Implementing Geographical Information System Grid Services to Support Computational Geophysics in a Service-Oriented Environment

Mehmet Aktas[(1)], Galip Aydin[(1),*],Andrea Donnellan[(2)], Geoffrey Fox[(3)], Robert Granat [(4)], Greg Lyzenga[(5)], Dennis McLeod[(6)], Shrideep Pallickara[(1)], Jay Parker[(7)], Marlon Pierce[(8),*], John Rundle[(9)], and Ahmet Sayar[(1)]

(1) Community Grids Lab, Indiana University, Bloomington, IN 47404-3730 (2) NASA Jet Propulsion Laboratory, Mail Stop 183-335, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, USA (email: Donnellan@jpl.nasa.gov). (3) Community Grids Laboratory, Departments of Computer Science, Physics, and School of Informatics, Indiana University, Bloomington, Indiana 47404-3730, USA (email: gcf@indiana.edu; phone +1 812-856-7977). (4) NASA JPL, Mail Stop 126-347, 4800 Oak Grove Drive, Pasadena, CA 91109-8099 (email: Robert.Granat@jpl.nasa.gov). (5) NASA JPL, Mail Stop 126-347, 4800 Oak Grove Drive, Pasadena, CA 91109-8099 (email: Gregory.Lyzenga@jpl.nasa.gov). (6) University of Southern Califoria, Mail Code 0781, 3651 Trousdale Parkway, Los Angeles, CA 90089-0742, USA (email: mcleod@pollux.usc.edu). (7) NASA Jet Propulsion Laboratory, Mail Stop 238-600, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, USA (email: jay.w.parker@jpl.nasa.gov). (8) Community Grids Laboratory, Indiana University, Bloomington, Indiana 47404-3730, USA (email: mpierce@cs.indiana.edu, phone: +1 812-856-1212). (9) Department of Physics, University of California-Davis, One Shields Avenue, Davis, CA 95616-8677 USA (email: rundle@physics.ucdavis.edu).

* Corresponding author

Abstract

We describe the architecture and implementation of the Solid Earth Research Virtual Observatory (SERVO)'s Complexity Computational Environment. We base our design on a globally scalable distributed "cyber-infrastructure," or Grid, built around a Web Services-based approach consistent with the extended Web Service Interoperability (WS-I+) model. In order to investigate problems in earthquake modeling and forecasting, we need to programmatically couple numerical simulation codes and data assimilation and mining tools to online observational data sets, including GPS stations, fault data, and seismic activity catalogs. These observational data sets are now available on-line in internet-accessible forms, and the quantity of this data is expected to grow explosively over the next decade. As part of our efforts in building SERVO, we are extending these online data repository capabilities so that they are not just available directly for human users, but may also be searched, filtered, and streamed to simulation codes that are also managed by SERVO services. SERVOGrid is

implemented using Service Oriented Architecture (SOA) principles. We use WSDL for service description and SOAP for message formats, and are implementing several supplemental Web Service specifications to address problems in information discovery, reliable messaging, and security. This framework allows us to build numerous services for executing and managing remote applications, moving data, and similar operations. An important sub-family of services that we are implementing is based on standards provided by the Open Geographical Information Systems (GIS) Consortium (OGC). Based on common XML data models (the Geographic Markup Language and the SensorML collection), these services provide access to abstract geographic features (GPS stations, faults, locations of seismic events), interactive map-making capabilities for user interaction and analysis, and access to streaming data. These data services are directly coupled to CCE applications, including data assimilation codes for creating earthquake hazard maps and fault-modeling applications.

Introduction

In this paper we describe the architecture and initial implementation of the International Solid Earth Research Virtual Observatory (iSERVO) [1]. We base our design on a globally scalable distributed computing infrastructure (often termed "cyber-infrastructure" or "Grid infrastructure" [2][3]) that enables on-line data repositories, modeling and simulation codes, data mining tools, and visualization applications to be combined into a single cooperating system. We build this infrastructure around Web Services-based approach. This report describes our efforts to couple science application codes to data sources using appropriate community standards.

