WEB-IS (Integrated System): An Overall View

Yunsong Wang[1], Evan F. Bollig[2], Benjamin J. Kadlec[2], Zachary A. Garbow[3], Gordon Erlebacher[1], David A. Yuen[2], Maxwell Rudolph[2], Lilli X. Yang[2,] Erik O.D. Sevre[2]

[1]School of Computational Science & Information Technology, Florida State University, Tallahassee, FL 32306-4120, U.S.A.

[2]Department of Geology and Geophysics, University of Minnesota Supercomputing Institute, University of Minnesota, Minneapolis, MN 55455-0219, U.S.A.

[3]IBM, Rochester, MN 55901-7829, U.S.A.

Abstract. WEB-IS, Web-based Integrated System, allows remote, interactive visualization of large-scale 3-D data over the Internet, along with data analysis and data mining. In this paper, we discuss the overall structure of WEB-IS. Up until now we have developed three sub-modules geared towards geophysical problems. WEB-IS1 allows geoscientists to navigate through their 3-D geophysical data, such as seismic structures or numerical simulations, and interactively analyze the statistics or apply data-mining techniques, such as cluster analysis. WEB-IS2 lets a user control Amira (a powerful 3-D visualization package) remotely and analyze, render and view large datasets across the Internet. WEB-IS3 is an imaging service that enables the user to control the scale of features to view through interactive zooming. In the near future, we propose to integrate the three components together through a middleware framework called NaradaBrokering (iNtegrated Asynchronous Real-time Adaptive Distributed Architecture, a distributed messaging infrastructure that can be used to intelligently route data between the originators and registered consumers) without regard for time or location. As a result, WEB-IS will improve its scalability and acquire properties of fault-tolerance. WEB-IS uses a combination of Java, C++, and through the use of NaradaBrokering will seamlessly integrate the server-side processing and user interaction utilities on the client. The server takes care of the processor intensive tasks, such as visualization and data mining, and sends either the resulting bitmap image or statistical results to the middleware across the Internet for viewing. WEB-IS is an easy-to-use service, which will eventually help geoscientists collaborate from different sites in a natural manner. It will be very useful in the next 10 years because of the increasing number of space missions and geophysical campaigns.

1. Introduction

Earth sciences cover many disciplines and specialties, ranging from seismology, geo-materials and space science. The datasets in the geosciences are now growing at what seems to be an exponential pace because of large-scale numerical simulations of minerals under high temperature and pressure conditions, 3D convection in the mantle and crust, geodynamic and fault-zone dynamics, and others, which are all thermal-mechanical phenomena with multiple scales that result from strong nonlinear interactions. Besides numerical simulations there is also a strong growth in the accumulation of many data sets from improved instrumentation, for example in atomic-force microscopy, and from a deluge of data garnered from seismic surveys and satellite observations. Besides the sheer size of the data sets, large-scale multi-regional, multi-institution collaborations result in the distribution and duplication of this data around the

world. Both these trends make it necessary, but difficult, for the earth scientist to effectively and efficiently access, visualize and analyze the available data in a timely manner (Erlebacher *et al.* 2001). There is a need to develop tools that efficiently extract relevant subsets of the data or effectively display subsets of the datasets to ease their manipulation and analysis by scientists geographically distributed. These tools should include computational tools that pre-process the data into more useful forms, visualization tools to help the geologist gain insight into the data, analysis tools that can perform relevant statistical analyses and appropriate middleware (software tools) to connect them together seamlessly and smoothly (Erlebacher and Yuen 2002, 2003).

The primary goal of the WEB-IS project in the geosciences is to develop 'middleware' that sits between the modeling and/or data analysis tools and the display systems that local or remote users want to use. This middleware should be transparent to the user, yet take care of tracking, book keeping, connectivity, sharing resources around the world. A well-developed middleware infrastructure that is fault-tolerant and scalable will go a long way towards providing stronger collaboration facilities for the earth scientists. Currently, many earth science researchers handle very large datasets, in excess of a terabyte that are difficult to share, or even display, in real time with the current equipment at their disposal. Research teams in the geosciences often develop ad-hoc approaches, incompatible with each other, and often hard to use. Many would benefit from a set of tools that conform to agreed-upon standards, which would allow sharing, displaying, analyzing, mining, and streaming their datasets, without worrying about their development. The dream is to provide a core set of tools, and a robust middleware system, that will let users develop specialized plug-ins that will automatically become available to others through this "middleware". Such an approach can be adopted for many areas in geosciences, spanning from petrologic data to mantle convection images.

In this paper, we give an overall view of WEB-IS, a first step in towards our dream infrastructure. WEB-IS (WEB based Integrated System) is a Web application that allows geoscience data to be prepossessed, analyzed, visualized and shared through the interaction between several clients and a server. WEB-IS makes interactive data exploration available to the end users. Special attention is paid to ensure that large-scale datasets do not impose unnecessary bandwidth and computational bottlenecks. A detailed description of the various components developed so far is available (Garbow *et al.* 2001, 2003a-b, Yang *et al.* 2003, Wang *et al.* 2003).

Many data sets in the geosciences are by their very nature becoming increasingly distributed and decentralized due to a larger number of research groups, increased sizes of datasets, and improvements to the network (i.e., Internet-3). These terabytes of data are stored in various places, ranging from disk farms at supercomputing centers to portable hard drives. Handling large, physically distributed datasets demands the availability of expensive hardware. These datasets cannot be transmitted across the network in their entirety without paying a very large time penalty, not to mention that they may be too large to store locally. For this reason, it is often necessary to pre-process the data and only transmit a well-chosen subset to one or more compute servers to analyze or visualization servers to display. Of course, the details of where and how the data migrates should be transparent to the user. Our system tries to address this problem and by providing a transparent middleware between physically distributed visualization servers, analysis servers, storage servers and clients, so that the user can visualize and analyze the large scale datasets stored in data server from client through distributed visualization or analysis server.

We have developed three WEB-IS sub modules to provide remote visualization resources that will permit earth scientist users to interact with their data even though they may not have at

their disposal high performance software and hardware resources. The results can be displayed on client web browsers on computers that range from laptops to handheld devices. WEB-IS components allow the user to consider various scenarios when analyzing their datasets. WEB-IS1, which is presently devoted to earthquake analysis, allows users to navigate through their rendered 3-D data and interactively analyze some statistical properties of their data or apply data mining techniques, such as cluster analysis. WEB-IS2 allows users to manipulate Amira (a powerful 3-D visualization package and has been employed recently by the science and engineering communities to gain insight into their data, http://www.amiravis.com) to control remotely and to analyze, render and view large datasets through the Internet. WEB-IS3 is an Imaging Service that permits features taken from high-resolution numerical simulations or microscopy images to be selected at low resolution, and increase that resolution through zooming into the data. The three WEB-IS nodes apply to different scenarios but serve the same objective: allow the user to concentrate on the science, not on the computer science. It is therefore necessary to make these components work together. These modules are currently all supported by the two-way interaction between multiple clients and a single server. A client-server architecture seeks to separate out user functions (user interface and display) from the more computationally intensive tasks. However, because large datasets and high-performance computing resources in geosciences are distributed across the globe, this architecture is no longer adequate. Collaboration and sharing of resources requires the deployment of a middleware framework whose role is to connect datasets, servers, computational and visualization servers, transparently to the end user. There must also be a protocol available to allow clients to access these datasets by harnessing the power of these large servers. In this case, the traditional client-server model can be a performance bottleneck and a single point of failure (Berman *et al*. 2003). A better model upon which to draw for inspiration is the GRID architecture (Foster and Kesselman, 2003). GRID computing is a type of distributed computing in which a wide-ranging network connects multiple computers whose resources can then be shared by all end-users with proper authentication. In a broad sense, we consider the GRID as a type of middleware, which connects services with clients. We adopted an event brokering system, NaradaBrokering (NB) (developed at Community GRID Lab at Indiana University), which has many built-in capabilities to track information, and isolate clients from servers/services, as our middleware system.

