Developing Metadata Services for Grid Enabling Scientific Applications

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. In a web-based scientific computing, the creation of parameter studies of scientific applications is required to conduct a large number of experiments through the dynamic graphic user interface, without paying the expense of great difficulty of use. The generated parameter spaces which include various problems are incorporated with the computation of the application in the computational environments on the grid. Simultaneously, for the grid-based computing, scientific applications are geographically distributed as the computing resources. In order to run a particular application on the certain site, we need a meaningful metadata model for applications as the adaptive application metadata service used by the job submission service. In this paper, we present how general XML approach and our design for the generation process of input parameters are deployed on the certain scientific application as the example and how application metadata is incorporated with the job submission service in SYNSEIS (SYNthetic SEISmogram generation) tool.

1 Introduction

The Grid project [1] is essentially a giant research effort among loosely federated groups to build a seamless computing infrastructure for distributed computing, for example, the cyberinfrastructure for geosciences, GEON [2], national computational grids, TeraGrid [3]. The ultimate vision of Grid computing is that it will be able to provide seamless access to politically and geographically distributed computing resources: a researcher somewhere potentially has access to all the computing power he/she needs to solve a particular problem (assuming he/she can afford this), and all resources are accessed in a homogeneous manner using Grid technologies [4].

In a grid computing environments, scientific applications are deployed into the Grid as resources. For interacting with applications on the Grid, the information of actual applications is necessary. For example, an actual application may be wrapped by XML objects as the application proxy, which can be used to invoke the application, either directly or through submission to a queuing system without modifying the application source code. Thus, we need a general purpose set of schemas that describes how to use a particular application. Having a general application description mechanism allows

* This research was funded by NSF grants EAR-0353590 and EAR 0225673.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 379 – 386, 2006.

user interfaces to be developed independently of the service deployment. And the application metadata may be discovered and bound dynamically through XML database for storing and querying. For this, we reuse and extend the Application XML Descriptors developed by the Community Grids Lab, Indiana University, Bloomington (More detailed description is available from [5]) as the information repository.

Going beyond simple submission and management of running jobs, for the repeated execution of the same application with different input parameters resulting in different outputs, the creating process of the parameter studies manually is a tedious, time-consuming, and error-prone. Scientists want to simplify the complex parameter study process for obtaining the solutions from their applications using wide varieties of input parameter values. Those parameter studies of high-performance distributed, scientific applications have been a more challenging problem for conducting a large number of experimental simulations. The parameter spaces are related to the individual problem sizes which is obtained through several successive stages on the portal.

In this paper, we describe our designs and initial efforts for building interacting metadata services for the parameter studies and the discovery of applications in SYNSEIS application tool. Using GEON grid environments [6] and national-scale TeraGrid supercomputer centers [3] for high performance computing, the SYNSEIS application tool is developed as a portlet object that can provide the hosting environments interacting with the codes and the data retrieval systems [7]. It is built using a service-based architecture for reusability and interoperability of each constituent service components which are exposed as Web services as well. The SYNSEIS targets a well-tested, parallel finite difference code, e3d developed by Lawrence Livermore National Laboratory [8], with a wide variety of input parameters. Input file data formats in e3d application are described in the various forms depending on the user's selection. We obviously want to support a more scaleable system with common data formats that may be shared between the legacy input file data formats which the code uses. The common data format may be translated in a variety of the legacy input formats. Also, from the portal architecture point of view, it becomes possible to develop general purpose tools for manipulating the common data elements and a well-defined framework for adding new applications. Using the meta-language approach, XML, we present XML schema and our design for related data services for describing the code input parameters. This XML object simply interacts with the hosting environments of the code, converting it to the legacy input data formats, and then allowing the code to be executed, submitted to batch queuing systems, and monitored by users.

