The Solid Earth Research Virtual Observatory: Web Services for Managing Geophysical Data and Applications

Marlon E. Pierce[1], Geoffrey C. Fox[1,2], Mehmet S. Aktas[1.2], Galip Aydin[1,2], Harshawardhan Gadgil[1,2], Zhigang Qi[1,2], and Ahmet Sayar[1,2]

[1]Community Grids Laboratory
Indiana University
501 North Morton Streeet
Bloomington, IN 47404

[2]Department of Computer Science
Indiana University
Lindley Hall, Room 215
150 S. Woodlawn Ave.
Bloomington, IN 47405-7104

{marpierc, gcf, maktas, gaydin, hgadgil, zqi, asayar}@indiana.edu

Web Services for Managing Geophysical Data and Applications

Abstract: *We describe our distributed systems research efforts to build the "cyberinfrastructure" components that constitute a geophysical Grid, or more accurately, a Grid of Grids. Service-oriented computing principles are used to build a distributed infrastructure of Web accessible components for accessing data and scientific applications. Our data services fall into two major categories: archival, database-backed services based around Geographical Information System (GIS) standards, and streaming services that can be used to filter and route real-time data sources such as Global Positioning System data streams. Execution support services include application execution management services and services for transferring remote files. These data and execution service families are bound together through metadata information and workflow services for service orchestration. Users may access the system through the QuakeSim scientific Web portal, which is built using a portlet component approach.*

Keywords: Cyberinfrastructure, Web Services, Grid Computing, Geographical Information Systems, Science Portals, Workflow

1. Introduction: Cyberinfrastructure for the Geosciences

The danger to human life and property posed by earthquakes has been demonstrated many times in the last several years. Earthquakes in Bam, Iran in December 2003, Sumatra in December 2004, and Kashmir in October 2005 all resulted in tens of thousands of deaths. The scale of these tragedies amply demonstrates the importance of understanding the immediate and long term causes and behaviors of interacting earthquake fault systems. Death tolls for recent large earthquakes in developed countries such as the United States and Japan have not been as catastrophic, but the economic cost of events such as the 1994 Northridge, California earthquake reinforce the need for increased understanding.

Geophysicists have developed a range of tools for investigating earthquake fault systems. Techniques include a) data mining techniques for analyzing time series data produced by sensors such as Global Positioning Systems; b) modeling codes that theoretically simulate the effects of the stresses associated with faults; c) data assimilation codes that attempt to find best-fit models to historical data records, with the best fitting models then used to forecast areas of increased hazard; and d) large scale probabilistic simulations that can model the behavior of extensive interacting fault systems such as the San Andreas fault system and its neighbors.

Such tools, which are typically driven by observational data inputs, are obviously excellent candidates for incorporating into a Grid [Foster2003] for geophysicists. Our project, the Solid Earth Virtual Observatory (SERVO) Grid, is developing the applications, the distributed computing infrastructure, the data management tools, and the Web portal user interfaces, for building this cyberinfrastructure [Atkins2003]. SERVOGrid is led by the NASA Jet Propulsion Laboratory and includes collaborators at the University of California-Davis, the University of Southern California, the University of California-Irvine, and Indiana University.

As with many other Grid projects, we began SERVOGrid with a focus on services to support computational applications, but as the project evolved, we realized that providing programming interfaces to remote data sources (both real time and archival) through Web Services was at least as important as managing the execution of remote applications. We thus consider SERVOGrid to be an example of a "Grid of Grids" [Fox2005a] in which we must integrate "Data Grid" and "Execution Grid" services (which are themselves independently useful) into a larger, cooperating system. In a Grid of Grids, services drawn from Data Grid and Execution Grid families must be integrated into composite applications. The development of tools for managing these distributed services is another important area of distributed systems research. An important subset of Grid management systems, known as Grid Workflow [Gannon2005], shepherds the execution of a composite application across several different, distributed services. Information management of Grids also falls in the service management category.

This paper summarizes the distributed computing systems research efforts of the SERVOGrid project. In Section 2 we provide a summary of requirements and describe a general Web Service architecture that can meet these requirements. We next summarize the Execution Grid and Geographical Information System Data Grid components that we have developed in Sections 3 and 4. User interface development using Web portals is an important component of our work and is summarized in Section 5. Service information and execution management research efforts are next discussed in Section 6. Approaches for monitoring deployed science portals and their supporting services are described in Section 7. Finally, open service architectures for science Grids imply broader community development models that have become pervasive in the current Internet but have not yet been fully exploited in scientific Grids. We conclude with a discussion of these issues in Section 8.

2. Applications, Data Requirements, and Architecture

To understand the SERVOGrid architecture, we first consider some of its geophysical application components and their data requirements. These codes are typically initially developed on desktops and workstations and then may in some cases be parallelized to run on clusters and supercomputers to solve problems of increasing size and complexity.

- **Disloc** models multiple dipping dislocations (faults) in an elastic half-space [Okada1985]. It uses fault models (such as may be obtained from the QuakeTables database, described below) as input.
- **Simplex** is an inversion code based on Disloc. Using observational surface stress values and an initial geometric model for one or more causative faults, Simplex can determine optimum fault model parameters.
- **GeoFEST** [Parker2005] is a three-dimensional viscoelastic finite element model for calculating nodal displacements and tractions. It allows for realistic fault geometry and characteristics, material properties, and body forces. GeoFEST relies upon fault models with geometric and material properties as input. These geometric models are converted into finite element meshes through supporting finite element mesh refining programs.
- **Virtual California** [Rundle2002] is a program to simulate interactions between vertical strike-slip faults using an elastic layer over a viscoelastic half-space. Virtual California relies upon fault and fault friction models as input.

