Enhancements to the DODS-Web Map Server Gateway

D. Holloway, P. Cornillon, J. Gallagher
Data Access Software LLC
P.O.Box 6, Saunderstown, RI 02874, U.S.A
C. Lynnes, G. Serafino, P. Sweatman, R. Mullinix
NASA Goddard Space Flight Center
Code 902, Greenbelt, MD, 20771

Abstract: **The OpenGIS standard for web-based geographic information systems (GIS) has shown promise for implementing wide-area distributed GIS. However, making existing numerical data available as maps can be time-consuming and expensive, especially for large data archives. The map transformation process requires extraction of a subset from data files, interpolation, representation as a raster image and in some cases reprojection. We use the Distributed Oceanographic Data System (DODS) as an underlying subsetting engine with a "gateway" server providing map transformation services. This gateway server is designed to support plug-and-play modules for interpolation, rasterization and reprojection. In addition, the gateway, originally developed to the OpenGIS Web Map Server (WMS) 1.0 specification, is being revised to the WMS 1.1.0 specification. On completion, it will be included with the standard DODS distribution. Consequently, data providers who are already serving data using DODS will be able to add OpenGIS map capabilities cheaply and easily. The plug-and-play module capability supports both object-oriented C++ modules and standalone programs. These standalone programs are easy to develop and test independently of the gateway; this capability also makes it easier to add in commercial or third-party programs. A generic standalone program has been developed to display color-slice maps for scalar data, arrow plots for two-dimensional vector data, and RGB rendering for three-component data.**

I. Introduction

The DODS-WMT Gateway project grew out of fortuitous similarities in functionality and protocol between the Distributed Oceanographic Data System (DODS)[1] and the Web Map Server (WMS) of the OpenGIS Consortium (OGC)[2]. The DODS began as an attempt to provide oceanographic data stored in a variety of formats on distributed servers, while WMS is targeted at Geographic Information Systems (GIS) users. However, both systems standardize data transport so that users can use their favorite analysis clients; both support spatial subsetting at the server; and both use the Hypertext Transfer Protocol (HTTP) and overloaded Universal Resource Locators (URLs) for the request protocol. It thus seemed useful to adapt DODS servers to service WMS requests, allowing data centers to maintain enhanced DODS servers for both DODS and WMS requests. The ability to leverage the diverse formats and subsetting supported by DODS for WMS duty is of special interest to the Goddard Distributed Active Archive Center (DAAC), a component of NASA's Earth Observing System with many datasets already available through DODS.

II. Web Map Server Architecture

The DODS WMS gateway comprises two key functional components: service capabilities advertisement and map generation. The first of these is the ability to publish the operations that the gateway is capable of performing. In response to a "GetCapabilities" request, the gateway produces an XML response describing the data and operations available for that service instance. The format and content of the Capabilities XML document are defined by Document Type Definitions (DTD) specific to different revisions of the WMS Implementation Specification. (The current specification, revision 1.1.1, differs in several ways from the initial revision 1.0.0 but is largely similar to revision 1.1.0[3], the currently supported version of the DODS WMS gateway.)

The most complex task of a Web Map Server is the set of operations to transform data into maps in response to a "GetMap" request. The server accesses and extracts the data for the client's request using the sample dimensions provided by the client or the default values stored in the server's capabilities XML document. The server also interpolates and may reproject the layer to conform to the spatial reference system (SRS) requested by the client. Map stylization operations then create the requested map visualization, which is encoded into the client's requested output format and returned to the client.

III. DODS-WMS Gateway Architecture

The distinguishing feature of the DODS-WMS gateway is the use of the DODS Data Access Protocol (DAP) to provide access to distributed data sources in a variety of storage formats, using an intermediate data model. The DAP also supports subsetting, reducing transmission requirements for input data layers, and since the DAP provides remote access to the input data, the data need not reside with the gateway server. This enables the gateway to combine local and remote data to create new map

visualizations not possible with local data repositories alone, similar to an OGC cascading map server.

The DODS-WMS gateway provides OGC WMS compliant maps made from diverse sets of input data, using different map transformations depending on the OGC request. Rather than add each transformation to the gateway code, external "plug-and-play" specialization modules are supported for map transformations. The "plug-and-play" feature uses an XML-based configuration mechanism to provide at runtime the information needed by the gateway server to instantiate and interoperate with external modules. Thus, new modules can be added without changing the server code, allowing the addition of new map visualizations, spatial reference systems and output map formats with no service interruption.