In this paper we present our vision of how to construct a system that supports remote access, interactivity, distributed resources, along with services and clients that respectively support global and local analysis of geophysical datasets. This new version will substantially enhance the capabilities of the current version of WEB-IS components to provide a seamless integration of web services that will be functional across heterogeneous hardware and software. Through the decoupling of clients and services, the integrated WEB-IS will harness the power of middleware frameworks to enhance remote collaboration among researchers.

Our aim is to successfully overcome the limitations of thin clients, for example, handheld devices, and allow them to be used as a “powerful” and portable resource. Our hope is that integrated WEB-IS will find acceptance in the world of the geophysicist. Advanced data analysis tools under development will be made available on compute servers at universities or at National Laboratories. In the next section we present a limited review of work in this general area. The structure of the current version of the WEB-IS system will be presented in Section 3. How WEB-IS is expected to fit within the context of an integrated grid infrastructure is given in section 4. Finally, we present our conclusions and perspectives.

To help navigate the range of technologies involved in this work, we list the acronyms used in Table 1, along with URLs when appropriate.

Acronym	Definition	URL
CGI	**C**ommon **G**ateway **I**nterface	http://www.w3.org/CGI/
CORBA	Common Object Request Broker Architecture	http://www.corba.org
GUI	**G**raphical **U**ser **I**nterface	
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol	http://www.w3c.org/Protocols/
JavaGL	**Java** implementation of Open**GL**	http://www.inf.ufsc.br/~awangenh/CG/javagl.html/
JAI	**J**ava **A**dvanced **I**maging API	http://java.sun.com/products/java-media/jai/
JPEG	**J**oint **P**hotographic **E**xperts **G**roup	http://www.jpeg.org
Mesa3D		http://www.mesa3d.org/
NB	**N**arada**B**rokering	http://www.naradabrokering.org/
PIL	**P**ython **I**mage **L**ibrary	http://www.pythonware.com/products/pil/
SOAP	**S**imple **O**bject **A**ccess **P**rotocol	http://www.w3.org/TR/SOAP/
Tcl	**T**ool **C**ommand **L**anguage	http://www.tcl.tk
URL	**U**niform **R**esource **L**ocator	http://www.w3.org/Addressing/
VAN	**V**isual **A**rea **N**etworking	http://www.sgi.com/visualization/van
WEB-IS	**W**eb-**B**ased **I**ntegrated **S**ystem	
WEB-IS1	**We**b-**B**ased **I**nterrogation **S**ystem, version **1** (with links to Clustering Analysis)	http://boy.msi.umn.edu/WEB-IS/
WEB-IS2	**We**b-**B**ased **I**nterrogation **S**ystem, version **2** (with links to Amira)	http://boy.msi.umn.edu/web-amira/
WEB-IS3	**We**b-**B**ased **I**nterrogation **S**ystem, version **3** (with links to Image Service)	http://tomo.msi.umn.edu/~max/
WDI	**W**eb-Based **D**ata	http://www.csit.fsu.edu/~garbowza/WDI/index.html

WSDL	**I**nterrogation **W**eb **S**ervice **D**escription **L**anguage	http://www.w3.org/TR/wsdl/
XML	**E**xtended **M**arkup **L**anguage	http://www.w3.org/XML/

Table 1: List of acronyms used in the text

2. Related work

We now give some other examples of ongoing related efforts, which may be useful for geoscientists. Cactus (GRID book 79) is an open-source problem-solving environment designed for scientists and engineers. Cactus originated in the academic research community, where it was developed and used by a large international collaboration of physicists and computational scientists for black hole simulations (GRID book p30).

GIGAviz (http://www.voxelvision.no/) is a client-server software product developed to visualize and interpret 3D seismic volumes using a server engine and additional client modules. GIGAviz allows seamless interchange of data between applications and databases. GIGAviz was used to process 3D datasets related to the oil and gas by enabling "on-the-fly" rendering, visualization and interpretation. However, GIGAViz based on the clusters of low-cost PCs is physically in a single location. This will limit its usage in some cases where data server and computation server are separated by a long distance.

Cornell's geosciences interactive information system (http://atlas.geo.cornell.edu) is a client-server software product developed as an offspring to the university's ongoing efforts to construct a geosciences information system for research purposes. It is composed of a rich digital data library, middleware composed of various software components, and a variety of interactive Web-accessible tools for efficient utilization and visualization of the data in classrooms. This product system is Internet-based, and free to the public. However, we pay more attention to wireless clients whose use has become widespread.

wCLUTO (http://www-users.cs.umn.edu/~karypis/cluto/), developed by Professor George Karypis of the Computer Science Department at the University of Minnesota, is a Web-enabled version of a stand-alone application designed to apply clustering methods to genomic information. The user can upload the data into the clustering tool, a choice of several clustering methods can be made and configured, and then data is presented to the user in a variety of visual formats, including a three-dimensional "mountain" view of the clusters. Parameters can be quickly modified to examine a variety of clustering results, and the resulting clusters can be downloaded either for manipulation by other programs or saved in a format for publication. However, wCLUTO is based on a client-server paradigm other than the GRID and the datasets have to be uploaded from client to server. Thus, large-scale datasets must be located within a close proximity of the computational server; the bandwidth and the storage power of the server may become the bottleneck. In our proposed architecture, one dataset could be shared and processed by multiple servers, and multiple clients and/or servers could use the result. Moreover, our architecture allows tasks to be assigned dynamically. wCLUTO does not allow for this.

In 2003, Silicon Graphics commercialized remote-visualization solution called Visual Area Networking (VAN) (http://www.sgi.com/visualization/van) to advance visualization and collaboration across multiple sites, allowing scientists around the world to work cooperatively

and share visual data. VAN provides instant, visual access to large data sets without consideration for data or client location. However, it is very expensive, which prevents its widespread adoption.

Garbow and colleagues have developed a web-based interrogative system, called Web-based Data Interrogation (WDI), based on the client-server model that allows the users to analyze data remotely over the Internet (Garbow *et al.* 2001, 2003a-b). The system promotes portability and generates dynamic results on-demand. This paper discusses the evolution of this software into what we now refer to as WEB-IS.

3. WEB-IS: General Description

Currently we have developed three WEB-IS components that allow the user to analyze their geophysical data and numerical simulations for different scenarios. The overall scheme is depicted in Figure 1, where the three different situations are shown. The reason for the different packages is to provide various options for the users to obtain different results for the same or different datasets, which would meet the different questions posed. This can range from simple zooming-in to get a better resolution or statistical analysis (Garbow *et al*., 2001). More components can be added on, such as it. However, it also depends on a centralized powerful server, the software is designed to run only on SGI devoted to job-submission and monitoring of the runs as a function of convergence parameters as in electronic band-structure calculations in mineral physics or requests for webcam shots in a CAVE environment (e.g., Cruz-Neira *et al.,* 1993). The various components promote portability, dynamic results on-demand, and collaboration among researchers separated by a long distance.