- **Pattern Informatics** [Tiampo2002] calculates regions of enhanced probability for future seismic activity based on the seismic record of the region. Pattern Informatics uses seismic data archives as input.
- **Regularized Deterministic Annealing Hidden Markov Model (RDAHMM)** [Granat2004] is a time series analysis program that produces feature vectors and probabilities for transitioning from one (abstract) class to another. RDAHMM is typically used to analyze GPS and seismic catalog archives, but it can be applied to any time series data.

As can be seen from the above summary, SERVOGrid applications span a large range of computational geophysical research and rely heavily on observational input data.

Architecture

SERVOGrid is implemented as a collection of Web Services for accessing data sources, execution codes, visualization and mapping applications, and other tools. User interfaces to these services are implemented as Java Server Pages wrapped as portlets [Pierce2002a] [Abdelnur2003], which are in turn aggregated into a central portal. Figure 1 depicts the basic system architecture.

As we describe in more detail in subsequent sections, we have built a specific set of Web Services that can be adapted to manage the various applications and their data sources. Figure 1 illustrates several sample interactions: the QuakeSim portal mediates the user's interactions with remote servers, and remote services must also communicate with each other. Figure 1 also indicates a simple Grid workflow, which may be managed by the portal or through a separate workflow engine. As we describe below, we use Apache Ant-based Web Services for managing various local (to a specific host) applications. We manage distributed system state, such as is required to step through several tasks on different hosts, through the Context Manager Service, described below. This allows complicated tasks to run independently of the user's portal session: users can receive updates of their project's progress when they log back into the portal later.

As is discussed in greater detail in Section 6, we chose the portlet approach for building our Web portal. This had the primary advantage of allowing us to plug in other portlets into our system. Thus it is not difficult to create a user interface that combines portlets to our Web Service Grid components as well as portlets to various Globus Toolkit services, collaboration tools, news and information services, and so forth.

3. Execution Grid Services

SERVOGrid is based on the so-called "WS-I+" approach to building Web Service-based Grid systems [Atkinson2005]. We have implemented all services using standard Web Service specifications for WSDL and SOAP and have not introduced any non-standard extensions to these specifications. Our initial focus in SERVOGrid was to support and develop application management tools for the codes listed in the previous section. All services are implemented in Java and use the Apache Axis Web Service engine, but these implementation details are not needed to invoke the services.

Application Descriptor Web Services: Describing applications and hosts with a data model (or, more ambitiously, an ontology) has been part of the design of many projects. Our approach to this problem is to use linked, independent XML schemas for separately describing the hosts and applications [Pierce2002b] [Youn2003]. The XML schemas are designed primarily to simplify the management of user interfaces to SERVOGRid applications. We thus focused on modeling the codes' execution details, such as number and type of input and output parameters. These descriptions of metadata can then be used to build portlet interfaces for managing codes that are available to portal users.

Our primary usage of the XML data models is in constructing job descriptions for remotely executing the applications using the complementary execution management services described below. We have not attempted to develop a taxonomy or ontological description of the codes or in to describe the class of problems a particular code may be used to solve. To address this latter problem, we have explored extensions of our application and host modeling using Semantic Web and Case-Based Reasoning techniques, as described in [Aktas2004].

Execution Management Services: In our design of execution services, we realized the Apache Ant project offered many powerful capabilities that could be adapted to the problem of running executables. It already implements numerous tasks useful for interacting with the file system, including tasks for executing programs and for creating, renaming, and deleting files and directories. These tasks can be arranged to express dependencies and simple control logic, making it an ideal tool for handling the simple pre- and post-processing jobs as well as running the executable. In SERVOGrid, we use this service to wrap the applications described in Section 2. We have also found it useful to use this service to wrap such tools as IDL, Matlab, and GMT to provide simple visualization. Ant is written in Java and is open source, so it is a simple matter to build an Apache Axis Web Service with it.

Ant also offers a simple extension mechanism for adding custom tasks. We use this capability to define tasks that are wrappers to Web Service clients for file transfer and context management services. Such tasks can be executed as part of the Ant dependency chain to pull in files from remote hosts and to call back state changes to the portal in long-running applications. This allows us to implement the basic backend-to-backend communications shown in Figure 1. Ant tasks also allow us to define simple workflows and job sequences, and (unlike basic Globus GRAM services) give us some control of the user's actions on the server side; i.e., the user only gets access to specific tasks and cannot run arbitrary commands on the server. This is enforced on the Web Service side as well as the user interface side.

To wrap a specific application, we develop an Ant *build.xml* file template. This XML file specifies all the tasks that are to be executed, their execution order, and their inter-dependencies. This template may be used for managing pre- and post-processing steps as well as for running the specific application. For example, we use this approach to couple GeoFEST with finite element mesh generating applications. We invoke the service by passing string arrays of parameter values to the template. We may also extend this general service to build WSDL interfaces that are specialized to specific services.

The execution service is not associated with user authentication. Users are authenticated through the portal, and the portal is responsible for keeping track of individual users's jobs and data. The codes themselves run on a generic account on the server.