Fig. 1 shows how the DODS-WMS Gateway components interact with other software in response to a WMS request. A WMS client sends a request over HTTP to the DODS-WMS Gateway server. The HTTP server passes the request via Common Gateway Interface (CGI) to the gateway software. The gateway acquires the data from a DODS server using the DODS Data Access Protocol (DAP) and then converts them into a mapped visualization matching the projection, bounds, image size and format requested by the client (Fig. 2).

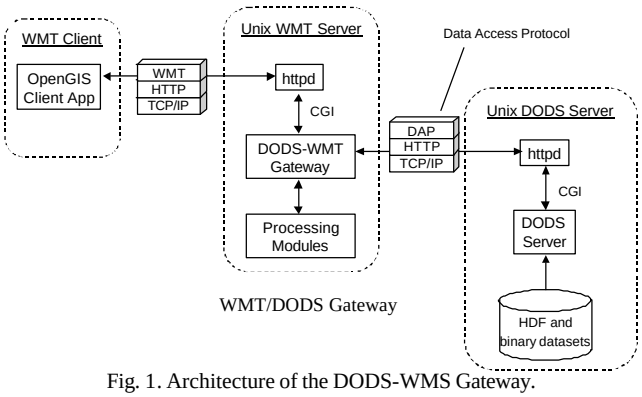

Fig. 1. Architecture of the DODS-WMS Gateway.

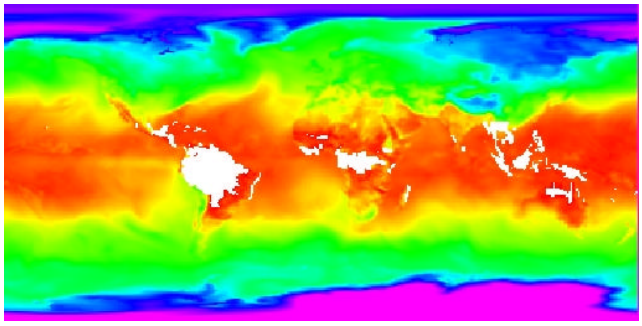

Fig. 2. Map visualization generated by DODS-WMS Gateway for global ground temperature, skin temperature of the surface (sea surface temperature over water) from the Data Assimilation System's 2-D late-look synoptic assimilation.

IV. Classes in the DODS-WMS Gateway

The DODS-WMS Gateway employs a number of C++ classes to satisfy a WMS request. These classes store and advertise the server's capabilities, provide access to and validate the client request parameters, and orchestrate the operations to generate the requested map. The rest of the gateway consists of the external "plug-and-play" specialization modules to support the spatial reference, visualization, and output format operations required for generating the requested map.

A. MapRequest Class

The MapRequest class encapsulates the behavior to store and access the client request for use by the gateway server. The class is also tasked with validating the client's request against the server's advertised capabilities, throwing an exception for malformed requests. The gateway server catches any MapRequest exception, constructs a valid exception response and returns it to the client application. The MapRequest class also identifies any use of "default" values for sample dimensions, as well as vendor specific parameters (VSP), maintaining that information for use by the classes performing the map transformations. Since the MapRequest class provides a discrete interface between the gateway server and the request syntax, changes in the WMS request parameters revision or usage are isolated to the MapRequest class, facilitating support for future WMS revisions. Since the MapRequest class provides exclusive access to the client request parameters, the gateway server passes a MapRequest instance to the classes responsible for generating the map transformations. This standardizes the parameters passed between various modules instantiated by the gateway server, and isolates all access to the map request parameters to a single class instance.

B. Capabilities Class

The Capabilities class encapsulates the behavior to advertise the server's map layers and operations that the service is capable of performing on them. The server's capability information is stored in an external XML document conforming to the WMS revision 1.1.0 Capabilities DTD. Using standard XML parsers, the Capabilities class transforms the external XML document into an internal Document Object Model (DOM) representation. The class provides accessor methods that traverse the DOM to retrieve any desired element or attribute, from the server's capability XML representation.

The MapRequest uses the Capabilities class to validate client requests against advertised capabilities, and to identify the default values for sample dimensions not specified in client requests. It also provides a discrete interface between the gateway server and the WMS Capabilities XML Specification, so that future revisions of the Capabilities XML specification are limited to one

class. Since the Capabilities class provides sole access to the server's capability information, the gateway server passes a reference to a Capabilities class instance to the classes responsible for generating the map transformations.