3.1 WEB-IS1

WEB-IS1 utilizes a client-server paradigm to provide a tool for remote visualization and analysis of large-scale earthquake datasets. In this setup, multiple clients can harness the power of a large visualization server, which handles 3-D off-screen remote image rendering and data analysis. The client component is a Java applet and is therefore platform-independent (Garbow *et al*. 2003a-b). WEB-IS1 has the ability to cluster synthetic earthquake events, visualize the results off-screen and analyze 3-D geophysical data such as earthquake data. Due to the initial development of WEB-IS1 for seismic events, only earthquake data is currently available for users to examine. In the future we hope to provide users the option for uploading their own data, such as GPS campaigns along fault zones and geomagnetic events.

A single copy of the dataset is stored on the server, which acts as a data-vault. Data redundancy will be achieved in the future by maintaining multiple copies of the data on geographically distributed machines.

3.1.1 Client – Server Infrastructure

Figure 2 illustrates design of WEB-IS1, which uses a combination of CORBA, C++, Java, Python, Mesa3D library and JavaGL, a Java implementation of OpenGL (http://www.inf.ufsc.br/~awangenh/CG/javagl.html) to provide the desired connectivity. The user accesses a Java applet, which transmits user commands using CORBA (http://www.mico.org), a middleware protocol developed by the Object Management Group (http.omg.org), which provides interoperability between objects on different hardware platforms and across multiple different programming languages. After receiving a request from the client via the CORBA bus, the server invokes the requested method. This might include calling a data-

mining program for cluster analysis, performing data analysis, or rendering the image. The server then returns the results, or the resulting image, to the client applet over the CORBA bus.

WEB-IS1 has the ability to cluster synthetic and real earthquake events, visualize the results off-screen and analyze the results. The WEB-IS1 Server uses of C/C++ for reasons of efficiency, the Mesa3D library for off-screen rendering, and Python scripts to interact with the server machine; the Client is a Java Applet, embedded within a simple HTML web page. To date, we have three versions of the WEB-IS1 Client, each providing different communication strategies.

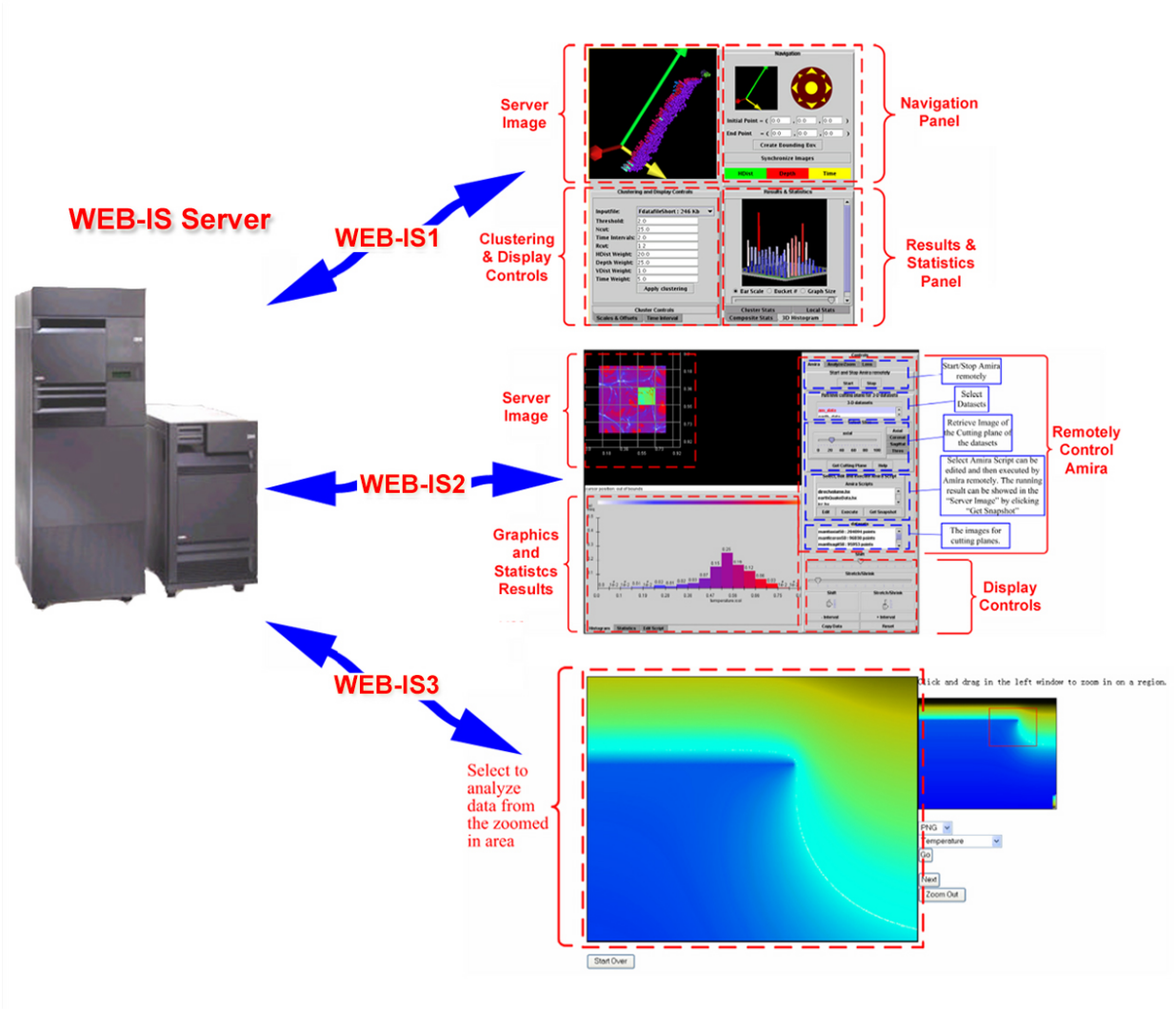

Figure 1 Schematic showing the present set up of the three modules of the WEB-IS systems applied to geophysical problems The first (WEB-IS1) handles earthquake events, the second (WEB-IS2) provides local and global statistics from convection data., and the third (WEB-IS3) is image-based and provides zooming capabilities. Additional modules, e.g., to

submit jobs and to monitor the convergence of solutions to eigensystems will be developed and integrated into the framework in the future.

The first client uses the CORBA packages provided by Java version 1.4 and higher. CORBA, a middleware protocol that was created by the Object Management Group (OMG.org), provides a direct connection between server and client, giving each client their own instance of the server process. MICO (MICO.org), a free and fully compliant version of the CORBA standard, is used on the server side, to complete the connection. CORBA provides interoperability between objects on different machines, among multiple different programming languages, and with no regard to the machine's platform. The applet communicates user interactions to the server. After receiving a request from the client via the CORBA bus, the server performs the requested operation. This can include calling a data-mining program for cluster analysis, performing data analysis, or rendering the image. The server then returns the results, or resulting image, to the client over the CORBA bus to the applet.