One of the core difficulties with this approach to application management services is that the services depend directly on the robustness of the application. Unless the service is properly constrained, it is very easy for naïve users to set up potentially troublesome inputs that cannot be handled by the application's logic. A common example is input that results in the code entering an infinite loop in which error messages are written repeatedly to a file until the host computer's file system is filled.

File Transfer and Management Services: We have implemented file transfer Web Services for moving data between various backend machines (see Figure 1). These services can also be used to upload and download data between the user's desktop and the backend host machines, using the portal server as an intermediary. This service may also work behind the scenes, called by the Ant execution service described above to move an output file to a GMT plotting service, for example. By coupling the file transfer service with the context management service, we have created virtual file spaces for the user: the context manager keeps track of the file location for a specific user project. This metadata can then be used to construct invocations of the file transfer service, which is responsible for actually transferring the file. Typically this is used to allow the user to directly download data or to transfer data from the execution host to a visualization service.

Context Management Services: The QuakeSim portal and its codes allow users to execute geophysical applications on a Grid, but we must go beyond this to provide a useful user experience. One way in which we address this is to provide an "electronic notebook" style user interface that lets users set up their problems in the form of projects. This project interface provides a way of maintaining user information (the codes selected, the input files used, the output generated, and so forth). The user interfaces are backed by a persistent metadata service that we call the context manager. We store user information using structured name-value pairs, which can be constructed as arbitrarily deep trees. We refer to these groups of metadata as context [Pierce2002b]. In our terminology, a context is an instance of a simple XML schema, and these XML instances are linked through parent-child relationships. A context has a simple, URI-like name that can be used to access its data. This service is part of the QuakeSim portal deployment, but it has been superseded in later projects by our implementation of the WS-Context specification [Aktas2006a].

4. Data and Information Grid Services

Quake Tables Fault Database

The QuakeTables Web Service and Web accessible database [Chen2003] [Grant2005] acts as a data repository for earthquake fault data in the SERVOGrid project. It includes location, geometric and material characteristics, and provenance information such as the source (author, journal information, etc) for a particular fault entry. QuakeTables provides both a human usable Web interface (wrapped as a portlet) and a WSDL-based Web Service programming interface. The former interface is useful for delivering data to human users while the latter interface is

useful for delivering data to applications. Using the latter interface, we have integrated QuakeTables with execution management services for GeoFEST, Disloc, and Simplex through the QuakeSim portal.

Geographical Information System Services for Archival Data

Geographical Information System (GIS) standards may be adapted to meet many of the data and metadata requirements surveyed in Section 2. The Open Geospatial Consortium [OGC] defines an extensible (and extensive) XML data model, the Geographic Markup Language (GML) [Cox2003], for describing geospatial and geo-temporal features. This common data model is then integrated with a number of services. The OGC defines standard service definitions and interoperability guidelines.

We have implemented the following "Information and Data Grid" Web Services:

- **Data Services:** We implemented the Web Feature Service [Vrertanos2002] to store and retrieve seismic data archives, GPS data archives, and faults (in a GIS variant of the QuakeTables service). An OGC feature is a GML description of a map object. The description may include both information on how to render the feature as well as other useful metadata, such as the name of the feature and other observation data associated with it. More information our work here is available from [Aydin2005].
- **Map Generation Services:** We implemented the OGC's Web Map Service specification [Beaujardierre2004] as a Web Service. The Web Map Service is used to generate vector and raster maps in various formats (SVG, JPEG, GIF, etc). These maps typically include realizations of abstract feature data obtained from Web Feature Services. Our Web Map Service can also integrate maps from other Web Map Servers as an overlay. This is described in more detail in [Sayar2006].
- **Information Services:** One useful feature of the OGC service specifications is that they include a standard XML metadata description ("capabilities") and query method that can be used to describe and obtain information about what the service does. The OGC defines information services (catalogs) for aggregating capability information. We decided, however, that these specifications were too GIS specific and could be substituted with more general, UDDI-based systems. We developed an extension of UDDI along these lines to support general metadata extensions and XPath queries, with specific realizations for the OGC capabilities file. See [Aktas2005a] and [Aktas2005b].

Real Time Data Grids

In addition to request-response style access of archived and curated data, geophysical applications also need to support the real-time and near real-time analysis of data streams. The specific problem within the SERVOGrid project is to provide the infrastructure for coupling real-time GPS data to time series analysis codes. The Scripps Orbital and Permanent Array Center maintains the California Real Time Network (CRTN) (see http://sopac.ucsd.edu/projects/realtime/), which we may directly access to obtain station position measurements at a rate of 1 Hz.

As described in Section 1, RDAHMM can be used to detect underlying mode changes in archived GPS signals. These modes, which can be determined independently of any *apriori* modeling, may be associated with underlying physical processes such as earthquakes and more

subtle aseismic events. Typically, RDAHMM has been applied to much longer time scales (daily time series values over several months or years), so the use of RDAHMM in the analysis of real-time data is an area of research. From the distributed systems research point of view, however, we must investigate the computing infrastructure that will enable real-time RDAHMM.