As external "plug-and-play" modules were implemented, the initial configuration XML began to resemble elements of the existing capabilities XML representation. Rather than duplicate the XML representation and traversal methods of the Capabilities class, the local configuration information was merged into the gateway server's capabilities XML document and the Capabilities class was extended with accessor methods to the configuration elements. This reduces the installation and maintenance burden on a server's administrator to a single XML document.

C. Plugin and PluginFactory

The external "plug-and-play" implementation comprises three elements: C++ wrapper classes to provide a common interface to the external specialization modules; a mechanism to load those classes at runtime; and local configuration information defining which external modules are required to satisfy the client's request.

The Plugin and PluginFactory classes allow the gateway server to load C++ classes at runtime. For the "plug-and-play" mechanism to operate, the C++ wrapper class uses virtual constructors and a method named "maker" that returns a new instance of the class. The class definition for the external "plug-and-play" module is compiled and stored in a shareable object library. Using configuration information from the server's capabilities XML, the PluginFactory class selects the correct shareable object library to load an instance of the specialization module at runtime. The Plugin class provides the interface to the specialization classes instantiated by the PluginFactory. Fig. 3 shows a Unified Modeling Language (UML) class diagram for the Plugin and PluginFactory classes and an example C++ wrapper class for a DEG specialization.

D. Specialization Classes

To satisfy a client's map request, the gateway server performs several discrete operations on the input data layer to create the requested map. These include SRS operations that extract the requested spatial subset from the data layer, interpolate and (sometimes) reproject to conform to the client's requested SRS. The resulting data are then styled into a visual representation and encoded into the requested output map format.

The external "plug-and-play" mechanism to support these operations relies on external specialization modules wrapped by C++ classes used by the gateway server to instantiate the specialization at runtime. The well-defined interface provided by the wrapper classes aids interoperability by standardizing arguments and argument passing mechanisms, while allowing the external specializations to be highly customized.

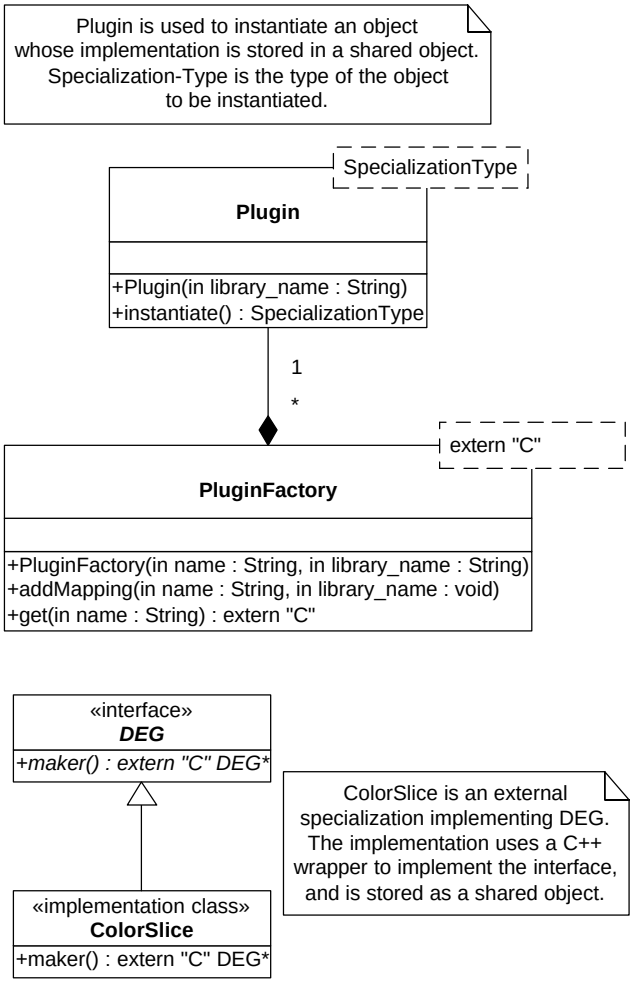

Fig. 3. Class diagram for "plug-and-play" mechanism.