The second version of the WEB-IS1 Client makes use of SOAP (http://ws.apache.org/soap/) as a step towards providing compatibility with Web-Service protocols, which will help decentralize clients and servers. SOAP, or Simple Object Access Protocol, is an XML based protocol that is lightweight and ignores the notion of a central server, making it an ideal protocol for GRID or distributed computing environments (Englander 2002). Several packages and toolkits currently exist to make development of SOAP/XML services easier. WEB-IS1 takes advantage of two such toolkits: gSOAP (http://www.cs.fsu.edu/~engelen/soap.html) and Apache SOAP (http://ws.apache.org/soap/). gSOAP, created at Florida State University, was designed to optimally bind C/C++ and SOAP. Apache SOAP, developed by the Apache SOAP community, is an implementation of the SOAP v1.1 and SOAP Messages with Attachments specifications for Java. For the most part, both toolkits are interoperable, which allowed us to retain our multi-language web-service and preserve the basic structure and features found in the original CORBA system. A major difference between SOAP and CORBA can be found server-side: CORBA, connects each client to a unique instance of the WEB-IS1 server; however, only one SOAP server may run per port on a server machine. This can actually be extremely handy for collaboration aspects since two users connecting to the same server will share server ports, along with their results. Any clustering or transformation changes done by one user are available to view by all other users that are connected to the same port.

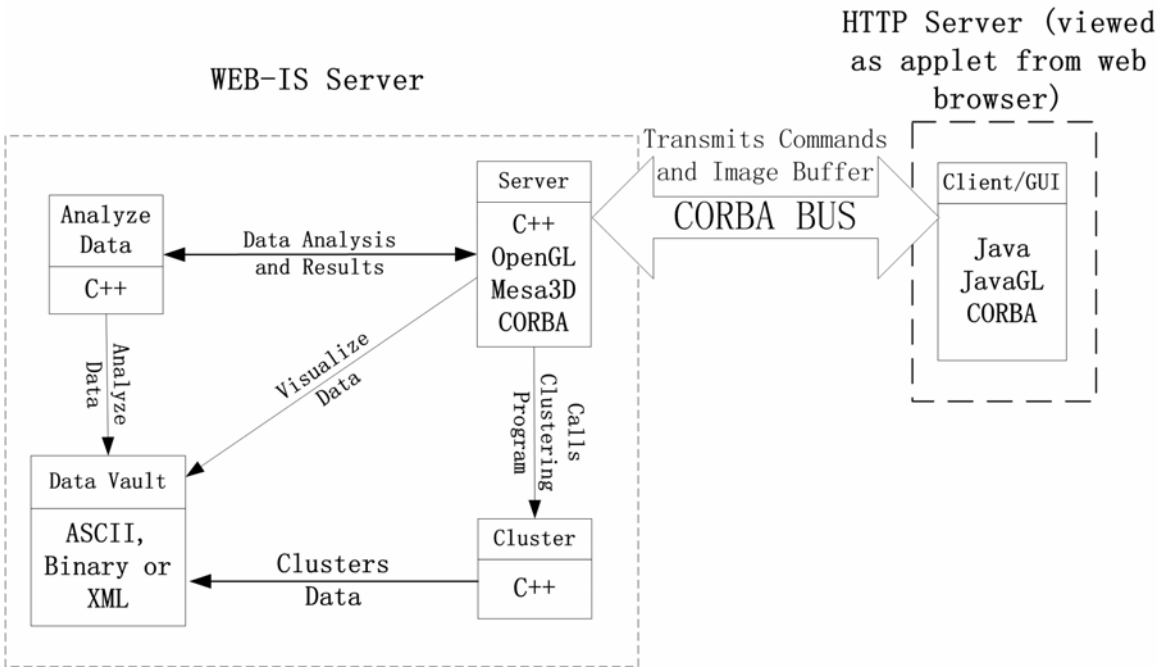

Figure 2. The user interacts with the Java applet (right), which provides the client-side GUI and a front-end for all interactions. The server (left) receives requests from the client, and performs these processor intensive tasks, returning the results to the client to be displayed. WEB-IS1 uses the CORBA as the middleware (Garbow *et al*., 2001).

The third Client version addresses the lack of sufficient resources on many clients. Our previous versions of WEB-IS1 require users to install the Java Run-Time Environment (JRE) 1.4.x for the CORBA and Swing packages. Unfortunately, at the time of this writing 1.4.x is the newest version of Java, and the many researchers in the field may not have it installed. Rather than force everyone to load the latest versions of these packages, we created a new WEB-IS1 client, following the guidelines of the previous version. The result is a Client compatible with all versions of Java starting with Java 1.1. Since the Swing packages were not always available, the new Client has a different visual appearance. However, the same primary functionality is maintained for clustering and visual data analysis. While the key components of WEB-IS1 have been included, many secondary features such as the client-side jGL implementations, time scaling, and statistics panels have been removed to make the applet more portable. The limited functionality in this applet has led us to name it the “Demo Applet”. Currently, the Demo Applet is used to introduce users to the core of the WEB-IS1 software, and let them decide whether to download the latest software packages to enable the use of current and future versions based on the latest software APIs and standards.

3.1.2 The user interface and functionality

A GUI on the client browser lets users interact with the system via icons and a pointer rather than text entry (See Figure 3). The image in the top left corner was generated off-screen using the Mesa OpenGL library on the server. Users can manipulate the data by clicking and dragging their mouse over this image.

The Navigation Panel (top right corner) acts like a remote control to the Server. It consists of a Rotator, a compass, various input fields, and two buttons (*Create Bounding Box* and *Synchronize Images*). The latter button is meant to send an aggregate set of commands to the server rather than each time the user presses a Navigation Panel component. All controls operate on the small image to the left of the compass, providing users with a real-time interactive display of how their data will appear.

The Clustering & Display Controls in the (bottom left corner provide the user with finer control over algorithm and navigation parameters. It is in this panel that users manipulate clustering parameters, adjust the axis scales, and/or broaden or narrow the time interval over which the clustering algorithm is applied. The bottom right of the applet displays various results depending on the selection at the bottom of the panel: statistics related to the clustering operation (*Cluster Stats* tab), the clustered dataset overall (*Composite Stats* tab), or statistics for the events inside the bounding box (*Local Stats* tab). One can also display a 3D Histogram (*3D Histogram* tab) that displays data in a 3D coordinate system as Depth versus Horizontal Distance versus Frequency.