We have developed a real-time stream filtering infrastructure by using topic based publish/subscribe software, NaradaBrokering [Pallickara2003]. Our system currently supports 70 individual GPS stations in CRTN. The system works as a sequence of filters that transform binary GPS data streams and is depicted in Figure 2. A detailed description is available from [Fox2005b] and is summarized here. Each network source stream (which typically corresponds to 5-10 individual GPS stations) is initially published in binary RYO format. Each network's binary message is then received by an ASCII decoder, which republishes on a new subtopic. We may add further filters, such as a de-multiplexing filter that can separate the 5-10 interleaved GPS station position values in each RYO message and republish them on 5-10 different topics. This allows other filters and applications to register for individual station feeds. We have also implemented signal archive filters that can store and replay network time series on demand. Application filters for stream analysis can be built in a similar way to these simpler filters. We have prototyped the integration of an RDAHMM filter with the real-time data streams, as shown in Figure 3. This is a proof of concept: the system deployment is stable and persistently available, but the RDAHMM results (the modes identified in the signal) should not currently be interpreted as meaningful. We are in the process of evaluating the proper usage of RDAHMM on these time series.

Our current system is deployed and stable for the 70 stations of the Southern California network. We are in the process of a systematic series of throughput and scaling tests of our software. Our goal in throughput testing is to test the performance of the system to determine the overhead associated with simple filters under various publish and subscribe loads. Our scaling test is to determine the overall load of signals that can be supported by a single message broker and to investigate scaling strategies that use multiple brokers that behave as a single logical message broker.

5. Science Web Portals and AJAX Techniques

The QuakeSim portal has been developed around the portlet component model, which allows a Web portal to be decoupled into an aggregating container and a set of semi-independent portlet plugins. Each portlet is a self-contained Web application that (directly or indirectly) implements the portlet programming interface. This allows portlet components to be easily exchanged between developers and deployed in multiple Web portals. QuakeSim has been implemented using the Jetspeed 1 portlet container.

Our initial development concerns when working with Jetspeed were to support development environments that were not entangled with the container; that is, portlets should be relatively free of calls to the parent container services. We still advocate this design principal. We also wanted to ensure that portlets could be developed as (more or less) standard Java Server Pages, rather than relying on Jetspeed specific templates. To meet these requirements, we developed a general purpose *WebFormPortlet* as an extension of the Jetspeed 1 portlet base class. WebFormPortlet features include

- Support for portlets running independently (on other web servers) of the portlet container.
- Support for shared session state (through Tomcat's JSESSIONID cookie) between the container and remote portlets.
- Support for HTML <form> processing: POST parameters can be passed from the aggregating portlet container to the remote portlet.
- Portlet navigation: URL links and form actions are rewritten to allow navigated pages to appear within the portlet container rather.
- Support for both HTTP and HTTPS connections.

Our Jetspeed-based approach and WebFormPortlet have subsequently been superseded by the Java Specification Request (JSR) 168 [Abdelnur2003] and the Apache Portlet Bridges projects, respectively. We are currently updating the QuakeSim portal to comply with these standards.

Portlets are primarily a server-side Java technology and have been widely adopted in a number of science gateway projects. In parallel development, improved support in major browsers for JavaScript standards has enabled the prominent re-emergence of client-side browser applications, with a number of examples such as Google Maps. These rich browser client interfaces are developed using techniques that are collectively dubbed Asynchronous JavaScript and XML (AJAX). AJAX itself is a methodology, not a specific piece of software. AJAX is not purely a browser-side technology but instead makes use of the now widely supported *XMLHttpRequest* object in JavaScript to decouple user interactions in the browser from the browser requests to remote services. This greatly increases the interactive capabilities of Web browser user interfaces while maintaining server-side data. We have very actively pursued integration of our GIS Web Services (particularly our Web Feature Services and their GPS, seismic, and fault data) with Google Maps. We find this particularly useful for simulating "server push" in our real-time GPS data work (described in Section 4), as shown in Figure 3.

6. Service Coordination and Workflow

General purpose Web services for data access and application management need to be combined into composite applications capable of performing specific scientific tasks. Such coordination in Grid computing is typically called scientific workflow and is an active research area for numerous groups. Example scientific workflow research projects include Kepler [Bowers2005], Taverna [Oinn2004], and SciFlo [Wilson2005]. See also [Gannon2006] and [Ludäscher2005] for overviews.

Workflow may be generally understood as the execution of a flowchart or (more simply) a directed acyclic graph of operations. Workflow systems at their most basic are ways for coupling (either graphically or through scripting) reusable components into composite applications. In Grid system, the components of the graph are proxies to remote services. Workflow systems for Grids thus must address the challenges of distributed execution environments: they must maintain a directory of available services, they must support non-local data, and they must be able to handle distributed state; that is, they must be able to monitor the stage of an execution workflow.

Typically, the above Grid workflow requirements are addressed within a single workflow engine. In our approach, we have researched the decoupling of execution and state management, and we have addressed the data transportation problems. Our workflow engine research project is called

HPSearch and is described in greater detail in [Gadgil2005]. HPSearch uses Javascript as a workflow scripting engine and can bind Web service proxies into flows. One of HPSearch's distinguishing features is its ability to handle and manage workflow components that communicate with data streams, or data sent continuously over network connections. This is particularly important for scientific workflows, which must process and transfer data between distributed components. Typically, Grid workflow engines do not take this directly into account, so data is often inefficiently passed directly back to the workflow engine, where it is resent to the next component. We have used the Pattern Informatics application (see Section 2) in several tests cases for our workflow system. This integration and our evaluation are described in more detail in [Aydin2005].

