GIS @ CGL

SERVO - ACES Abstract

Implementing GIS Grid Services for the International Solid Earth Research Virtual Observatory

- Galip Aydin(1), Marlon Pierce(1), Geoffrey Fox(1), Mehmet Aktas(1) and Ahmet Sayar(1)
- (1) Community Grids Lab, Indiana University, Bloomington, Indiana, 47404, USA (e-mail: { marpierc, gcf, maktas, asayar}@indiana.edu phone: +1 812-856-1212).

Abstract

- **We describe our continuing work on implementing Open GIS Consortium (OGC) compatible Grid Services for the International Solid Earth Research Virtual Observatory. Our initial efforts focused on collecting Earthquake and GPS data from various sources, converting them to GML and enabling query capabilities using pure Web Services approach. We have extended this work by creating Web Services implementations of OGC-Web Feature Service and OGC-Web Map Service for serving GML-formatted data. We also describe a Fault Tolerant High Performance Information System (FTHPIS) to tie these services together.**

Introduction

Advances in Geographical Information Systems (GIS) introduce several challenges for acquiring, processing and sharing geo-data among interested parties. Different organizations and commercial vendors develop their own data models and storage structures. Consequently the data is expressed in various formats and stored on various databases. This makes it hard to achieve for the scientific applications or clients to easily recognize and process data acquired from other sources. On the other hand the nature of the geographical applications requires seamless integration of spatial data from a range of providers to produce layers, maps etc. As a result we see the interoperability between applications and data stores as a significant goal for any GIS.

As an example of how this goal can be accomplished we describe our design of a Service Oriented Architecture for serving a subset of geographical information. We first review the proprietary data formats in our domain of interest and summarize our initial work for generating a common data format. The next section explains how we employed pure Web Services approach for data conversion, storage and query capabilities. The next section gives a brief discussion about our experience and findings on XML and Relational Databases, and the user interfaces we created for testing the Web Services.

The rest of the paper presents our current involvement in Open GIS Consortium (OGC) Web Services implementations and the design of an Information System for GIS.

A Service-Oriented Architecture

The geographical information we used in this project is investigated in two major categories: GPS time series and Earthquake catalogs. GPS data includes daily Geodetic and Cartesian positions and velocities and their estimates from an array of GPS stations. Earthquake data consists of seismic event catalogs and detailed information about fault lines.

In our Solid Earth Virtual Observatory project, consumers of these data range from humans to scientific applications like RDAHMM, GeoFEST, Simplex and Disloc (1). Providing data to these consumers requires access to widely distributed and completely independent data stores. Many of the datasets are ASCII formatted text or tar files. Some are available through public FTP servers while others are provided over HTTP. (2)

Goals of the Project

We designed a service-based architecture for solving the aforementioned challenges. However before implementing this system we identified several goals to make the scope of this project clear. These goals are as follows:

- Making GPS and Seismic data easily available for humans and applications alike

- Providing seamless access to data repositories and computing resources
- Providing a common data format for each information area
- Supporting search capabilities on the catalogs for certain properties, filtering the search results, and retrieving the results in various formats
- Integrating data with the scientific applications

Data Format Issues and GML

Depending on the data provider's use the catalogs are formatted in various ways. Any application or user who wants to make use of these legacy formats should understand what each column or data segment means and should write scripts to convert them into the target application's own legacy format. This introduces the tedious and resource consuming conversion problem. Also extending any scientific application's scope with integrating new formats would not be easy for the same reason.

Therefore this was the initial challenge for us with making the geographic data easily available: *data in this domain comes from different sources in different formats*. So we decided to build a common data format which encapsulates all legacy formats.

Our choice for the common data format was GML - Geography Markup Language (3) for its strong Open GIS Consortium (4) support and for its being widely accepted as a common exchange format for spatial information in GIS community worldwide. GML provides support for the geometry elements corresponding to the Point, LineString, LinearRing, Polygon, MultiPoint, MultiLineString, MultiPolygon and

GeometryCollection.

- Geography Markup Language is an XML grammar written in XML Schema for the modeling, transport, and storage of geographic information.
- GML provides a variety of kinds of objects for describing geography including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values. (5)

We put together a list of the legacy formats used in our domain of interest and designed GML Schemas that unify them all. Our GML Schemas support following formats:

- Seismic data formats:
  - SCSN, SCEDC, Dinger-Shearer, Haukkson
- GPS data formats:
  - JPL, SOPAC, USGS

We also wrote a GML schema for Faults (available in both GML2 and GML3).