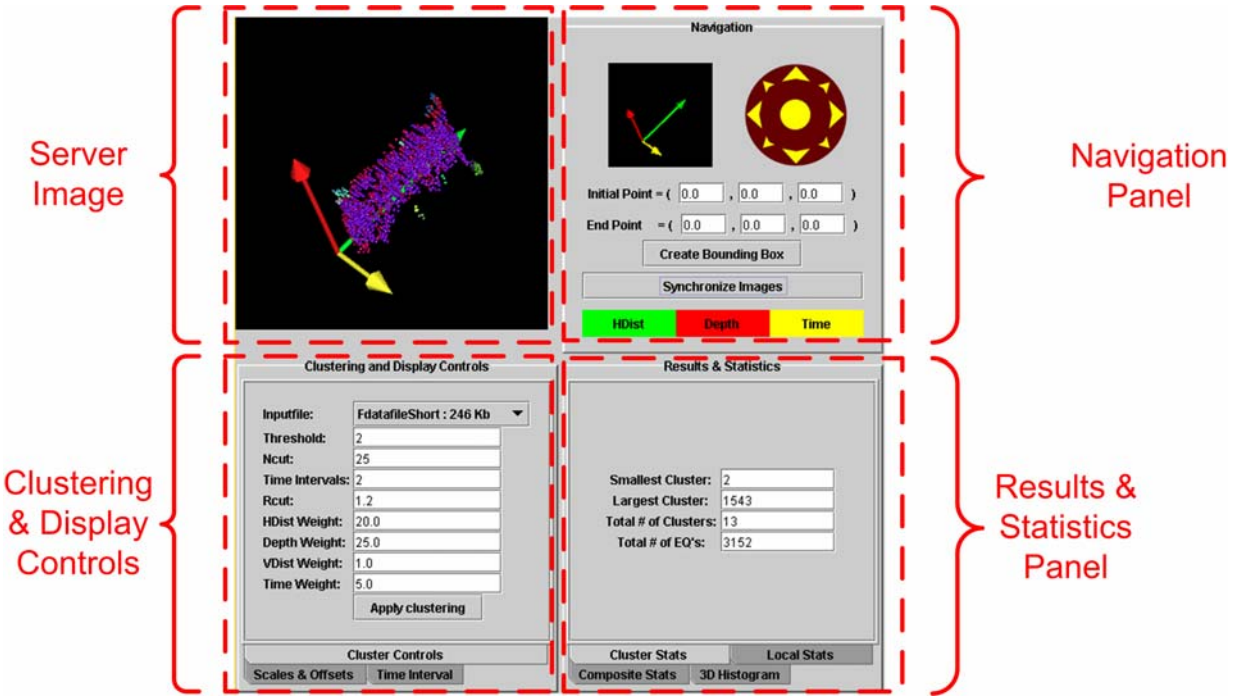

Figure 3 WEB-IS1 (http://boy.msi.umn.edu/WEB-IS) allows users to navigate through their rendered 3-D earthquake data and interactively analyze the data for statistics or apply data mining techniques, such as cluster analysis.

3.1.3 Scenarios (Examples)

WEB-IS1 is ideal for visualizing datasets that describe events, such as earthquakes, which can be displayed as single data points. To take full advantage of the available 3-D visualization facilities, each event should be described by at least three parameters. Since WEB-IS1 uses cluster analysis to identify existing patterns among the events, these should have some degree of non-zero correlation (Garbow *et al*. 2001). The homegrown clustering algorithm used

by WEB-IS1 was written for datasets that contain a time dimension. Our current research seeks to incorporate a generic clustering toolkit, wCLUTO (Rasmussen *et al.* 2003) that would allow for more general types of events. Mesa3D (http://www.mesa3d.org/), an open source alternative to OpenGL, was used to generate three-dimensional displays of the clustered events. The benefits of viewing data in 3D are self-evident especially when analyzing physical data composed of tens and hundreds of thousands of events. Even better would be the use of stereographic projection, which will be considered in the future. All images displayed by WEB-IS1 are rendered off screen on the server. As a result, any computer that supports the Mesa library can act as the server. This capability has led us to explore the deployment of WEB-IS1 within the context of GRID architectures.

WEB-IS1 was originally developed to explore and analyze synthetic seismic events (Dzwinel *et al.* 2003, Garbow *et al.* 2003b, Yuen *et al.* 2003). Each event has two spatial dimensions (horizontal location and depth) and two dimensions for time and magnitude. Although the dataset is four-dimensional, only the two spatial dimensions and time are displayed in a 3D reference frame. Each event is represented as a sphere, larger diameters representing stronger events. WEB-IS1 clusters the data before rendering. Each cluster receives a unique color, which is associated with each of its events. The format of the data as a four-dimensional dataset conveniently fits the number of properties that WEB-IS1 is able to display.

Recently WEB-IS1 was enhanced to handle real datasets from Japan (Ito and Yoshioka, 2002). These seismic events were recorded with three spatial coordinates (horizontal, vertical, and depth distances) in addition to time and magnitude. The complete record of spatial coordinates provided the unique opportunity to visualize realistic data in WEB-IS1. However, the Japanese dataset contains more dimensions than did the synthetic data. Consequently, we were forced to rethink how to represent the data visually. As a first step, the data was plotted in a Cartesian system whose axes correspond to each set the three spatial coordinates. The magnitude was again correlated to the sphere size, and each sphere had the color associated with its containing cluster. Unfortunately, time was left out.

To visualize the progression of time required a new approach. We set up WEB-IS1 to visualize according to a user-selected time interval. Only the data that occurs within the selected interval is displayed. Users can reduce the time interval down to a single second or expand it to include multiple years. Like frames in a movie reel, users are able to select smaller time intervals to see data occur in slow motion or larger time intervals to increase the speed of the events.

WEB-IS1 is currently undergoing a transformation from a static form of visualization to a dynamic form. Under the static visualization all data dimensions are directly mapped to the number of dimensions that WEB-IS can handle visually (i.e. visualizing synthetic data). Adapting WEB-IS1 to the Japanese data is an example of how the program is upgrading to a dynamic level, where the software handles data characterized by a variable number of attributes. The idea is to have WEB-IS visualize the data with a default dimension mapping of characteristics to spatial dimensions, but then allow users to change which dimensions they see. Datasets could have seven or more dimensions and at any given time users will be able to visualize four or five of them. This method of visualizing data has good and poor attributes. While a display of four or five dimensions greatly reduces the amount of data displayed, at the expense of losing information. Possibly important correlations between dimensions might be missed. One the other hand, it is extremely unlikely that the correlations will occur over more than 4-5 dimensions at the same time. If the user is able to choose the mapping, this disadvantage

is not expected to be severe. We are also investigating the possibilities of using WEB-IS1 for data-analysis outside the geological and geophysical scope, such as in the bioinformatics field.

3.1.4 Disadvantages

WEB-IS1 uses a clustering algorithm developed in-house. Although rather efficient on a single processor, the performance of the algorithm would be greatly enhanced through parallelization. The integrated WEB-IS address focus this issue by assigning multiple servers the task of clustering the data. Task scheduling will be handled through NaradaBrokering. Currently, only the original version of the WEB-IS1 client uses CORBA rather than SOAP as a remote calling protocol. Because CORBA must maintain state among its connections, its scalability does not rival that of SOAP, which is a stateless protocol. Moreover, it is more complex than SOAP (Livingston 2002). CORBA enables resource sharing within a single organization while SOAP aims to bridge the sharing of resources among disparate organizations possibly located behind firewalls (van Engelen *et al.* 2003). The integrated WEB-IS will use SOAP as the protocol to allow clients to tap into server resources possibly located behind a firewall, and make the power of these large servers more widely available through the use of Web services. This is discussed further in section 4.

3.2 WEB-IS2

WEB-IS2 is a web-based interactive application that allows users on a client machine to remotely control, display, and share visual information using Amira located on a remote server. The GUI (see Figure 5) provides a convenient front-end to facilitate the remote use of Amira assuming the accessibility of a reliable and speedy network. The only requirements imposed on the client are that the browser be Java 1.4 enabled, which is easily satisfied. Consequently, WEB-IS2 can be used on almost any platform.

Amira is a 3D visualization and modeling system that allows researchers to visualize scientific datasets from various application areas, e.g., medicine, biology, chemistry, physics, engineering, and the geosciences. Amira uses many advanced algorithms that capitalize on the hardware available on many of the current commodity graphics cards. Users construct programs visually with flow charts of modules that represent tasks or via Tcl scripts (Tool Command Language).