Distributed workflows are composed out of Web Service components collected for a specific task. This is in fact the strength of the Web Service approach: services can be collected to solve a wide variety of problems. Sophisticated, specialized composite applications are composed of general purpose service components. To realize this in a distributed system, we need dynamically updated information systems that describe services so that they can be assembled into workflows, and we need information systems that can manage the internal state of these composite applications. The latter case is best understood as a communication of the stage in the execution of the workflow. Different components in the system will need to be updated as each service completes or fails in its execution, and we need a place to put various pieces of supporting metadata, such as the URLs for generating output files. Our research in this area is based on the Web Service specifications UDDI and WS-Context. More detailed information on architecture, implementation, and testing are available from [Atkas2006a] [Atkas2006b]

7. Unit and System Testing of Grid Deployments

One of the difficulties in managing Grids and Grid portal deployments is monitoring the system for failures. Similarly, science portal systems don't easily fit within a traditional "white-box" style of unit testing, which assumes knowledge of the source code of all parts of the system, since they depend both on remote clients (users' browsers) and remote Web services that in turn wrap binary executables. The Inca project [Smallen2004] is an example effort in Grid testing and monitoring. User interface-driven testing, such as is needed by science portals, is an open area of systems research, but can be very useful as a way of encapsulating composite tests that simulate real usage scenarios. We describe here some of our pragmatic solutions to these problems.

In SERVOGrid, we have developed a "black box" system testing environment for the portal. This functionally tests the running portal instance to simulate user interactions with the portal. Unit tests verify that the HTML returned by the browser contains expected responses and does not contain error messages. Since the responses are generated through interactions with remote services, we can indirectly test these services to make sure they respond correctly to known inputs. We have found the Apache Ant, Apache JMeter, and HttpUnit projects to be especially useful for this task. JMeter and HttpUnit provide "black-box" testing that simulates a user's interactions with a portal. We have developed an HttpUnit-based test suite that enables us to simulate portal usage: the tests can be used to log into the portal, navigate user pages, and launch applications. These tests are run daily to monitor the system.

8. Conclusions and Future Work

This article has reviewed the development work of the SERVOGrid project. We have adopted a Web Service-based architecture for encapsulating geophysical applications and data sources as simple Web services with well-defined programming interfaces. This allows diverse clients to interact with our services and provides an open architecture for other application developers to reuse our services and our portlet components.

Web services and portlet-based portals have been adopted by a number of projects, with GEON [Youn2005] and LEAD [Plale2005] serving as two relevant examples in geo-sciences and atmospheric sciences, respectively. In our opinion, one of the shortcomings of all of these projects is not in the basic architecture but in the lack of reuse of the projects' running service instances in clients built by third party groups. In the general Web development community, so called "mash-up" web applications are widely used. These are novel, easily-developed user interfaces built by using Web client programming interfaces (typically written in JavaScript) to connect to remote services from general service providers. An interesting list of publicly available service interfaces is available from http://www.programmableweb.com/apis. Enabling the easier development of science mash-ups is a potentially important future development for broadening the impact of cyberinfrastructure developments. The SERVOGrid project has followed an open, service-oriented architecture compatible with the "programmable web" paradigm, but we need to investigate lighter-weight bridges to our services.

The other major challenge for SERVOGrid and for science portals and gateways in general is the technical challenges of integrating AJAX-style Web development with portlet development. AJAX, when done well, provides a substantial improvement in the user interface capabilities over purely server-side technologies. AJAX relies heavily on JavaScript in the client browser, while JSR 168 portlets are a server-side technology. It is an important activity for Web portals in the geosciences to develop JavaScript libraries for easily integrating services into such technologies as Google Maps. The current JSR 168 standard has some important limitations in dealing with JavaScript library imports, but this should be addressed in the next version of the standard (JSR 286) that is currently being developed. More generally, integrating the two represents a challenge in state management: the information in the user's browser must be kept synchronized with the information on the server side, since the server side objects act as proxies to multiple remote Grid resources.

The technical issues we have raised in this section are solvable problems and are currently under investigation by several groups. More importantly the implications of open architectures adopted by our project and others need to be more fully exploited. We hope that the next generation of science cyberinfrastructure will be driven by a much more community-driven development environment. We believe the geosciences will be an important example in this new development model.

Acknowledgements

This work was supported by the National Aeronautics and Space Administration's Computational Technologies (CT) and Advanced Information System Technology (AIST) programs. We thank Dr. Yehuda Bock and his group at the Scripps Orbital and Permanent Array Center for their assistance with the California Real Time Network.

References

[Abdelnur2003] Abdelnur, A., Chien, E., and Hepper, S., (eds.) (2003), Portlet Specification 1.0. Available from http://www.jcp.org/en/jsr/detail?id=168.

[Aktas2004] Mehmet S. Aktas, Marlon E. Pierce, Geoffrey Fox, David B. Leake: A Web based Conversational Case-Based Recommender System for Ontology aided Metadata Discovery. GRID 2004: 69-75.

[Aktas2006a] Mehmet S. Aktas, Geoffrey C. Fox, Marlon Pierce: Information Services for Dynamically Assembled Semantic Grids. Proceedings of 1st International Conference on SKG2005 Semantics, Knowledge and Grid Beijing China November 27-29 2005. (To appear) in IEEE Proceedings of 1st International Conference on SKG2005 Semantics, Knowledge and Grid Beijing China November 27-29 2005.

