Design and Implementation of Services for a Synthetic Seismogram Calculation Tool on the Grid

Choonhan Youn, Tim Kaiser, Cindy Santini, and Dogan Seber

San Diego Supercomputer Center, University of California at San Diego,
9500 Gilman Drive,
La Jolla, CA 92093-0505
{cyoun, tkaiser, csantini, seber}@sdsc.edu

Abstract. We have built user environments that simplify and provide interactive access to data, models, and compute resources as well as integrate various distributed computational services to study earthquake waveforms utilizing 3D models and compute resources within the Geosciences Network (GEONgrid) and national computational grids, such as TeraGrid. These data and computing services are implemented using a Web services approach and are incorporated in a service-based portal architecture. We then illustrate how these data, models, and services can be used to build distributed, interactive scientific applications.

1 Introduction

Application-specific web-based scientific portals provide user-centric views of computational grid technologies based on global, large-scale distributed computing for scientific applications [1], [2]. Building on a foundation of distributed service components, we can typically construct a sophisticated, domain-specific portal system. These services may be built on top of Grid technologies such as Globus, Legion, and Condor, and web technologies and standards that are developed for Internet computing and are used to provide browser-based access to High Performance Computing (HPC) systems. Numerous such portals have been developed, with varying degrees of specializations. Some examples include NASA's Information Power Grid [3], NPACI's Hotpage [4] and application-specific Problem Solving Environments, PNNL's Extensible Computational Chemistry Environment (Ecce) system [5], UNICORE [6], Gateway System [7], and NMI's Open Grid Computing Environment (OGCE) portal [8].

As part of GEON's (Geoscience Network) computational and open grid computing environment (GEON project [9] funded by NSF), we have started developing a domain-specific application portal (SYNSEIS – SYNthetic SEISmogram generation tool) to help seismologists as well as any other researchers to calculate realistic 3D regional seismic waveforms using a well-tested, finite difference code, e3d. E3d was developed by the Lawrence Livermore National Laboratory [10]. This system is also designed to be used in day-to-day activities of researchers, especially EarthScope scientists who will be accessing data from hundreds of stations everyday and need to process the data in a timely fashion.

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3514, pp. 469 – 476, 2005.

As the emergence of rich clients and rich Internet applications become important, we develop a user-friendly interface to SYNSEIS using Macromedia Flash MX [11]. Within our SYNSEIS user interface, we provide several components: an interactive mapping tool, event/station/waveform extraction tools that allow users to seamlessly access IRIS Data Management Center (DMC)'s remote archives [12], simulation of synthetic seismograms on the TeraGrid machines, and monitoring the job status.

This paper describes our design and initial efforts for building SYNSEIS application tool as one of the main GEON's science applications on the GEONgrid portal. The GEONgrid portal provides the unifying hosting environment for managing geoscience data and applications via the GEONgrid [13]. It is built out of portlet containers that access distributed data and computational resources via Web services, Grid services and other Internet computing technologies. Using GEON grid environments and national-scale TeraGrid supercomputer centers [14] for high performance computing, the SYNSEIS application is developed as a portlet object that can provide the hosting environments interacting with the codes and the data retrieval systems. It is also built using a service-based architecture for reusability and interoperability of each constituent service components which are exposed as Web services. This allows multiple use-case scenarios. Each service component can be reused by other researchers and portal developers.

2 SYNSEIS Architecture

The SYNSEIS architecture is based on a Web Services model which has received a great deal of attention from both the commercial and the Grid computing communities, the latter through the Globus group's proposed Open Grid Services Architecture (OGSA) [15], illustrated in Figure 1. Web Services [16] essentially do not present new concepts for distributed computing but rather implement important simplifying, standards-compliant ways for entities to find and invoke the appropriate remote service. These have important implications to the field of computational or science portals.