WEB-IS2 allows users to harness the visualization power of Amira and the associated remote graphics hardware over the Internet, avoiding the use of customized visualization software. The current version of WEB-IS2 provides two demo datasets from numerical simulations of mantle dynamics to illustrate some its basic features. One dataset was generated through numerical simulations of 3D thermal convection in the Earth's mantle (Dubuffet *et al.* 2001) while the second dataset is composed of synthetic earthquake events.

3.2.1 Client – Server Architecture

Figure 4 shows the architecture of WEB-IS2. Multiple clients have the ability to connect to a server on which Amira is executed. The client interfaces with the server through the intermediary of CGI (Common Gateway Interface) scripts. The CGI is a standard for interfacing external applications with an information server. CGI scripts execute in real-time and output dynamic information. They are invoked by an HTTP server, executed, and output is returned to the client. During execution, a CGI script sends commands to service programs, such as Amira, retrieves their output, and returns them to the client. Clients communicate with the CGI scripts

on the server through input parameters encoded into a URL (Uniform Resource Locator, http://www.w3.org/Addressing/).

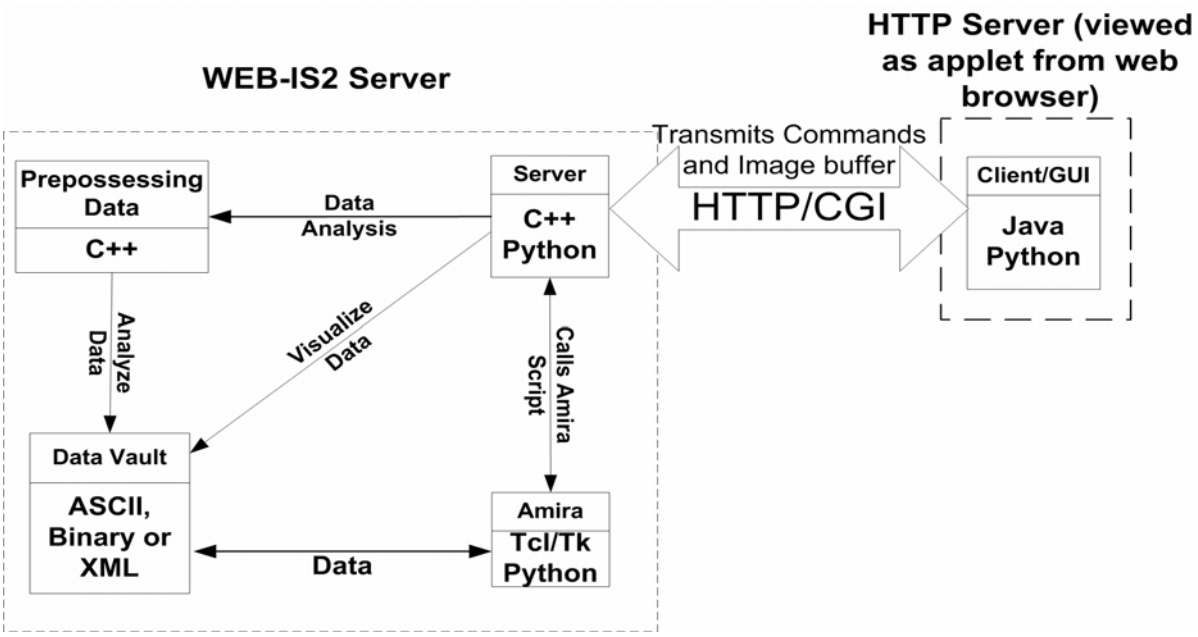

Figure 4. The user interacts with the Java applet (right), which provides the client-side GUI and a front-end for all interactions. The server (left) receives requests from the client via CGI(Common Gateway Interface, http://www.w3.org/CGI/), prepossesses data and then sends request to Amira, Amira performs these processor intensive tasks and generates the image mapped to the datasets as client's request, returning the results to the client to be displayed.

3.2.2 User interface and functionality

WEB-IS2 allows users initiate and terminate Amira on the server by interacting with the client GUI. Once initiated, an Amira script is loaded and run on the Amira server. The user can select the desired dataset and visualize the cutting planes in the *Server Image* section of the GUI. He can also select and edit a particular Amira script file to run on the server. The results of running the script on the server can be checked by capturing a snapshot of the viewer, which is displayed in *Server Image*.

Currently, Amira is only used to take cuts of a 3D dataset along the *x*-*y*, *x*-*z*, or *y*-*z* coordinate directions, using the Amira's ortho module. The data are assumed to be defined on a Cartesian mesh. Alternative cutting options on more general grids are easily implemented since Amira already provides appropriate routines. A master CGI script controls the entire process.

WEB-IS2 makes extensive use of Python (http://www.python.org), Tcl and C/C++. CGI scripts are written in Python, the message communication between the Common Gateway Interface (CGI) and the Amira server are written with C++; and the communication program interfaces to Amira through the intermediary of Tcl scripts. (The Amira console only accepts external commands written in Tcl.) Statistical processing and other analysis routines are written in C or C++ to maximize performance.

The client interface is a Java applet; the images are two-dimensional bitmaps sent from the server in jpeg format. Consequently, the requirement for client-side storage and network resources are relatively low, which is allows the use of client handheld devices.

3.2.3 Disadvantages

An important restriction on the use of our system is the current lack of off-screen rendering capability for the Amira software package. Therefore, the server must be a dedicated machine, lest there be interference when accessed by a local user. Furthermore, the rendering appears on the screen, which necessitates that a user be logged onto the server console to ensure that Amira can be activated remotely. Recent research has demonstrated the possibility of off-screen rendering of graphics programs without the need to modify their source code (Stegmaier *et al.* 2002).

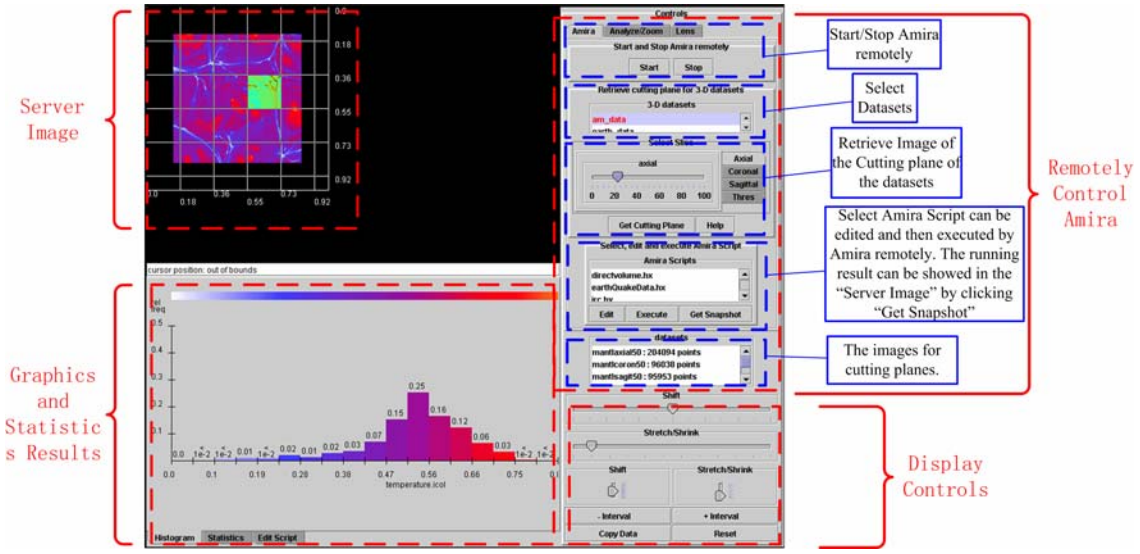

Figure 5. WEB-IS2 allows user to manipulate Amira controls remotely and to analyze, render and view large datasets through the Internet. The datasets is taken from strongly time-dependent 3-D convection in a Cartesian domain (Erlebacher, Yuen and Dubuffet, 2002)

WEB-IS2 can activate multiple Amira instances from one or multiple client browsers and control them individually, although all copies of Amira are currently instantiated on a single server. The port number associated with a particular Amira instance identifies the particular session. However, when multiple clients send different commands to the Amira server the displays will interfere with each other unless precautions are taken. Off-screen rendering and Grid services are two possible approaches to handle this problem. If the Amira server is implemented as a Grid service, the actual computations could be distributed across several computers, the results collected, and finally returned to the client. This approach would require the clustering algorithms and visualization techniques to be parallelized, resulting in a faster and more powerful virtual server.