[Aktas2006b] Mehmet S. Aktas, Geoffrey C. Fox, Marlon Pierce: Fault Tolerant High Performance Information Services for Dynamic Collections of Grid and Web Services. (To appear) in "Semantic Grid and Knowledge Grid: Systems and Applications" Special Issue of Future Generation Computer Systems: The International Journal of Grid Computing: Theory, Models and Applications.

[Atkins2003] D. E. Atkins, K. K. Droegemeier, S. I. Feldman, H. Garcia-Molina, M. L. Klein, D. G. Messerschmitt, P. Messina, J. P. Ostriker, and M. H. Wright, “Revolutionizing Science and Engineering Through Cyberinfrastructure.” Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure, January 2003. Available from http://www.nsf.gov/cise/sci/reports/atkins.pdf.

[Atkinson2005] Malcolm P. Atkinson, David De Roure, Alistair N. Dunlop, Geoffrey Fox, Peter Henderson, Anthony J. G. Hey, Norman W. Paton, Steven Newhouse, Savas Parastatidis, Anne E. Trefethen, Paul Watson, Jim Webber: Web Service Grids: an evolutionary approach. Concurrency - Practice and Experience 17(2-4): 377-389 (2005).

[Aydin2005] Galip Aydin, Mehmet S. Aktas, Geoffrey C. Fox, Harshawardhan Gadgil, Marlon Pierce, Ahmet Sayar SERVOGrid Complexity Computational Environments (CCE) Integrated Performance Analysis Proceedings of Grid Computing Conference, 2005. The 6th IEEE/ACM International Workshop 13-14 Nov. 2005. Page(s): 256 - 261 DOI

[Beaujardierre2004] de La Beaujardiere, Jeff, Web Map Service, OGC project document reference number OGC 04-024.

[Bowers2005] Shawn Bowers, Bertram Ludäscher: Actor-Oriented Design of Scientific Workflows. ER 2005: 369-384; Efrat Jaeger, Ilkay Altintas, Jianting Zhang, Bertram Ludäscher, Deana Pennington, William Michener: A Scientific Workflow Approach to Distributed Geospatial Data Processing using Web Services. SSDBM 2005: 87-90.

[Chen2003] Chen, A., Donnellan, A., McLeod, D., Fox, G., Parker, J., Rundle, J., Grant, L., Pierce, M., Gould, M., Chung, S., and Gao, S., Interoperability and Semantics for Heterogeneous

Earthquake Science Data, International Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data, Sanibel Island, FL, October 2003.

[Cox2003] Cox, S., Daisey, P., Lake, R., Portele, C., and Whiteside, A. (eds) (2003), OpenGIS Geography Markup Language (GML) Implementation Specification. OpenGIS project document reference number OGC 02-023r4, Version 3.0

[Foster2004] Foster, I. and Kesselman, C., (eds.) The Grid 2: Blueprint for a new Computing Infrastructure, Morgan Kaufmann (2004)

[Fox2005a] Geoffrey Fox, Sang Lim, Shrideep Pallickara, Marlon Pierce: Message-based cellular peer-to-peer grids: foundations for secure federation and autonomic services. Future Generation Comp. Syst. 21(3): 401-415 (2005).

[Fox2005b] Geoffrey Fox, Galip Aydin, Harshawardhan Gadgil, Shrideep Pallickara, Marlon E. Pierce, Wenjun Wu: Management of Real-Time Streaming Data Grid Services. GCC 2005: 3-12

[Gadgil2005] Harshawardhan Gadgil, Geoffrey Fox, Shrideep Pallickara, M. Pierce, R. Granat: A scripting based architecture for management of streams and services in real-time grid applications. CCGRID 2005: 710-717.

[Gannon2006] Dennis Gannon and Geoffrey Fox Workflow in Grid Systems Editorial of special issue of Concurrency & Computation: Practice & Experience (in press).

[Granat2004] Granat, R. A., *Regularized Deterministic Annealing EM for Hidden Markov Models*, Ph.D. Thesis, University of California, Los Angeles, 2004.

[Grant2005] Grant L.B., A. Donnellan, D. McLeod, M. Pierce, G.C. Fox, A.Y. Chen, M.M. Gould, S.S. Sung, P.B. Rundle, A Web-Service Based Universal Approach to Heterogeneous Fault Databases, Computing in Science and Engineering Special Issue on Multi-Physics Modeling, in press.

[Ludäscher2005] Bertram Ludäscher, Carole A. Goble: Guest editors' introduction to the special section on scientific workflows. SIGMOD Record 34(3): 3-4 (2005)

[Oinn2004] Thomas M. Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, R. Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat, Peter Li: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20(17): 3045-3054 (2004).

[Okada1985] Okada,Y.(1985) Surface deformation due to shear and tensile faults in a half-space, Bull.Seism.Soc.Am., 75, 1135-1154.

[Pallickara2003] Shrideep Pallickara, Geoffrey Fox: NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. Middleware 2003: 41-61.

[Parker2005] Parker, J., Lyzenga, G., Donnellan, A., Judd M., Baker T., Norton C., Tisdale E., Li P., Using the GeoFEST faulted region simulation system, Proceedings of the 4th ACES Workshop, Beijing China (in Press)

[Pierce2002a] Marlon E. Pierce, Geoffrey Fox, Choon-Han Youn, Stephen Mock, Kurt Mueller, Ozgur Balsoy: Interoperable Web services for computational portals. SC 2002: 1-12.