The next step after creating a common exchange format for the catalogs was to design a service based system to process these data:

The first thing needed to be done in the system was to collect data from various sources. GPS and seismicity catalogs are provided by several organizations and they are available to the public via FTP or HTTP servers. We wrote clients for retrieving the catalogs and saving them in our server for further processing. These retrieval applications are also provided as Web Services so that any workflow scheme can schedule regular catalog updates.

Our system allows user to select one of the two paths to follow after acquiring the catalogs:

- Convert the catalogs into GML format and save them as XML files. These files are then inserted into an XML database.
- Insert the catalog data into relational database.

The choice in this step is introduced because of the use of both relational and XML databases in the system. We extensively tested and used both types of the databases.

Databases

Since the message exchange format of Web Services is XML and since we convert our data to XML at the beginning of the process we thought that using native XML databases would simplify the design. We wrote tools for both Berkeley DB XML (Sleepycat) (6) and Apache Xindice (7) to insert/update/search our GML data. But after testing these extensively we found out that XML databases perform poorly over large XML documents. We then decided to use a relational database (MySQL (8)) which demonstrates acceptable query time for the complex queries.

Searching the Catalogs

We provide a Web Service for searching the catalog databases. The results are sent to the client application as GML documents. Since there is only one GML Schema for a particular data group (GPS or Seismicity) it is easy to validate and parse the received GML document and extract the information. XPATH (9) is used as the query language for the XML Databases.

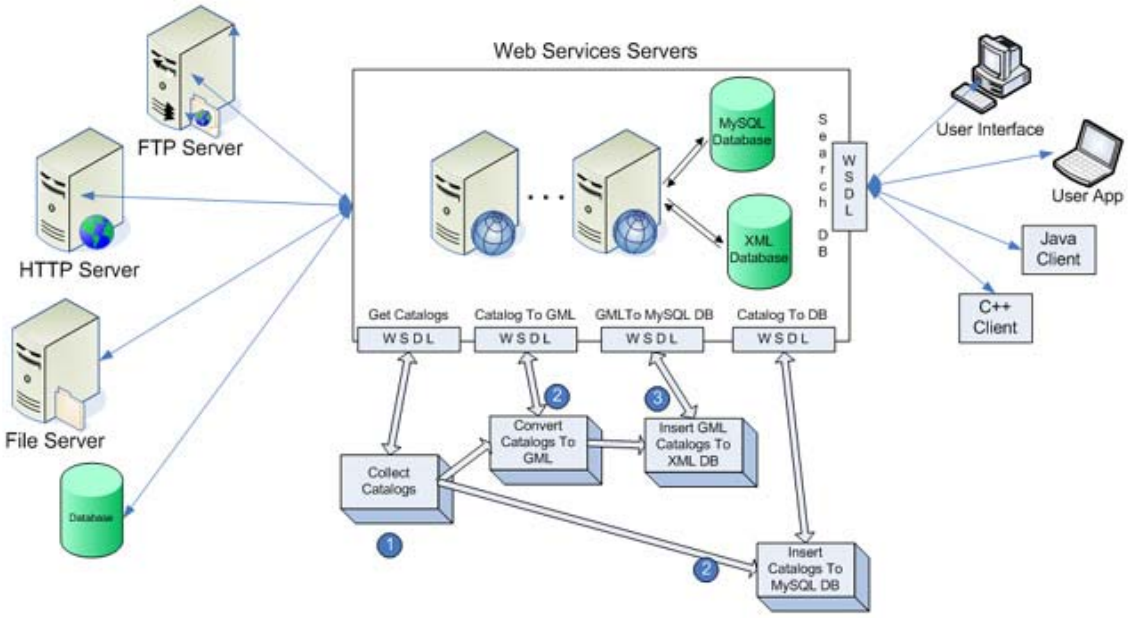

Figure 1: Major parts of the architecture and a sample workflow for processing geo-data using Web Services.

User Interfaces

We have created a set of user interfaces to the Web Services for demonstration purposes. These interfaces also show the steps for making a set of geographic data available to search and retrieve via web services. The search client enables users to create database queries via web page forms. (10) The search results are shown to the user as a simple text page. We use XML Pull Parser (11) to quickly extract data from received GML file and create the web page output.

Clients

Our architecture provides several WSDL interfaces to enable both human and application clients to use the services. WSDL does not require the client to be bound to a particular programming language or environment and this gives the users the freedom to use any type of client tools such as Java classes or C++ applications. To demonstrate this feature we have created a simple C++ client using GSOAP (12) along with our JSP interfaces and Java clients.

OGC Web Services

In addition to the development of GML, OGC defined several web service specifications for exchanging

geographical information. Initial versions of these are not fully compatible with WS standards. They are based on servlets that use HTTP Post and Get. We made several additions to OGC standards to create OGC Services with Web Service standards.