Challenges for Solid Earth Research

The Solid Earth Science Working Group of the United States National Aeronautics and Space Administration (NASA) has identified several challenges for Earth Science research [4]. Particularly relevant for iSERVO are the following:

- How can the study of strongly correlated solid earth systems be enabled by space-based data sets?
- What can numerical simulations reveal about the physical processes that characterize these systems?
- How do the interactions in these systems lead to space-time correlations and patterns?
- What are the important feedback loops that mode-lock the system behavior?
- How do processes on a multiplicity of different scales interact to produce the emergent structures that are observed?
- Do the correlations allow for the capability to forecast the system behavior?

In order to investigate these questions, we need to couple numerical simulation codes and data mining tools to observational data sets. This observational data (including crustal fault data from the literature, GPS data, and seismic activity data) are now available on-line in internet-accessible forms, and the quantity of this data is expected to grow explosively over the next decade.

The challenges in solid earth modeling motivate a number of interesting research and development issues in distributed computer science and informatics. Key among these are providing programmatic access to distributed data sources; coupling remote data sources to application codes, including automated searching and filtering; coupling of complementary application codes that are deployed on geographically separated host computers; and providing human level interfaces to these remote services.

The iSERVO team possesses a broad range of skills and tools that may be used to investigate solid earth research challenges. Team expertise includes the development high performance modeling and simulation applications for both the study of large, interacting earthquake systems and the detailed study of individual fault properties; federated database and ontology design; geological characterization of faults; and high performance visualization codes. Welding all of these components into a common distributed computing infrastructure is the subject for the rest of this paper.

A Web Service Grid Architecture

Problems in managing distributed computing resources, applications, and data have been studied for many years (see [2], [3]). Typical desired functionality in these systems includes remote command execution, data transfer, security, and high performance messaging. To scale globally, these systems must abandon tight coupling approaches such as distributed object systems and micro-second latency solutions such as MPI and adopt instead a Service Oriented Architecture (SOA) [5] that is compatible with millisecond (or longer) communication speeds. SOAs are implemented around two basic components: service definition languages (which describe how to invoke the remote service) and message formats for over-the-wire transmissions. In iSERVO, we have adopted the Web Service approach to building an SOA: we use WSDL (http://www.w3c.org/TR/wsdl) for service description and SOAP (http://www.w3.org/TR/soap/) for message formats.

Web Service systems have an important design feature: services are decoupled from the user interface components. This enables us to build (in principal) a number of different services that can interact with the same remote service. Browser-based computing portals are typical of this sort of user interface and have been the subject of research and development work for a number of years [6]. Currently this field is undergoing a revolution as component-based portal systems are being widely adopted, and standard component programming interfaces have been released (for details, see http://jcp.org/aboutJava/communityprocess/final/jsr168/index.html). This approach so-called “portlet” enables reusability of components: portals may be built out of standard parts that aggregate content and functionality from many different sources.

SOA and portal standards are not the only relevant standards for building systems such as iSERVO. The Open Geographical Information Systems (GIS) Consortium (OGC) (http://www.opengis.org) defines a number of standards for modeling earth surface feature data and services for interacting with this data. The data models are expressed in the XML-based Geography Markup Language (GML), and the OGC service framework is being adapted to use the Web Service model.

Implementing iSERVO

We have implemented an initial set of services and portal components for addressing the problems described in the introduction. We have followed a Web Service-based Grid design described above that uses Web Service standards. The components of the system and their interactions are summarized in Figure 1. Users interact with remote services through a Web browser portal that is run by the User Interface Server (UIS). This portal generates dynamic web pages that collect input information from the user and deliver response messages. The UIS does not directly implement services such as job submission and file transfer. Instead, it maintains client proxies to these remote services. These proxies are responsible for generating the SOAP messages appropriate to the particular services' WSDL descriptions and for receiving the responses from the services. The UIS and most services are implemented in Java using the Apache Axis toolkit (http://ws.apache.org/axis/), but we have also implemented C++ services using gSOAP (http://www.cs.fsu.edu/~engelen/soap.html) for simple remote visualization.

A typical interaction involves the user selecting a code through the portal, setting up an input file in part through interactions with databases (such as the QuakeTables Fault Database[7]), invoking the code and monitoring its progress, and having the output visualized through various third party tools of varying sophistication. These interactions are based on a dataflow model: services communicate by exchanging data files, which must be pulled from one server to another.