[Pierce2002b] Marlon E. Pierce, Choon-Han Youn, Geoffrey Fox: The Gateway computational Web portal. Concurrency and Computation: Practice and Experience 14(13-15): 1411-1426 (2002).

[Plale2005] Beth Plale, Dennis Gannon, Daniel A. Reed, Sara J. Graves, Kelvin Droegemeier, Bob Wilhelmson, Mohan Ramamurthy: Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD. International Conference on Computational Science (2) 2005: 624-631

[Rundle2002] Rundle, JB, PB Rundle, W Klein, J Martins, KF Tiampo, A Donnellan and LH Kellogg, GEM plate boundary simulations for the Plate Boundary Observatory: Understanding the physics of earthquakes on complex fault systems, *Pure and Appl. Geophys.*, 159, 2357-2381 2002.

[Sayar2006]Ahmet Sayar, Marlon E. Pierce, Geoffrey Fox: Integrating AJAX Approach into GIS Visualization Web Services. AICT/ICIW 2006: 169

[SensorML] Sensor Model Language (SensorML) Project Web Site: http://vast.nsstc.uah.edu/SensorML/.

[Smallen2004] Shava Smallen, Catherine Olschanowsky, Kate Ericson, Pete Beckman, Jennifer M. Schopf: The Inca Test Harness and Reporting Framework. SC 2004: 55.

[Tiampo2002] Tiampo, K.F., Rundle, J.B., McGinnis, S., and Klein, W. Pattern dynamics and forecast methods in seismically active regions, Pure and Applied Geophysics, 159, 2002.

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

[Wilson2005] Brian Wilson, Benyang Tang, Gerald Manipon, Dominic Mazzoni, Eric Fetzer, Annmarie Eldering, Amy Braverman, Elaine R. Dobinson, Tom Yunck: GENESIS SciFlo: Scientific Knowledge Creation on the Grid using a Semantically-Enabled Dataflow Execution Environment. SSDBM 2005: 83-86.

[Youn2003] Choon-Han Youn, Marlon E. Pierce, Geoffrey Fox: Building Problem Solving Environments with Application Web Service Toolkits. International Conference on Computational Science 2003: 403-412.

[Youn2005] Choonhan Youn, Tim Kaiser, Cindy Santini, Dogan Seber: Design and Implementation of Services for a Synthetic Seismogram Calculation Tool on the Grid. International Conference on Computational Science (1) 2005: 469-476.

Figures

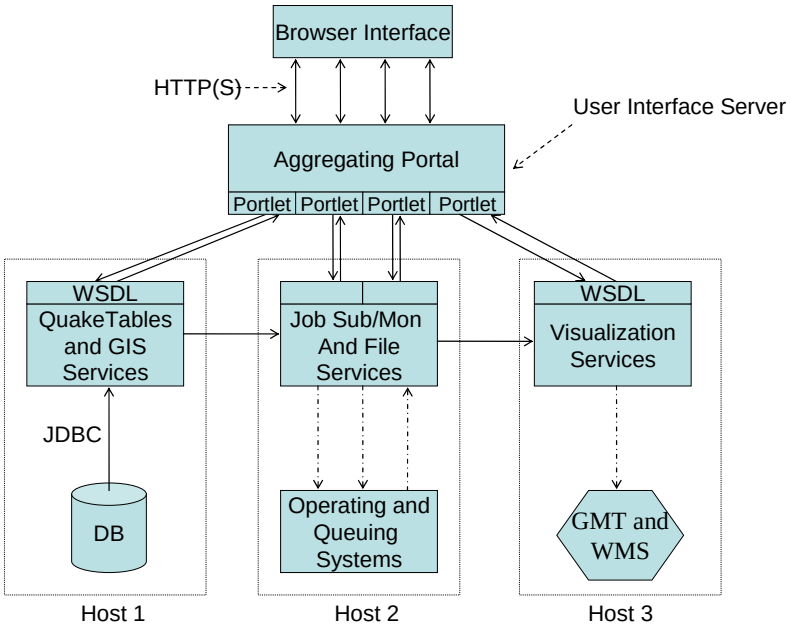
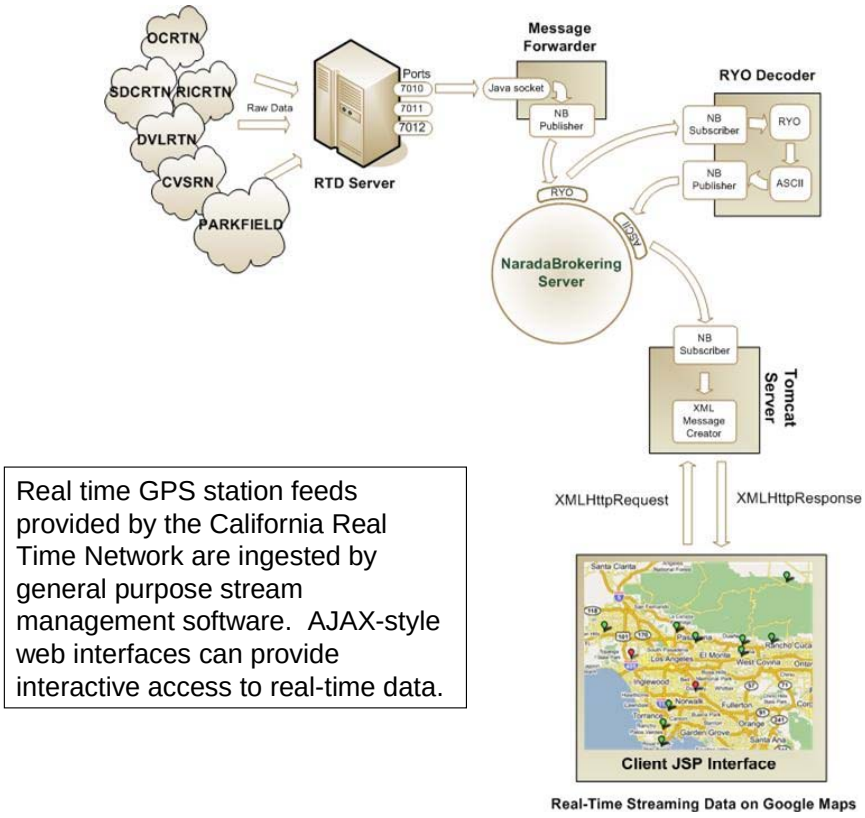


