance

As previously mentioned, we have two versions of WCS, one providing per-file based service without the on-demand mosaic capability and the other offering virtual coverages that are not tied to specific physical data files. For the per-file based service, the performance is not an issue regarding single coverage requests, except for rare cases when a client requests very high resolution output. (We have internally set certain requirements to prevent such a situation from happening.) On the other hand, time series studies can translate into a large number of requests during a short period of time (e.g., hundreds or thousands of simultaneous requests), in which case some requests may time out as the machine hosting the server slows down or runs out of memory or temporary disk space. (We have not yet conducted such scaling tests.) For the version that offers virtual coverage, the server is also at risk of performance problems because it

essentially offers unlimited spatial and temporal[1] extents. Due to the virtual coverage offering, scalability is strongly driven by client requests. Theoretically, the server should be able to provide a coverage that covers the entire spatial and temporal extents, e.g., a single global coverage from all the available years (two years in our CEOP use case) at nominal spatial and temporal resolutions (0.25 degree and one-day in CEOP use case). In actuality, our server does not support such requests. We allow only daily requests, primarily due to the number of files needed to access at the server side (240 files per day). Currently the temporal scale-up is provided through the Hyrax server where daily coverages for specific CEOP sites are staged. It is also possible to configure the WCS server to individual CEOP sites and eliminate the need to access all physical files. Our current design is to make the WCS general enough but make use of the scalability of the Hyrax server to meet user's long time series requirement.

Lesson Learned: The staging of daily coverages from the WCS server in the Hyrax server seems to be effective: it can provide users with long term time series for different CEOP sites, while insulating the WCS server from large requests that would seriously impair the WCS machine's performance.

9 Conclusions

It would be nice to think that a WCS server can be designed and implemented in a generic sense, without any consideration of the characteristics of the expected clients and users on the other side. Indeed, one of the key goals of standardizing protocols such as this is to abstract the user access from the server implementation. However, our experience seems to indicate that consideration must be given to those client and user factors in order for the WCS access to be useful to the target community. In addition to client/user factors, specific characteristics of the datasets to be supported can also exert a strong influence on the implementation requirements of the server. These characteristics are particularly exposed when offering long time periods of data to users that have a particular interest in the time dimension, as is the case with the CEOP users community. While this does not negate the usefulness of the WCS protocol in such cases, it does point to challenges in the server implementation. Though these challenges are solvable, they may nonetheless temper expectations for quick adoption of WCS within the Earth science research community unless user and client influences inform WCS server implementations.

10 Informative References

[1] Lynnes, C., Wang., W, Holloway, D., Enloe, Y and M. Min (2008). "Provenance within Data Interoperability Standards", ESDS-RFC-015.

11 Authors

The CEOP Satellite Data Server Project Team:

Wenli Yang, George Mason University, wyang1@gmu.edu

---

[1] The temporal extent is actually limited by a product's temporal availability in the server's archive.

Christopher Lynnes, NASA GSFC, chris.lynnes@nasa.gov

Daniel Holloway, OPeNDAP.org, d.holloway@opendap.org

Yonsook Enloe, SGT, yonsook@mindspring.com

Min Min, George Mason University, mmin1@gmu.edu

Appendix A - Glossary

ACCESS - Advancing Collaborative Connections for Earth System Science

AIRS – Atmospheric Infrared Sounder

BES – Back-end Server

CEOP – Coordinated Energy and Water Cycle Observations Project

CEOS – Committee on Earth Observation Satellites

CF1 – Climate and Forecast convention, version 1

CRS – Coordinate Reference System

CRST – Coordinate Reference System Transformation

DAP – Data Access Protocol

ESDS – Earth Science Data Systems

GES DISC – Goddard Earth Sciences Data and Information Services Center

GrADS – Gridded Analysis and Display Tool

MODIS – Moderate Resolution Imaging Spectroradiometer

NetCDF – Network Common Data Form

OGC – Open Geospatial Consortium

OLFS – OPeNDAP Lightweight Frontend Server

OPeNDAP – Open-source Project for a Network Data Access Protocol

QC – Quality Control

URL – Universal Reference Locator

WCS – Web Coverage Service

WGISS – Working Group on Information Systems and Services

XML – eXtensible Markup Language