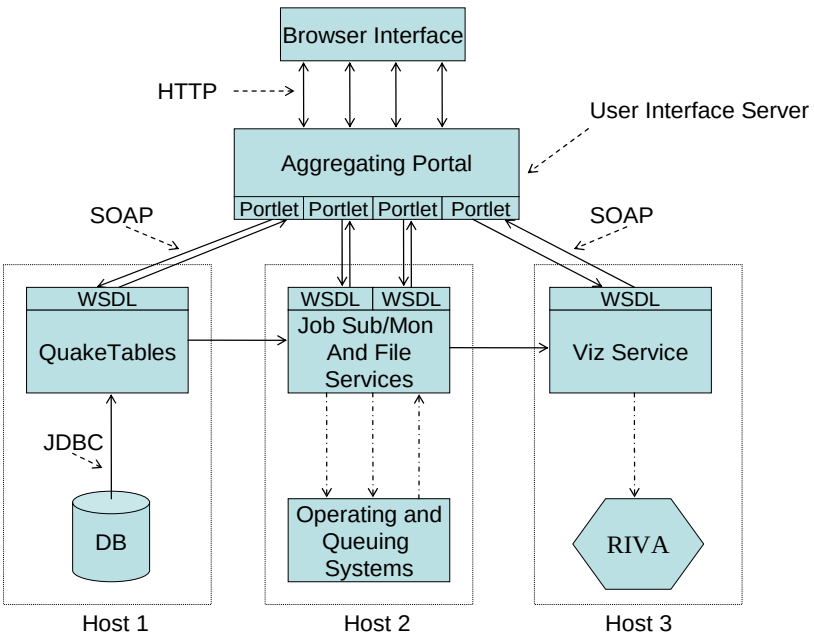

Figure 1 The architecture for the iSERO portal and services uses Web Service and portal standards.

In building iSERVO, we have implemented a number of innovations on the standard model components. The portlet component model normally assumes local portlets with content that navigates to other web sites (news portals such as Yahoo and CNN are

examples). We have built extensions to this simple model to allow portlet content to be managed remotely, have its display maintained within its component window through a series of navigations, maintain HTTP sessions state with remote content, pass HTTP GET and POST variables, and support SSL security.

Basic iSERVO services include remote command execution, file upload and download, and host-to-host file transfer. We do not directly alter the geophysical applications included in the portal but instead follow a "proxy wrapping" approach [8]. Typically, applications require preprocessing of input files, post processing, and in general require task executions that are distributed across many different hosts. To support this sort of distributed service orchestration, we have developed a simple "workflow" service based on the Apache Ant project (http://ant.apache.org/). This service uses Ant as an engine that may be invoked remotely (as a service on Host 2 in Figure 1) and may also coordinate service invocations on remote hosts, as needed to complete its task.

iSERVO couples typical "Execution Grid" services such as described above with "Data Grid" services. iSERVO applications work with many different data sources, and we have developed services to automate the coupling of this data to application services. A typical problem is as follows: the iSERVO application RDAHMM (a Hidden Markov Model application) needs as input either GPS or seismic activity records. Both data sources are available online, but there is no programmatic way of working with this data. Instead, it is typically downloaded and edited by hand. To solve this problem, we have implemented GML-based services for describing these data records, and in the process we have unified several different data formats. These services allow the application user to build search filters on the desired data set (for example, returning events larger than magnitude 5.0 on a particular region of interest since 1990). Additional filters reformat the data into one suitable for RDAHMM, and the data is then shipped to the location of the remote executable.

Geographical Information System (GIS) Data Services

iSERVO data service requirements represent an excellent opportunity for further work leveraging open standards for services that will tie iSERVO to this larger community, allowing us to potentially incorporate many additional third party data sources and tools. The NASA OnEarth project (http://onearth.jpl.nasa.gov/) is an excellent example of a GIS project that may be incorporated with iSERVO in the future. As part of our GIS development work, we are currently re-implementing the OGC standard services Web Feature Service and Web Map Service as iSERVO-compatible Web Services.