One of the major services OGC defines is *Web Feature Service* (13) that describes standards to publish, update, and delete geographic Features. A GML feature is the abstraction of a real world phenomenon.

We designed a Web Service version of OGC WFS that provides WSDL interfaces for the required capabilities. Instead of using HTTP Post, the user or the client application communicates with the WFS via standard WS clients. The requests should be valid WFS requests. The results of the requests are sent to the user as GML documents.

One important property of the WFS is that it can serve multiple feature types. Different features from different data stores are integrated with the WFS and the clients do not realize that the features are retrieved from several sources.

We also implement a Web Service version of Web Map Service to generate maps (14).

The Web Map Service gets data from various Web Feature Services. The interaction between these two services is based on SOAP messaging because of the Web Service standards.

GIS Information Services

As the number of available Web Feature Services increased, an immerging need for GIS Information Services appeared to support the discovery and handling these geospatial services. To this end, we design a Fault Tolerant High Performance Information System (FTHPIS) for GIS Services in particular for Web Feature Services.

In a FTHPIS, there is a need for registry services to make the information about services available. We use the Universal Description, Discovery, and Integration (UDDI) (15) specifications in our design as centralized registry. UDDI offers users a unified and systematic way to find service providers through a centralized registry of services. We design an extension to existing UDDI Specifications in order to provide dynamically updated service registry data.

UDDI provides a centralized approach to Web Services registry research problem. However, one should be able to locate a Web Services satisfying a query, in a decentralized distributed environment as well. To do this, discovery architecture needs to be defined. Discovery architecture defines appropriate services, interfaces, operations and protocol bindings for discovery. As we consider the volatile behavior of Web Services, such decentralized approach should provide dynamic discovery of services where the temporarily connected Web Service can be discovered. To this end, we design our implementation based on WS-Discovery Specifications which was recently released.

WS-Discovery defines a multicast protocol to locate services. It allows dynamic discovery of services in ad hoc and managed networks. However, WS-Discovery does not define a metadata model to describe Web Service. To this end, we also define an abstract metadata model for Web Service. This will allow us to pose more complex queries for WS-Discovery.

Metadata associated with Web Services can be separated as dynamic metadata and static metadata. Dynamic metadata is the session (or state) metadata generated by the individual interactions with Web Services. Such metadata describes the context of the session and has a lifetime. There are different approaches specifying session metadata. For instance, WS-GAF uses WS-Context to provide abstract context defining such metadata. Static metadata is the metadata describing WS profile such as its usage cost, availability, bandwidth, computing power, storage capability and etc…

We adapt Resource Description Framework (RDF) syntax to express WS metadata. RDF provides an important framework for metadata. By using RDF, we are able to construct complex statements describing information about Web Services. Also, RDF provides unique properties defined by URIs describing resources. One can use RDF statements to create complex associations between metadata of different resources.

Acknowledgments

This work was funded by the Computational Technologies Program and the Advanced Information Systems Technology Program, both of NASA's Earth Science Technology Office. We gratefully acknowledge Ms. Michele Judd for project management.

References

- QuakeSim Project Home Page: http://quakesim.jpl.nasa.gov/.
  For project documentation, see http://quakesim.jpl.nasa.gov/milestones.html.
- More information about the supported data formats and GML schemas is available on:
  http://grids.ucs.indiana.edu/users/gaydin/servo
- Geography Markup Language. GML 3.0 specification:
  http://www.opengeospatial.org/docs/02-023r4.pdf
- Open GIS Consortium – (Also known as Open Geospatial Consortium) .
  Web Page:http://www.opengeospatial.org/
- GML 3.0 Specification, 27.
- Berkeley DB XML product page: http://www.sleepycat.com/
- Apache Xindice project page: http://xml.apache.org/xindice/
- MySQL product page : http://www.mysql.com/
- W3C XML Path Language XPATH: Version 1.0 specification
  http://www.w3.org/TR/xpath
- For the sample user interfaces see:
  http://gf3.ucs.indiana.edu:6060/cce/sql
- XML Pull Parser project page: http://www.extreme.indiana.edu/xgws/xsoap/xpp/
- GSOAP-SOAP C/C++ services and clients. Project page:
  http://www.cs.fsu.edu/~engelen/soap.html
- Web Feature Service 1.0 specification: http://www.opengis.org/docs/02-058.pdf
- Web Map Service 1.3 specification: http://www.opengeospatial.org/specs/
- UDDI - The Universal Description, Discovery and Integration protocol web page:
  http://www.uddi.org