The user interacts with the SYNSEIS application tool through a web browser, which accesses a central user interface server that contains the collection of Web service clients in Web service connectors of Macromedia Flash MX [17]. The central goal of the SYNSEIS user interface (Figure 2) is to allow users to analyze seismic data hosted at the IRIS DMC which stored the recorded seismogram data, network and station data, and the earthquake event data [12] and calculate realistic 3D seismograms using the computational resources via the grid system. Using the data archival service, the user selects the bounding box area on the map and an event-station pair within the region, and then plots the waveform data which is accessed via IRIS DMC for the given event-station pairs. In order to run the application, e3d, the user selects the computational resource and creates the input data as an XML object using the input form provided by an interactive user interface for submitting the job. The input format for e3d application may be expressed in XML document which provides a useful data exchange language [18]. The XML object may be used to encode and provide structure to code input files and output data. We use XML descriptions as an independent format that may be used to generate input files for the codes in the proper legacy format.

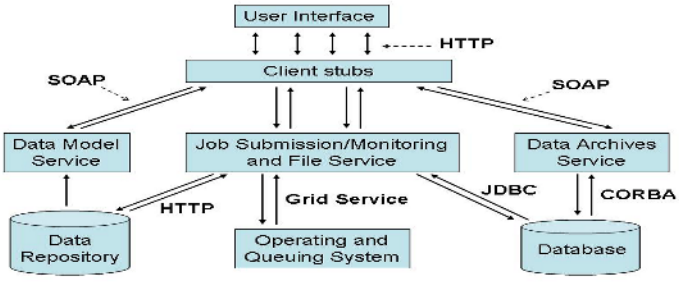

Fig. 1. SYNSEIS architecture with Web service invocations of remote services. Arrows indicate remote invocations with indicated protocols

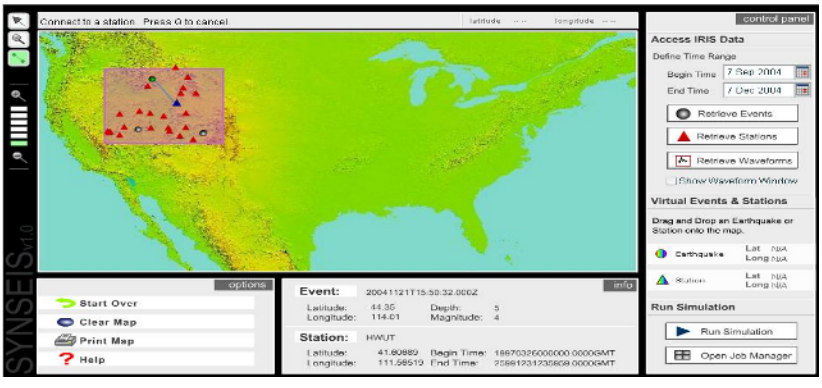

Fig. 2. SYNSEIS's interactive graphic user interface

A SYNSEIS user interface server maintains several client stubs that provide local interfaces to various remote services. These remote services are described in the WSDL format [19] and are invoked via SOAP [20] over HTTP. These services are invoked on various service-providing hosts, which may in turn interact with local or remote databases, data repositories, or remote queuing and system environments via grid services as shown in Figure 1.

The merit of the service-based architecture which we pursue in SYNSEIS tool allows the portal user to browse the geologic data model's service on a particular host, specify model parameters, and create the input data file and transfer it to the job submission service on a particular host. This service host maintains remote application executables, and the application may be run with the appropriate input data on a queuing system which may include a cluster or parallel computer. Following with the completion of the job execution, the user may transfer the simulation outputs back to another database or file system, or download the files to his/her desktop. Based on these service components that are described in Section 3, portal developers can also compose them in their portal environments easily.

3 Job and File Management

3.1 Job Submission with File Transfer

A job submission service is a basic component required to integrate the computational grid with Web Services. This service may execute operating system calls directly or may interact with Grid services through client APIs. We implement this service with file transfer on top of Grid technologies using Java CoG Kit [21] which is the Globus interface to run jobs on remote computational resources in a secure and authenticated manner. SOAP requests having the user input (XML string) and a compute host name as input parameters can be sent out to the targeting SOAP server which manages and interacts with computational grids.

Before submitting the job to the targeted host, this user XML document is stored into the data repository and is reused by the user later (for example, to modify some input parameters) for resubmitting the job, or getting the output file names.