We note that the GIS community has other data model and service standards than those defined by the OGC: The commercial vendor ESRI provides another prominent set of data standards along with extensive client tools. Our adoption of OGC standards is intended to take advantage of the significant amount of freely available GIS data that already exists in OGC formats. More importantly, OGC standards define an open architecture that may be integrate with Grid/Web service standards for distributed scientific computing discussed in the previous sections. We note further that ESRI and OGC interoperability tools already exist for obvious reasons, so adopting OGC standards does not preclude later integration of our data services with sophisticated ESRI software clients.

Advances in Geographical Information Systems (GIS) introduce several challenges for acquiring, processing and sharing data among interested parties. Different research groups, organizations, and commercial vendors develop their own data models and storage structures. Consequently the data is expressed in various formats and stored in various archives. These archives are often remotely accessible only through simple protocols (like FTP) that do not allow queries and filtering and which are difficult to integrate with geophysical applications. 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 existing 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.

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; and
- Integrating data with the scientific applications.

Figure 2 illustrates the major components of the system for achieving these goals. Existing public archives maintained by the Southern California Earthquake Center (SCEC) and the Southern California Integrated GPS Network (SCIGN) are accessed through Web Services that download and reformat the data into GML (steps 1 and 2 in Figure 2). Data sources that we relied upon are more extensively documented at www.crisisgrid.org. We then store the converted data in either native XML or relational databases (step 3).

The above steps summarize administrative services that need to be performed once per external archive for initialization, followed by regular updates. Application users do not need to use these services. They do, however, make use of the search services (right hand side of Figure 2). These are also Web Services defined in WSDL and so may be accessed by various client programs.

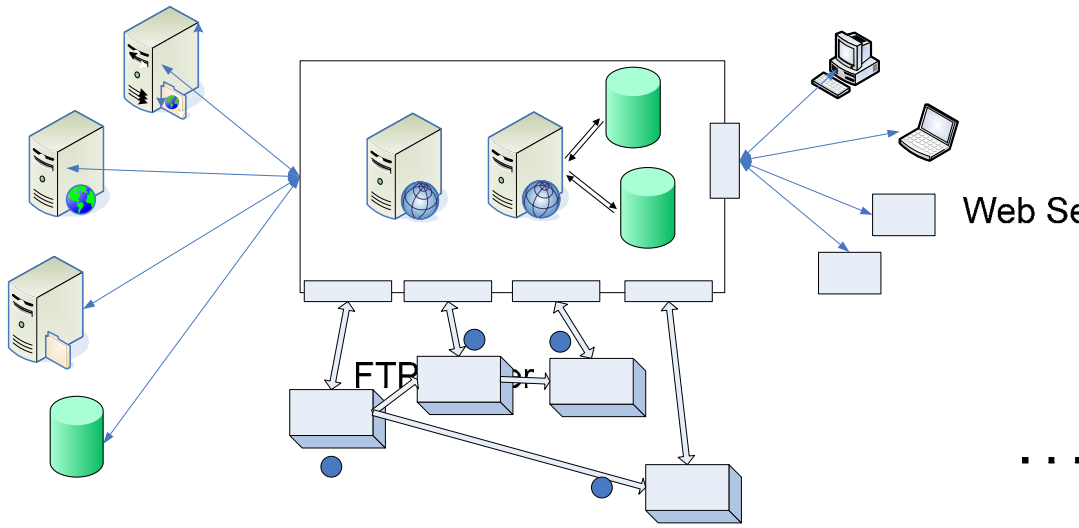

Figure 2 Major parts of the architecture and a sample workflow for processing geo-data using Web Services.

Designing OGC Web Services

As we have described above, GML is a data modeling language that can be used to encode geophysical data. We may then store this data in various archival systems and design Web Services that can query, retrieve, and update the data. These web services are compatible with the "Execution Grid" services illustrated in Figure 1.

These Web Services, because they use a generic data model, may be standardized and generalized. The OpenGIS Consortium defines specifications for several such services, with the Web Feature Service and the Web Map Service as two prominent examples. These services, unfortunately, are not designed to be Web Service compatible (they do not use WSDL or SOAP but rather lower level HTTP GET/POST conventions for messaging). In order to adapt these services to the QuakeSim architecture while taking advantage of existing OGC resources, we have redesigned these OGC services to use Web Service standards.