In our current implementation, animations are only possible through repeated snapshots of Amira's viewer window. The ability to get the snapshot of the running script in real time will be

enhanced in the future version. Note that real time displays can be generated with the techniques described in Stegmaier *et al.* (2002). Finally, the GUI requires simplification.

3.3 WEB-IS3

Increases in the computational power of modern supercomputers have led to an increasing gap between the resolution of digital simulations and the resolution of conventional display devices. This problem is compounded by the increasing size of datasets from simulations and the bandwidth constraints of the Internet. In recent simulations, we have modeled subduction zone dynamics (Gerya and Yuen, 2003) using approximately 100 million markers that track data across many different fields such as temperature, viscosity, density, and chemical composition. See also Rudolph *et al*. (2003, 2004) for technical details concerning the visualization and zoomed-in techniques. Because commercially available software accessible to us could not handle these datasets, we developed a novel set of tools tailored to high-resolution, multi-variable, multi-scale simulations. These tools can be adapted to other applications with large datasets.

To address this discrepancy in resolution we considered techniques to handle both remote-visualization and the visualization of locally stored data. Our approach to remote visualization is a web-based, Image Service (WEB-IS3) (with interactive zooming) that enables the user to explore time-dependent datasets, select variables, and control the spatial scales. At local sites, the data is visualized on high-resolution display power-walls (http://www.lcse.umn.edu) with at least 10 million pixels and the calculations are performed on multiple CPUs. The high resolution and computational speed helps improve our understanding of causal and temporal relationships between multiple physical properties.

WEB-IS3 aims to view very high-resolution images in a low-resolution environment. Images are initially loaded at low resolution. By interactively selecting a sub-area, users can zoom-in to visualize finer details. This process can be repeated until the resolution of the displayed image matches that of the original image stored on the server. WEB-IS3 is entirely based on image analysis, with no consideration for the underlying dataset, which avoids the need to load nonessential data. Furthermore, storage of the raw data is no longer required on the server. Only high the original resolution images are stored in a compressed format. The initial design of WEB-IS3 provides users with the ability to load images in JPEG or PNG format due to their high quality and strong compression properties. However, new formats can be added as necessary. They can also be applied to looking at complex phase diagrams in mineral physics, rendered by wavelets (Vasilyev *et al.*, 2004).

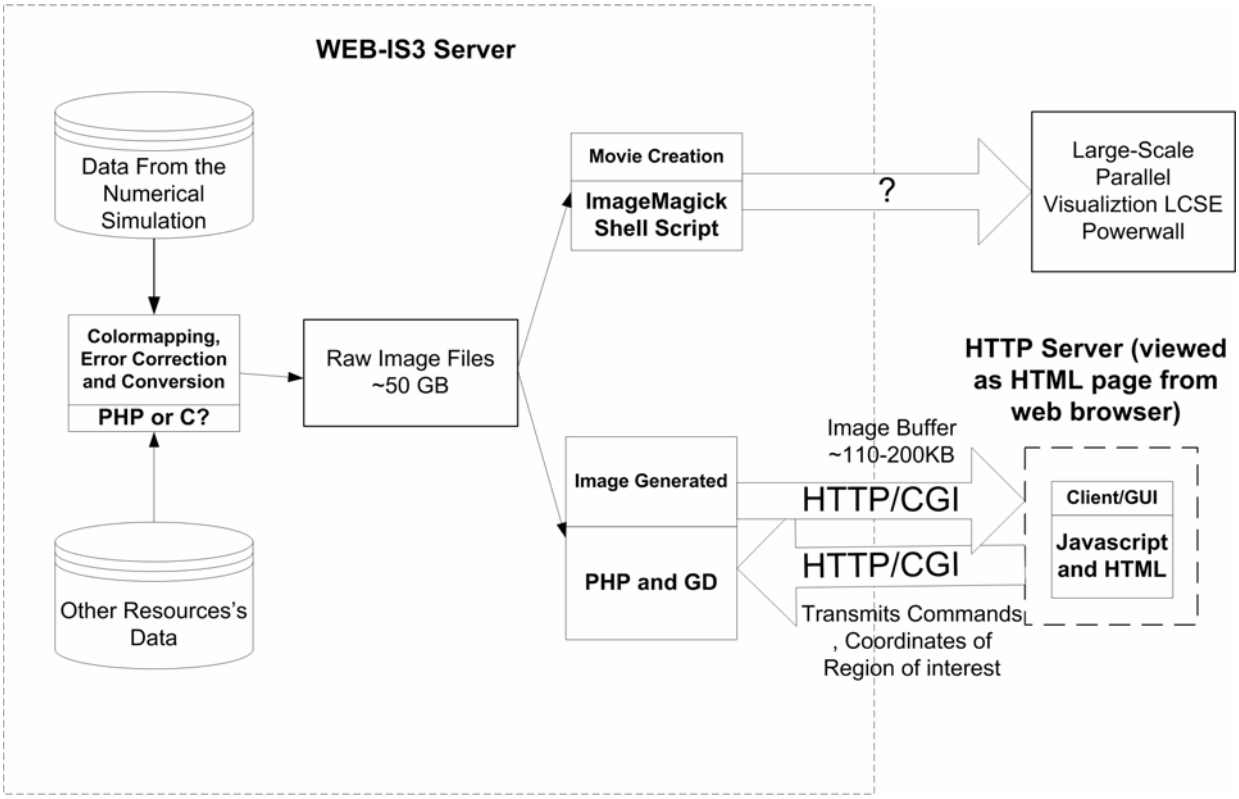

Figure 6. The user interacts with web interface (right), which provides the client-side GUI and a front-end for all interactions.

The architecture of WEB-IS3 showed in Figure 6. The server (left) receives requests from the client via CGI, prepossesses data, sends an appropriate request to computation server, which performs these processor intensive tasks, generates the image, and returns the results to either the client device of the powerwall. Through the GUI of WEB-IS3 (see Figure 7), user can repeatedly zoom further into the image, or select to analyze data from the zoomed in area. The *Reset* button resets the image scaling to the original settings.