Figure 1 The SERVOGrid architecture consists of distributed Web Services accessed through the QuakeSim user portal. Solid arrows represent network connections (SOAP over HTTP unless otherwise indicated). Broken arrows represent interactions with the local operating system. As shown in the figure, the portal may serve (in effect) as the state and workflow management system for the services. We also provide these as separate services through the HPSearch and WS-Context development work.



Real time GPS station feeds provided by the California Real Time Network are ingested by general purpose stream management software. AJAX-style web interfaces can provide interactive access to real-time data.

Figure 2 Live GPS data streams are processed using message filters that publish and subscribe to messaging topics. Open arrows represent TCP/IP connections. GPS network data (typically from 5-10 individual GPS stations per network) is streamed from sockets from the RTD server. Message forwarding publishers connect to these sockets and publish to topics on a publish/subscribe broker (NaradaBrokering in the figure). Message filters such as the RYO Decoder in the figure are subscribers to topics and can be used to transform or analyze messages. The results of these messages are published on new topics. For example, the individual stations' position data can be extracted from the composite signal, and each individual station's real time positions can be made available on a new topic.

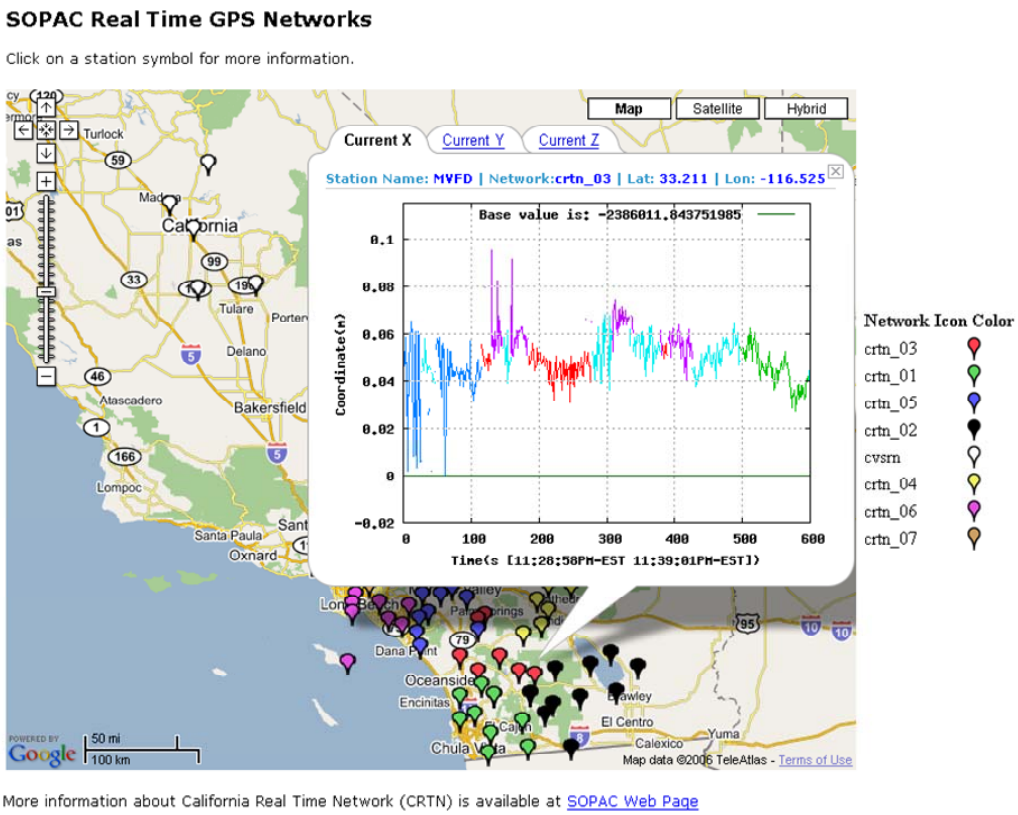


Figure 3 Real-time GPS position data for an individual GPS station is analyzed on-the-fly with RDAHMM and displayed with Google maps. The colors in the graph indicate underlying modes in the GPS signal determined by RDAHMM's analysis.