The Web Feature Service (WFS) [9] describes standards to publish, update, and delete geographic features, such as faults and GPS stations. 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 using SOAP messaging. 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 are implementing a Web Service version of Web Map Service to generate maps [10]. The Web Map Service gets data from various Web Feature Services. We are also building bridging services that allow our Web Service-compatible WMS to interact with non-Web Service versions of WMS. This will allow us to make use of extensive

existing mapping resources, such as the NASA OnEarth project (http://onearth.jpl.nasa.gov/). The interaction between these two services is based on SOAP messaging because of the Web Service standards.

GIS Information Services

Services such as the Web Map and Web Feature service, because they are generic, must provide additional, descriptive metadata in order to be useful. The problem is simple: a client may interact with two different Web Feature Services in exactly the same way (the WSDL is the same), but the Web Feature Services may hold different data. One, for example, may contain GPS data for the Western United States while the other has GPS data for Northern Japan. Clients must be able to query information services that encode (in standard formats) all the necessary information, or metadata, that enables the client to connect to the desired service. This is an example of the very general problem of managing information about Web Services. To address these problems, we are designing a general purpose information system, the Fault Tolerant High Performance Information System (FTHPIS), that we are applying initially to problems in GIS information management .

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) [11] 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 the Web Services registry research problem. However, one should be able to locate a Web Services satisfying a query in a decentralized, dynamically changing, distributed environment as well. To do this, discovery architecture needs to be defined. 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.

We classify metadata associated with Web Services 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-Context [12] provides an abstract context defining such metadata. Static metadata is the metadata describing a Web Service profile such as its usage cost, availability, bandwidth, computing power, storage capability, etc. We extend existing UDDI and WS-Context specifications in order to associate metadata with Web Service descriptions.

Acknowledgments

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

References

[1] QuakeSim Project Home Page: http://quakesim.jpl.nasa.gov/. For project documentation, see http://quakesim.jpl.nasa.gov/milestones.html.

[2] Foster, I. and C. Kesselman, 2003, *The Grid 2: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann.

[3] Berman, F., G. C. Fox, and T. Hey, eds., 2003, *Grid Computing: Making the Global Infrastructure a Reality* John Wiley & Sons, Chichester, England, ISBN 0-470-85319-0, March 2003. http://www.grid2002.org

[4] Solomon, S. C., (chair), 2002, "Living on a Restless Planet", Solid Earth Science Working Group Report. Available from http://solidearth.jpl.nasa.gov/PDF/SESWG_final_combined.pdf.

[5] Booth, D., H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard, *Web Services Architecture*. W3C Working Group Note 11 February 2004. Available from http://www.w3.org/TR/ws-arch/.

[6] G. Fox and A. Hey, eds. *Concurrency and Computation: Practice and Experience*, Vol. 14, No. 13-15 (2002).

[7] Chen, A. Y., S. Chung, S. Gao, D. McLeod, A. Donnellan, J. Parker, G. Fox, M. Pierce, M. Gould, L. Grant, and J. Rundle, 2003. Interoperability and semantics for heterogeneous earthquake science data. 2003 Semantic Web Technologies for Searching and Retrieving Scientific Data Conference, October 20, 2003, Sanibel Island, Florida.

[8] Youn, C., M. E. Pierce, and G. C. Fox., 2003 *Building Problem Solving Environments with Application Web Service Toolkits* To be published in Future Generation Computing Systems Magazine (in press).

[9] Vretanos, P (ed.) (2002), *Web Feature Service Implementation Specification*, OpenGIS project document: OGC 02-058, version 1.0.0.

[10] de La Beaujardiere, Jeff (2004), *Web Map Service*, OGC project document reference number OGC 04-024.

[11] Bellwood, T., Clement, L., and von Riegen, C. (eds) (2003), *UDDI Version 3.0.1: UDDI Spec Technical Committee Specification.* Available from http://uddi.org/pubs/uddi-v3.0.1-20031014.htm.

[12] Bunting, B., Chapman, M., Hurley, O., Little, M., Mischkinky, J., Newcomer, E., Webber, J., and Swenson, K., *Web Services Context (WS-Context)*, available from http://www.arjuna.com/library/specs/ws_caf_1-0/WS-CTX.pdf.