The job RSL (Resource Specification Language) [22] script which contains the application metadata is needed to run the Globus job through the gatekeeper. It is created and integrated with Application Information Web Service developed by the Community Grids Lab, Indiana University, Bloomington (More detailed description is available from [23]) as the information repository describing e3d application for the seismic simulation and other system commands for dealing with the job files such as ls, rm, qstat, and so on. And the GASS (Global Access to Secondary Storage) server is created to receive the standard output/error. The GASS URL is also set as the stdout/stderr parameter in this job RSL script. This will stream the job output/error to GASS server that is redirected to the Job output listener.

Before submitting the job RSL script to the gatekeeper (GRAM server), the user input XML file that the application needs can be transferred to one of the hosts of computational grids (currently, we are using national computational grids, TeraGrid) through the GridFTP. After submitting the job RSL script, the output results describing the running job are caught from the GASS server. The information is parsed to get the job information to create the job table that specifies User ID, Job ID, Job Host Name, Job Error Handling message, user XML input's file name, Job directory name, Job submission time, and Job status. The job table is also stored into the job database. Finally, this job information is returned back to the client that invokes this job submission service as WSDL complex type.

3.2 Job Monitoring with File Transfer

A job monitoring service is also very basic service component in most of computational portals. Users are easily able to check the current job status from the queuing system about jobs on the high performance computing resources through the Web service client APIs. The job monitoring service is implemented using Java CoG Kit, like submitting the job. It is used by clients through a SOAP access to monitor the execution of a job running in a remote queuing system. Basically, a job submission service returns a unique job identifier (Job ID from the job information)

that can be used for enquiry about the job status. If the job is submitted to a batch scheduler it is in the pending state while sitting in the queue waiting to be executed. The job may become suspended due to pre-emption mechanisms. In case of normal completion the job status is "Done", otherwise the job is "Failed". In our case, once the user's job is done, generated outputs are transferred to the data repository for allowing the user to access using the file transfer service.

The input parameters to this service consist of the user's job ID, the job output directory name, and the host name of the computational resource, such as, "SDSC", "NCSA". The service implementation is designed as a persistent data factory so that retrieval for a particular job can be performed easily without invoking the grid service depending on the job status. First of all, the user-saved job information is retrieved from the job database. If the job status column indicates "Done", the data are wrapped into URLs and returned to the client immediately. If the job status is not "Done", then a job RSL script, which specifies the execution of the job status, will be created to run the Globus job, integrating with Application Information Web Service [23]. For redirecting the standard output/error for the job running from the remote machine, the GASS server is started. The GASS URL is added into the generated job RSL script as the stdout/stderr parameter. Then the generated job script is submitted to the targeting gatekeeper (GRAM server).

The output results about the job status of the scheduler are transmitted to the GASS server that is running locally, and are then parsed for getting the job status. If the job is not completed yet, the job status of the particular submitted job is updated and saved into the job database. Otherwise, the user job's output files are transferred from the remote host, and the job status is updated and stored in the job database. For doing that, there is one more additional step. The job RSL script is generated for checking the list of the file names in the remote job directory, and submitted to the destination host (GRAM server) again. The output results which contain the list of the output file names are parsed to get the used XML file name and the generated output file names. This XML file is retrieved from the job data repository, and then unmarshalled into e3d's input XML schema [18] to extract the pieces of output file names. Based on that, the prefixes of the output file names are created for obtaining the targeted output files. This generated output list is used for transferring the output files to the job data repository via the GridFTP service. On the completion of file transferring, the URLs of the output files are returned to the clients for downloading.

4 Data Management

4.1 Data Model Service

SYNSEIS application allows users to select a region of interest on an interactive United States map and extract related geological data model which contains Moho and sediment thickness data via a Web service invocation which is remotely accessed through SOAP messages over HTTP. We have defined a WSDL interface for setting the boundary with geographic coordinates (longitude, latitude). This service may load

the whole US Moho or sediment data and then calculate the location using the bounding box data provided by the input parameters. The selected region's data for the data model is returned to the users to construct a 3D geologic model. We implement this service in Java. Typically we use it to take a subset of the whole data, keeping the same data format. The different data models provided by the different institutions are also available to this service. In the SYNSEIS user interface, each data model can be viewed as a map after receiving the data model from the Web service.

