iSERVO: Implementing the International Solid Earth Research Virtual Observatory

Andrea Donnellan[(1)], Geoffrey Fox[(2)], Robert Granat [(3)], Lisa Grant[(4)], Greg Lyzenga[(5)], Dennis McLeod[(6)], Shrideep Pallickara[(7)], Jay Parker[(8)], Marlon Pierce[(9),*], John Rundle[(10)], and Terry Tullis[(11)]

(1) NASA Jet Propulsion Laboratory, Mail Stop 183-335, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, USA (email: Donnellan@jpl.nasa.gov). (2) 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). (3) NASA JPL, Mail Stop 126-347, 4800 Oak Grove Drive, Pasadena, CA 91109-8099 (email: Robert.Granat@jpl.nasa.gov). (4) Environmental Analysis and Design, University of California-Irvine, Irvine, California, 92697-7070, USA (email: lgrant@uci.edu) (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) Community Grids Laboratory, Indiana University, Bloomington, Indiana 47404-3730, USA (email: spallick@cs.indiana.edu). (8) NASA Jet Propulsion Laboratory, Mail Stop 238-600, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, USA (email: jay.w.parker@jpl.nasa.gov). (9) Community Grids Laboratory, Indiana University, Bloomington, Indiana 47404-3730, USA (email: mpierce@cs.indiana.edu, phone: +1 812-856-1212). (10) Department of Physics, University of California-Davis, One Shields Avenue, Davis, CA 95616-8677 USA (email: rundle@physics.ucdavis.edu). (11) Department of Geological Sciences, Brown University, Providence, RI 02912-1846 USA (email: Terry_Tullis@brown.edu).

* Corresponding author

Abstract

We describe the goals and initial implementation of the International Solid Earth Virtual Observatory (iSERVO). This system is built using a Web Services approach to Grid computing infrastructure and is accessed via a component-based Web portal user interface. We describe our implementations of services used by this system, including Geographical Information Systems (GIS)-based data grid services for accessing remote data repositories and job management services for controlling multiple execution steps. iSERVO is an example of a larger trend to build globally scalable scientific computing infrastructures using the Service Oriented Architecture approach. Adoption of this approach raises a number of research challenges in millisecond-latency message systems suitable

for internet-enabled scientific applications. We review our research in these areas.

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 simply "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.

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 are 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.

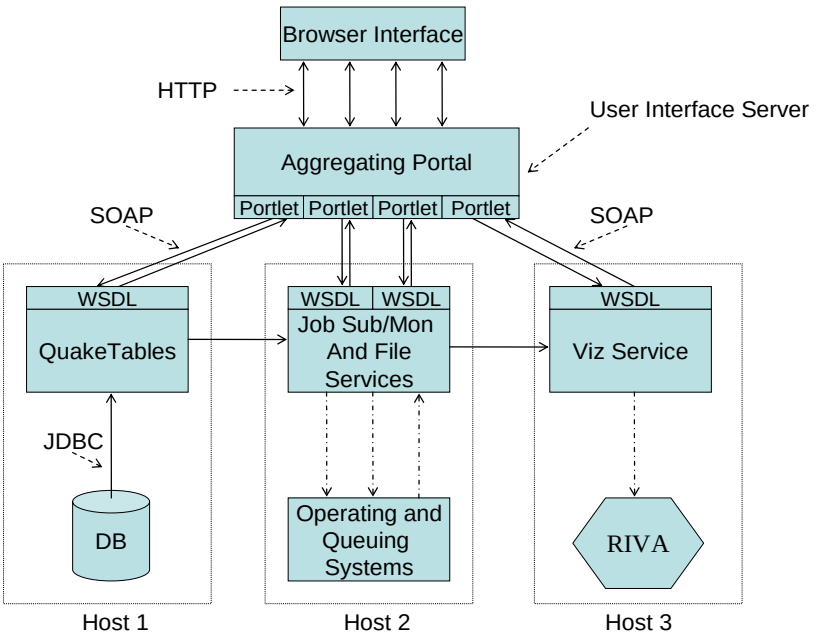

Figure 1 The architecture for the iSERO portal and services uses Web Service and portal standards.

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. We thus replace the process of downloading and hand-editing the entire catalog. Implementation details are covered in a companion ACES abstract submission (G. Aydin, et al).

iSERVO data service requirements represent an excellent opportunity for further work leveraging OGC 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.

Future Directions for iSERVO

Web Services in the SOA approach communicate with SOAP messages in a loosely coupled fashion. Such systems demand a number of features: fault tolerance, reliable messaging, message level security (such as authorization and encryption), and message virtualization for firewall tunneling. We term this general class of messages as the Web Service “Internet-On-Internet” (IOI) problem: many Web Service standards reimplement common TCP/IP features within the SOAP message. We see this as an interesting development, as it allows us to use a messaging system infrastructure (which we do control) to provide quality of service that is independent of the underlying network (which we do not control). We have implemented such a messaging system infrastructure, NaradaBrokering (http://www.naradabrokering.org), and are extending it to support Web Service invocations natively.

A common objection to the Web Service approach is that it is too slow. Message speeds across network connections are on the order of milliseconds at best, and so unsuitable for classic metacomputing. For typical iSERVO applications, this is not an issue: model runs may take several hours or days to complete. The applications themselves may be deployed on clusters or supercomputers and may be parallelized by traditional techniques; we treat such applications as a single service component [8]. However, there are classes of problems, particularly in interactive remote visualization and high performance transfers of large data sets, in which maximum network performance is

needed. We are currently researching this within the NaradaBrokering system. The IOI approach for Web Services will allow us to replace TCP/IP with much more efficient UDP transmissions, while retaining desirable TCP/IP features in the SOAP messages.

On top of the IOI infrastructure, we must provide information services: how can one encode in machine readable way what a particular service in Figure 1 actually does? What data does the service provide or require? How do other components in the system discover it? How can it be classified? How can complicated service interactions be coordinated? We term the higher level information Grid that manages this sort of information as the "Context and Information Environment" (CIE). All the information requirements that we have enumerated are part of a larger problem in metadata management. In Web and Grid Services, this is an open problem with many competing solutions. iSERVO represents a excellent test case of a real Grid with real information system requirements that can be used to validate the competing solutions.

Acknowledgments

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

References

[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).