2 Related Work

We briefly review here some motivating examples for the parameter studies. Nimrod [9] a tool for managing the execution of parameterized simulations on distributed workstations and combining the advantages of remote queue management systems with those of distributed applications. This tool builds a simple description of the parameters and the necessary control scripts about a particular application for running

the code and generates a job for each set from the parameter creation. And then this job is submitted to the remote host and any required files are also transferred to the host.

Unlike Nimrod, users have the ability to access multiple job submission environments including any combination of queue systems such as PBS, LSF, Condor, and so on. In order to continue the job submission autonomously without the continued presence of the parameter study tool, ILab [10] has constructed the GUI for controlling parameter studies using the perl script and Tk tool kit. After creating input files from the parameter studies the job launching process is initiated.

Nimrod and ILab's parameter studies tool is restricted to the parameterization of the input files. On the contrary, ZENTURIO [11] uses a direct-based language to annotate arbitrary files and specify arbitrary application parameters as well as performance metrics. Using this language-based approach, a large number of experiments are potentially generated and submitted to the host.

QuakeSim project developed by the Community Grids Lab, Indiana University, Bloomington [12] is science portal for the earthquake simulation modeling codes. It provides the unifying hosting environment for managing the various applications. The applications are wrapped as XML objects that can provide simple interactions with the hosting environments of the codes, allowing the codes to be executed, submitted to batch queuing systems, and monitored by users through the browser. For support interactions between simulation codes, there are common formats, well-defined XML tags for making legacy input and output parameters using Web service approach.

3 XML Schema for Code Input

XML is an important information technology as it can build and organize metadata to describe resources and the raw data generated either by codes or by scientific instruments. This metadata will enable more precise search methods as envisaged by the Semantic Web. XML has the advantage of being human-readable and hierarchically organized, but is verbose and thus not ideal for very large datasets. Instead, it is more often useful to have the XML metadata description point to the location of the data and describe how that data is formatted, compressed, and to be handled. It may also be transmitted easily between distributed components by using Web service. XML may be used to encode and provide the data structure to code input data files in a structured way. We may thus be quite specific about which definitions of location, resolution, geology, the external source, surfaces, volumes, stations, layers, or other parameters we are actually using within a particular XML document. We do not expect that our definitions will be a final standard adopted by all, but it is useful to qualify all our definitions.

When we examined the inputs for the e3d application which is using in SYNSEIS tool currently, it became apparent that the data may be split into two portions in Figure 1: the code-basic data definitions, and code-optional data definitions for incorporating various code parameters, such as number of stations, and layers. We highlight the major elements here. We structure our XML dialect definitions as being composed of the following:

- Grid Dimension: describes the location, dimension, and the grid spacing for the grid.
- Time Stepping: includes the number of time steps and the time step increment.
- Velocity Model: includes various parameters needed to characterize the griddled velocity model such as p, s, r, or attenuation coefficients.
- Source: includes type, location, amplitude, frequency, fault parameters.
- Seismogram Output: includes location, output name, and mode for writing the seismogram in SAC format [16].
- Image Output: includes the number of time steps for producing a series of mapview images through the surface grid nodes.
- Volume Output: includes the number of time steps for outputting individual data volumes at selected time steps.
- Layer: includes parameters for the crustal model format.