The external specialization modules can be C++ modules or standalone programs. For standalone programs, the C++ wrapper class provides the interface between the gateway server classes and the external executable. The wrapper provides the input data to and executes the program, returning the processed data to the gateway server. So that a single C++ wrapper class can support many similar standalone programs or provide different arguments to the same standalone program, an optional CDATA element is included in the configuration XML for the specialization elements. The free-text nature of the CDATA elements in the configuration XML supports command-line arguments or other textual information needed to execute the standalone program. Since the specialization elements in the configuration XML are coupled to data layer names, parameters to the external standalone programs can be uniquely defined for each layer advertised by the gateway. Optionally, because

the gateway server supports layer inheritance as defined in the WMS Capabilities XML specification, specializations can be defined for a parent layer node to be inherited by its children unless explicitly overridden.

The intent of the external "plug-and-play" standalone modules and the configuration XML is to facilitate adding new standalone executables without requiring new C++ wrappers to be customized each time. Also, support for external standalone programs allows nonprogrammers to incorporate higher-level languages (e.g. IDL, Matlab), as well as third-party programs that may not come with source code or libraries, to provide the desired specializations for generating maps.

1) SRS (Spatial Reference System) The SRS specialization modules ensure that the generated map transformations conform to the client's requested spatial reference system. Since the client's requested SRS may differ from the "native" SRS of the input data layer, the specialization may need to perform interpolation and reprojection operations on the input data layer. To enable the use of different interpolation and reprojection techniques, these operations may be encoded as external "plug-and-play" modules as well.

An operation performed by all SRS specializations is to transform the client's requested spatial bounding box (BBOX) and other sample dimension request parameters into a form consistent for accessing the advertised data layer. For input data layers accessed using the DODS DAP, the result of this operation is a fully qualified DODS URL, or potentially a set of URLs for client requests that access multiple files. While the gateway facilitates access to data by using the DODS DAP, the server design does not require the DAP to be the only access method for the input data layers. To support DODS access, a Layer class encapsulates the access methodology required to interact with the DAP class libraries, and is used by the SRS specializations for DODS-accessible data layers.

2) CatalogURL The majority of DODS-accessible datasets that could be served using the DODS-WMS Gateway consist of multi-file data archives. The CatalogURL specialization provides a customizable mechanism to translate a WMS client's sample dimension request parameters into a DODS inventory request. The CatalogURL specialization returns a set of base DODS URLs representing the granules associated with the client's request parameters.

3) DEG (Display Element Generator) The DEG specialization modules create the client's requested map style, or visualization, using the output of the SRS specialization modules. A generic DEG specialization module has been developed to display color-slice maps for scalar data, using an external palette file. Future plans include contour plots for scalar data, arrow plots for two-dimensional vector data and RGB rendering for three-component data. Since the DEG specializations can be implemented as standalone external programs, they can be developed and tested independently of the gateway. This also makes it possible to use commercial or third-party programs for the map visualizations.

4) Format The Format specialization modules encode the styled map visualizations produced by the DEG specializations into the client's requested output response format, *e.g.* JPEG. They support the WMS feature allowing a client to request multiple map layers in one request, with a single output response containing all requested layers. The server maintains the order of the requested layers using the leftmost layer request parameter as the bottommost map in the output format response.

V. Orchestrating The Map Transformation

While the specific operations required to perform map transformations on the input data layers may be complex, managing the process is straightforward. To satisfy a WMS client request, the gateway server's main class creates a MapRequest and Capabilities class instance. The server initially determines the client's requested service and request type.