4.2 Data Archival Service

IRIS (Incorporated Research Institutions for Seismology) is a university research consortium dedicated to exploring the Earth's interior through the collection and distribution of seismological data. IRIS DMC receives earthquake and seismic data from a variety of Data Collection Centers and is responsible for the long term archive and distribution of all IRIS generated data [12]. For accessing those data centers, the Data Handling Interface (DHI) servers at the FISSURES Project provide a framework for seismology software and data transmission based on CORBA [24] and Java Platforms that are developed in an open and cooperative fashion [25]. There are three servers at the DHI servers: the Network Server for providing information about networks, stations, sites, channels, and responses, the Event Server for accessing the event metadata including event origins, magnitudes, predicted arrival times and channels, and the Seismogram Server for providing for retrieval of seismograms consisting of the near real-time data and the archived data.

Based on the DHI clients that are implemented using Java CORBA IDL (Interface Definition Language), implementation, and utility tools provided by the FISSURES project, we implement the data archival Web service for easily integrating and reusing any applications. This service may be built on top of the DHI clients, which access the CORBA servers. We refer to the actual service interface for manipulating the data acquisition as the data retrieval manager. This is defined in WSDL and exposes the following methods for retrieving the archived data from the IRIS DMC:

- A user can retrieve one or more station data which contain the network code, the stations begin time and end time, the station channel data (BH*) from the network server within a given bounding box area (min, max point of the longitude and latitude).
- A user can search for one or more event data that have the origin time, magnitude, depth, longitude, and latitude from the event server within the same bounding box area in a region time period.
- A user can retrieve one or more seismogram waveform data from the seismogram server using a station and event pair and a time range. Depending on the event date, the seismogram data can be retrieved from the near real-time server, or from the archived server.

The above method calls are used for internally manipulating IRIS data. Externally, the seismogram data instance is represented as a set of URL wrapper classes. We store these instances from the seismogram server persistently on the file system.

5 Conclusion and Future Directions

We presented our initial architecture of an application-specific tool called SYNSEIS which was built using Web services, relevant job services and data management implementations. The architecture is composed of three major service components. First, we provide a job submission Web Service to run the relevant applications in TeraGrid or GEONgrid using Grid technologies, especially Globus, including a file transfer. After submitting the job that contains users' input parameters, the job information is saved into the job database. This enables us to extract out the job results and check out the job status. Next, we provide a means for obtaining a job status. This job monitoring Web Service is done through the execution of the command of queuing system for checking the job status using the grid services, providing the file transfer. Once the job is finished, the job outputs are transferred to the data repository which is running on a SOAP server. Finally, we must wrap the crustal data models in useful services that can be plugged into our SYNSEIS application tool. These modules must implement a set of specified interfaces for manipulating the bounding box on the US map. We provide data archival service interfaces for retrieving real seismogram waveform data from IRIS DMC as well.

There are several possible future revisions in our architecture. The first is to provide a secure role-based authorization control to fully integrate into the GEONgrid portal. In order to allow users to submit their jobs and check the job status, SYNSEIS SWF (the file format used by Macromedia Flash) code needs to use the user proxy credential generated by the user after logging on to the portal. The second possible improvement we plan to make is in the schema of the Application Information Web Service for integrating with the job script generator. The Application Descriptor schema contains the HostBinding element that indicates the Host Descriptor, which describes the hosting environment, especially the location, type, parameters of the queuing system. In our application tool, more system commands are required for doing the job and data handling. So, for a more general way, we will extend this schema to put the system environments. The third one is to add more user capabilities in the user interface for complex earthquake simulations. Currently, we provide simply the event-station pair and point source case. We will take into consideration providing event-multiple stations situations, a line source implementation, and multiple seismogram plots. The fourth possible modification is in the nature of the implementation for the effective Web service design. As shown in Figure 1, we have designed this application tool to compose of pure Web service components with Grids technologies. The job management Web services have been implemented on top of the system level Grid services in the back-end. And the data archive service has also been built on top of CORBA services in the back-end. However, we must also explore alternative stateful Web services mechanism that will implement applications that manage the state. This mechanism would remove some of the duplicate control capabilities and communications from the SOAP server. The definition of conventions for managing state may be handled through standard ways such as WSRF (Web Service Resource Framework) [26] or OGSI (Open Grid Services Infrastructure) [27] so that applications discover, bind, and communicate with stateful resources in standard and interoperable ways.