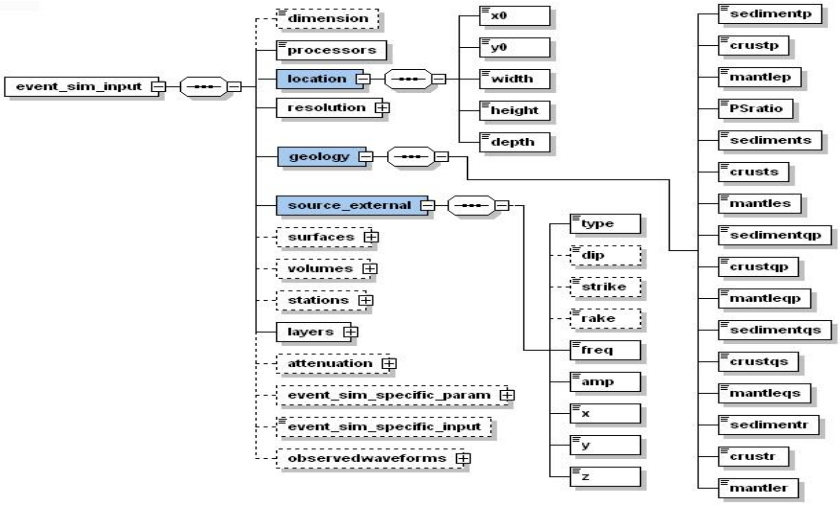

Fig. 1. The main block diagram of the e3d XML Schema

Note that we are not modifying the codes to take directly the XML input. Rather, we use XML descriptions as an independent format that may be used to generate input files for the codes in the proper legacy format. This XML schema simply defines the information necessary to implement the input files in a particular application.

4 Implementing User Interfaces

In the design of user interface, it is very difficult to be faithful to one's functions as well as be beautiful. If the user interface design is too functional, it is stiff and formal. And if that design places great emphasis on the beauty, it is easy to be decorative as well. So, our goal of user interface design is to get the benefit at both functions and the beauty at the same time.

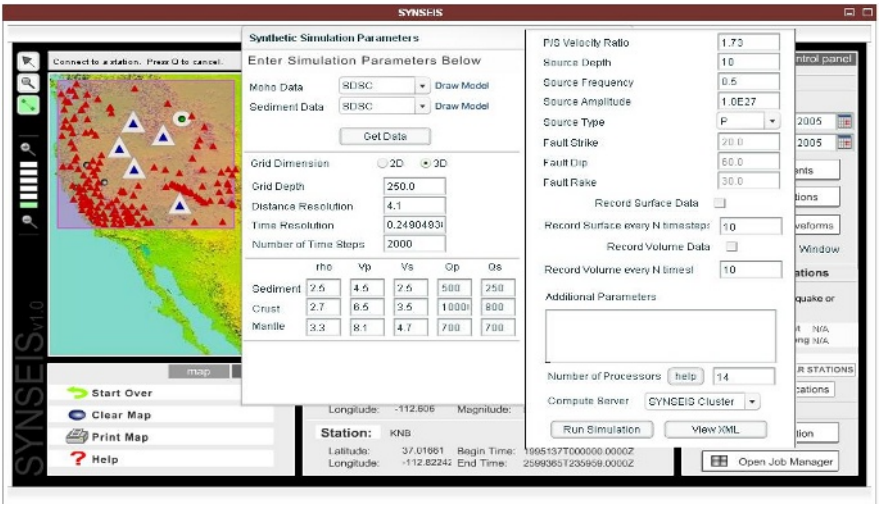

Fig. 2. SYNSEIS's interactive graphic user interface

More or less the e3d application has more complex parameter creation process. We have developed to construct our SYNSEIS Graphic User Interface using the flash application [13] as shown in Figure 2 for minimizing the difficulty of building a set of parameterized input files. Our aim in our SYNSEIS tool is also to provide users with a nice and a variety of user-centric and dynamic environments on the web for representing the input parameters of the earthquake experiments. In Figure 2, as one of scenarios, users are able to select some certain area in US map using the mapping tool and compose the location, several event points and station locations by using the data retrieval Web service [7], and the selected points for the experiment within the boundary area. And then another window specifies other input parameters to allow users to do the graphical selection of the appropriate parameter data fields and designate the set of values by using data model Web service [7]. If users set up the "Distance Resolution" field in Figure 2, "Time Resolution" and "Source Frequency" have been parameterized automatically because of unsuccessful run of the scientific program under the consideration. After the text selection of the appropriate fields, finally the XML input file is generated for running the experiments. Because the parameter creation process is integrated within SYNSEIS GUI, its use is quite easy, intuitive, and trivial.