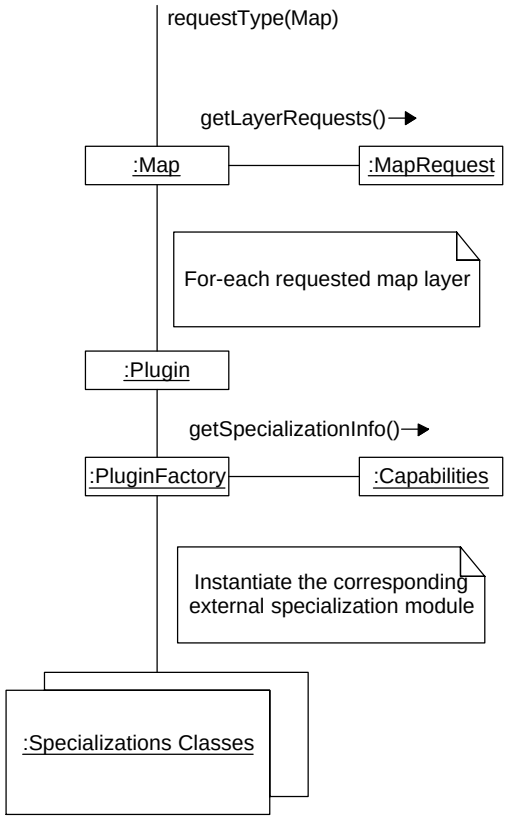

Fig. 4. Collaboration Diagram for Map Class

Fig. 4 shows the collaboration between gateway classes to satisfy a "GetMap" request. The gateway server passes references to MapRequest and Capabilities class instances to the Map class. The Map class uses MapRequest and Capabilities information to determine which "plug-and-play" specialization modules will satisfy the client's map request. Then, for each requested layer/style pair, external "plug-and-play" specialization modules are instantiated by the Plugin and PluginFactory classes. Each instantiated specialization module is instructed to perform its operation and the resulting map visualizations are passed to a "plug-and-play" Format specialization. On completion, the gateway server returns the formatted output response to the client application and exits.

With slight modification to the MapRequest and Capabilities classes, additional "plug-and-play" Format specializations and a Coverage class to mimic the behavior of the Map class, the gateway server can be easily extended to support OGC Web Coverage Service requests. Also, combining "plug-and-play" visualization modules with a Coverage service can extend the gateway server to support OGC Coverage Portrayal Service requests.

VI. Conclusion

The combination of "plug-and-play" specialization modules, external XML configuration information, and the DODS Data Access Protocol provides a general, cost-effective and flexible approach to facilitate serving a wide range of earth science data to OpenGIS client applications. External, standalone "plug-and-play" specialization modules make it easy to develop and test new visualization and format operations, or make use of commercial and third-party applications. Over time, a library of specialization modules should accumulate so that eventually, a dataset can be added simply by modifying a configuration file. Integrating the server's configuration XML with its capabilities XML allows the gateway server to be highly customizable. Use of the DODS DAP facilitates access to data by reducing the format handling requirements for gateway server, and provides a subsetting capability for the input data layers. Also, the DAP enables gateways that are not co-located with the data underlying the advertised maps.

The enhancements to the DODS-WMS Gateway for compliance with revision 1.1.0 of the WMS Implementation Specification provide a number of benefits. These include enhanced exception handling and informational messages to send back to the requesting client application, additional support for refining the layer sample dimension specifications with clearly defined default value usage, and better support for customization of SRS operations for individual data layers. Additionally, as the other OGC web service specifications mature, the enhancements to support the map request and service capabilities provide a foundation to support OGC Web Coverage, and Coverage Portrayal Services.

The greatest benefit resulting from the enhancement effort is the new support for external "plug-and-play" specialization modules. This will facilitate adding new operations to the gateway server, potentially by nonprogrammers, for generating new map transformations or simply serving additional data products. These new operations can be added to an operational gateway server with minimal interruption in service, and do not require modifications to the gateway server software or the software to be rebuilt to support these new operations.

As part of the DODS-WMS Gateway enhancement effort, the gateway server is being packaged for distribution as part of the suite of applications freely available from the DODS project web site. This includes technical documentation describing the design and operation of the gateway server, as well as a guide to developing new external "plug-and-play" specialization modules. Documentation is also provided for installation of the gateway server, the combined configuration and capabilities XML document, and describing the example "plug-and-play" specialization modules. The software itself will be freely available for download in both binary and source distributions from the DODS project web site.

Acknowledgements

We thank the Earth Science Technology Office at NASA for funding the development of the DODS-WMS Gateway.

References

[1] J. Gallagher and G. Milkowski, 1995. Data Transport Within The Distributed Oceanographic Data System, Fourth International World Wide Web Conference, December 11-14, Boston, Massachusetts, USA, http://www.w3.org/Conferences/WWW4/Papers/67/

[2] OpenGIS Consortium, 2000. "OpenGIS® Web Map Service Interface Implementation Specification, Version 1.0.0", http://www.opengis.org/techno/specs/00-028.pdf.

[3] OpenGIS Consortium, 2001. "OpenGIS® Web Map Service Interface Implementation Specification, Version 1.1.0", http://www.opengis.org/techno/specs/01-047r2.pdf.