References

1. Foster, I., Kesselman, C. (ed.): The Grid: Blueprint for a New Computing Infrastructure. Morgan Kauffmann, 1999
2. Berman, F., Fox, G., and Hey, T.: Grid Computing: Making the Global Infrastructure a Reality. Wiley, 2003
3. W.E. Johnston, D. Gannon, B. Nitzberg. Grids as Production Computing Environments: The Engineering Aspects of NASA's Information Power Grid. Proceedings 8th IEEE International Symposium on High Performance Distributed Computing, 1999
4. M.P. Thomas, J.R. Boisseau. Building Grid computing portals: the NPACI Grid portal toolkit. In Grid Computing: Making the Global Infrastructure a Reality, F. Berman, G. Fox, and T. Hey. John Wiley & Sons, Chichester, England, March 2003
5. K. Schuchardt, B. Didier, G. Black. Ecce--a problem-solving environment's evolution toward Grid services and a Web architecture. Concurrency and Computation: Practice and Experience, Vol. 14, No. 13-15, pp 1221-1239 (2002)
6. D.W. Erwin. UNICORE—a Grid computing environment. Concurrency and Computation: Practice and Experience, Vol. 14, No. 13-15, pp 1395-1410 (2002)
7. Gateway Project. See http://www.gatewayportal.org
8. NMI Portal – Open Grid Computing Environment (OGCE). See http://www.collab-ogce.org/nmi/index.jsp
9. GEON (Cyberinfrastructure for the Geoscience) project. See http://www.geongrid.org
10. Larsen, S.: e3d: 2D/3D Elastic Finite-Difference Wave Propagation Code. Available from http://www.seismo.unr.edu/ftp/pub/louie/class/455/e3d/e3d.txt
11. Allaire, J.: Macromedia Flash MX—A next-generation rich client. March 2002. Available from http://www.macromedia.com/devnet/mx/flash/whitepapers/richclient.pdf
12. IRIS. See http://www.iris.washington.edu
13. GEONgrid portal. Available from https://geon01.sdsc.edu:8282/gridsphere/gridsphere
14. Teragrid project. Available from http://www.teragrid.org
15. I. Foster, et al. The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. See http://www.globus.org/research/papers/ogsa.pdf
16. Graham, S. et al.: Building Web Services with Java. SAMS, Indianapolis, 2002
17. Allaire, J.: Macromedia MX: Components and Web Services. April 2002. Available from http://www.macromedia.com/devnet/mx/coldfusion/whitepapers/components_ws.pdf
18. Input XML Schema for the e3d application. Available from http://geon01.sdsc.edu:8484/GEONappws/ApplSchema/event_sim_input_new.xsd
19. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Service Description Language(WSDL) version1.1. W3C Note 15 March 2001. See http://www.w3c.org/TR/wsdl
20. Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., Nielsen, H. F.: SOAP Version 1.2 Part 1: Messaging Framework. W3C Recommendation 24 June 2003. Available from http://www.w3.org/TR/soap12-part1
21. Java CoG Kit. Available from http://www-unix.globus.org/cog/java/
22. RSL v1.0. See http://www.globus.org/gram/rsl_spec1.html
23. Youn, C., Pierce, M., and Fox, G., "Building Problem Solving Environments with Application Web Service Toolkits" ICCS 2003 Workshop on Complex Problem Solving Environments for Grid Computing, LNCS 2660, pp. 403-412, 2003
24. Object Management Group. Available from http://www.omg.org
25. Fissures project. Available from http://www.seis.sc.edu/software/Fissures/
26. The WS-Resource Framework. Available from http://www.globus.org/wsrf
27. Open Grid Services Infrastructure Working Group (OGSI-WG). Available from http://forge.gridforum.org/projects/ogsi-wg