5 Interactions of the Metadata Service for the Application

We describe how SYNSEIS system exports the XML input file to the specific input data formats of the code and integrate the metadata for the application. In Figure 3, we may express the input file for e3d application using XML format. XML generator collects and composes some data needed for running the code from the user interface, for example, map services, data model Web service, and data descriptors. Using our

specific job submission Web service [7], this generated XML input file is saved into the XML data repository for archival reference, re-use, and modification. Users may independently modify this user XML file for their own experiment purposes and resubmit it for running the application code on the archival session provided by SYNSEIS archival service for domain experts. This input XML file is therefore recyclable, if desired. For this experiment, this XML input file is transferred into the targeted remote host via the grid file transfer protocol. Simultaneously, RSL (Resource Specification Language) [14] generator creates the job script based on the application metadata for running the Globus job through the gatekeeper. This application metadata consists of mainly three parts: application descriptors, host descriptors, and queue descriptors. We use and integrate application information Web service [5] as the information repository describing e3d application for the seismic simulation and other system commands for dealing with the job files.

Through the gatekeeper, this job script that describes the remote perl script which takes XML input file as the argument is executed. When the gatekeeper runs this script on the remote host, the "Input File" generator creates several input files which are for actually running the code and the queue script in a remote certain directory for this experiment. At this stage, we must export the XML to the legacy input format for a particular application.

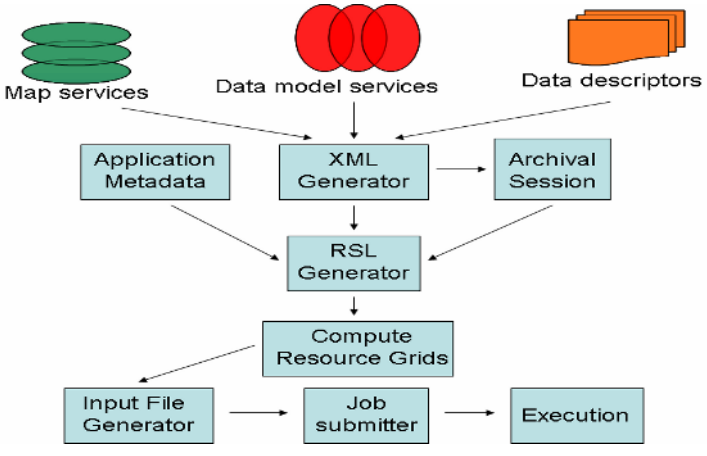

Fig. 3. Processing steps for interactions with the metadata for the application

We assume that a cluster of machines has the job scheduler such as PBS, or Condor. That is, in order for the job submitter to launch jobs run, we need the local or remote compute environments that may require any of Globus, PBS, and other job scheduler. This job scheduler which is accessed from Globus interface [15] is used for queuing and starting jobs. Finally, this batch queue script containing PBS directives followed by shell commands is submitted to the job scheduler by executing the remote perl script.

6 Conclusion and Future Directions

We have built an application-specific tool called SYNSEIS using Web services approach. Since this architecture is based on the service-oriented, those useful services can be plugged into any general frameworks and be put together on the workbench. At the implementation phase of those services, some computing services are required for integrating with the metadata services on the framework. For example, we provide a job submission Web Service to run the relevant applications in a computational and hosting environment, TeraGrid or GEONgrid using Grid technologies, especially Globus interface, including a file transfer. So, we have presented the application XML metadata for constructing the job script for running the code within this service. This application metadata for describing the actual application which we suggested can be used by application web service, having providing the application interfaces as the application proxy. 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 keep extending this schema to put the system environments.

In the design of the user interface, we have also implemented the easy-to-use parameterization process for complex earthquake simulations using SYNSEIS SWF (the file format used by Macromedia Flash) code. Through the dynamic graphic user interface, users can select and compose the data items. And then XML input data object is generated for the user experiments on the computational environments. Currently, since we provide simply the event-station pair and point source case for this experiment, we will take into consideration providing event-multiple stations situations, a line source implementation, and multiple seismogram plots additionally.

On the back end, at the time of submitting the job that contains users' XML file, the remote perl script consists of mainly two parts: the converting the XML file into the legacy input file and the job launching process. Because its process is tightly coupled, if both parts are not run successfully, for example, the converting process is done, but the job submission is failed, the remote perl script running is not useful even if the converting process is successful. For more modularity and reusability, those steps are redesigned. Creating the input files and the job directory for this experiment from the XML file will be generated by using Web service. For example, the QuakeSim portal [12] has the capability for exporting and importing the XML input file into the legacy format between applications running. Using the common XML format, QuakeSim applications are able to communicate each other via the Data hub. The current job submission is used for the job scheduler directly. That is, the job is submitted to the job scheduler directly. As the effective way, in order to get full functionalities provided by Globus interface, the launching and monitoring job will be performed through the Globus metacomputing middleware.

References

1. Berman, F., Fox, G., and Hey, T.: Grid Computing: Making the Global Infrastructure a Reality. Wiley, 2003
2. Cyberinfrastructure for the Geosciences: http://www.geongrid.org
3. TeraGrid project: http://www.teragrid.org
4. Fox, G. C., Gannon, D., and Thomas, M. A Summary of Grid Computing Environments. *Concurrency and Computation: Practice and Experience*, Vol. 14, No. 13-15, pp 1035-1044, 2002
5. 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
6. C.Youn, C. Baru, K. Bhatia, S. Chandra, K. Lin, A. Memon, G. Memon and D. Seber. GEONGrid Portal: Design and Implementations. *GCE 2005: Workshop on Grid Computing Portals* held in conjunction with SC 2005, Seattle, WA, USA, November 12-18, 2005
7. C. Youn, T. Kaiser, C. Santini and D. Seber. Design and Implementation of Services for a Synthetic Seismogram Calculation Tool on the Grid. *ICCS 2005: 5th International Conference*, Atlanta, GA, USA, May 22-25, 2005, Proceedings, Part 1, LNCS 3514, pp. 469-476, 2005
8. 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
9. Abramson, D., Sosic, R.,Giddy, J., Hall, B.: Nimrod: A Tool for Performing Parametised Simulations using DistributedWorkstations. The 4th IEEE Symposium on High Performance Distributed Computing, Virginia, August 1995 IEEE Computer Society Press, Silver Spring, MD, 1995, pp. 520–528
10. M. Yarrow, K.M. McCann, R. Biswas, R.F. Van der Wijngaart, An Advanced User Interface Approach for Complex Parameter Study Process Specification on the Information Power Grid, in Proceedings of the First IEEE/ACM International Workshop on Grid Computing, Bangalore, India, December 2000, Lecture Notes in Computer Science, Vol. 1971, Springer, London, UK 2000, 146 – 157
11. R. Prodan, T. Fahringer, ZENTURIO: a grid middleware-based tool for experiment management of parallel and distributed applications. Journal of Parallel and Distributed Computing, Vol. 64 (6), Academic Press, Orlando, FL, USA, 2004, pp. 693 – 707
12. M. Pierce, C. Youn and G. Fox, Interacting Data Services for Distributed Earthquake Modeling, ICCS 2003 Workshop on Computational Earthquake Physics and Solid Earth System Simulation, LNCS 2659, pp. 863-872, 2003
13. Allaire, J.: Macromedia Flash MX—A next-generation rich client. March 2002. Available from http://www.macromedia.com/devnet/mx/flash/whitepapers/richclient.pdf
14. RSL v1.0. See http://www-fp.globus.org/gram/rsl_spec1.html
15. Gregor von Laszewski, Ian Foster, Jarek Gawor, and Peter Lane. A Java Commodity Grid Kit," Concurrency and Computation: Practice and Experience, vol. 13, no. 8-9, pp. 643-662, 2001
16. SAC – Seismic Analysis Code: http://www.llnl.gov/sac/