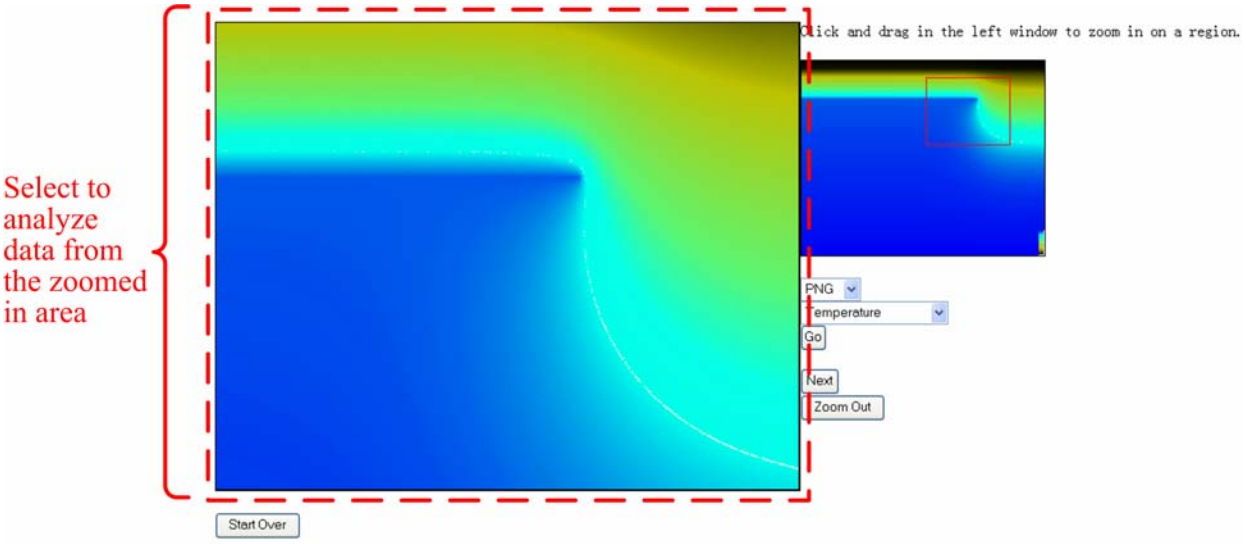

Figure 7. WEB-IS3 is an imaging service that displays selected features of subduction dynamics from a low-resolution environment to one with increased resolution by zooming into the data (Rudolph *et al*. 2003, 2004).

3.3.4 Disadvantages

The datasets source on server is a high-resolution image. While avoiding much processing, our approach also limits the ability to query data when the upper limit of resolution is reached. Furthermore, when using JPEG as a compression algorithm, distortions in the image appear at higher resolutions. For this reason, PNG (lossless compression) is preferred, although in most cases, the compression is not as effective. Of course, the need to create all the images prior to their access by the user is a disadvantage.

4. Integrated WEB-IS

In this section we present an integrated view of our WEB-IS system, sufficiently generic to be applied in many areas in the geosciences and other application areas. The integrated WEB-IS differs from previous work in that we are interested in adapting our tools to a GRID-like environment (http://www.grid.org/home.htm) to better handle distributed environments and the increased heterogeneity of the hardware/software platforms on which teams of researchers will conduct future scientific collaborations. To this end, NaradaBrokering (see next section) will permits our Client-Server computing to scale more effectively than traditional client-server systems that must expand or change a server' infrastructure in order to grow, and since both client and server subscribe to NB, decentralization is attractive with respect to scalability and fault tolerance. In spite of these assets, there are currently several drawbacks to our present WEB-IS set-up. These arise from the way we provide the three services separately. Currently, all computations are executed on a single server. Clearly, the most important problem is that as an increasing number of users access the system, it eventually becomes overloaded. It is therefore be important to distribute services across many servers and ensure that each service is multiply redundant. The same is true for large datasets: they should be stored redundantly at multiple locations (similarly to mirroring in archival sites). In the integrated WEB-IS system, The WEB-IS components (clients and servers) will be connected across a GRID. A GRID-like infrastructure can be implemented using NaradaBrokering (NB), designed to run on a large

network of brokering nodes (see next Section). While current Grid systems are mostly based on point-to-point messaging, NB connects providers and consumers of information through *topics*, while the middleware is responsible for message routing.
We briefly describe the philosophy behind NB and some of its attributes (section 4.1), and follow this by a more detailed discussion of the architecture of Integrated WEB-IS (section 4.2).

4.1 NaradaBrokering

NaradaBrokering (iNtegrated Asynchronous Real-time Adaptive Distributed Architecture) is a distributed messaging infrastructure that can be used to intelligently route data between the originators and registered consumers (Fox and Pallickara 2002, 2003). Its strength lies in the complete decoupling between message source and destination. Rather than specify the precise origin and destination of a message with an IP address, or any other tag uniquely identifying the actual hardware, messages are tagged by topic. A newsgroup server nicely illustrates the principles behind NB. Publishers of news items send information to the news server labeled by topic. Subscribers can choose to read specific news groups, and selected topics within them. A topic might contain several news items, which follow what is known as a thread. Similarly to the news server, NB is a "black box" that accepts (and stores) input messages (tagged by topic) sent by the publishers, and routes them to any end client that has subscribed to that particular topic. NB keeps track of each topic, and routes message contents to the appropriate subscribers. An important consequence of this strategy is that several publishers can send messages to, and several subscribers can receive messages from, a given topic (not necessarily the same topic). By its very nature, NB facilitates the development of collaborative systems. The functionality and scalability of NB is enhanced by forming broker networks. (A broker is any machine, connected to the Internet, on which the NB kernel is installed.) Publishers send messages through a broker originator, while subscribers receive messages from, possibly, another broker. NB takes care of ensuring that the messages requested by the subscribers are received. NB has already been applied to develop a distance education system, audio/video conferencing (Bulut *et al.* 2002), and to tunnel through proxies and firewalls. More interestingly, a client can subscribe to a topic prior to the first message published to that topic. The subscriber will then automatically receive the message when available. This feature will make it possible for various services to subscribe to topics that describe their capabilities, for example "storage". Any message published under the heading "storage" will automatically be handled by the appropriate services. Topics can have several attributes attached to them when initially defined. Message attributes control their properties: how long do they remain within NB, or are they destroyed once the first subscriber receives them. Entities within the system use the broker network to effectively communicate and exchange data with each other. These interacting entities could be any combination of users, resources, services and proxies. As mentioned above, NB has a cluster-based architecture, which allows the system to scale to support large client concentrations, addition of new broker nodes and high volume of messages between interacting clients. Source code to NB is available at http://www.naradabrokering.org.

The NB infrastructure will provide us with the ability to dynamically link WEB-IS computational servers, visualization servers, storage servers and WEB-IS clients. These will support the execution of large-scale, resource-intensive, and distributed visualization tasks through a systematic use of topics to help route messages from clients to their intended servers, *without the user having explicit knowledge of the servers where tasks will be executed.* NB-based WEB-IS will be distributed, heterogeneous and dynamic.

4.2 The integrated WEB-IS infrastructure

Currently all three components of WEB-IS execute on a single server. NB will enable the server side of WEB-IS to be installed redundantly on multiple servers throughout the world. Through the proper use of topics, tracking of server capabilities, benchmarking of network bandwidth, we expect client requests to be served by the most appropriate service. Redundant services, and decoupling the servers from the clients, will provide fault tolerance, security, and portability to our system. For example, if a particular server crashes, NB will automatically connect a client to a server with appropriate capabilities. Security features are currently being integrated into NB, which when complete, will automatically provide secure communication within WEB-IS. As for the servers, clients can connect from anywhere, thus increasing portability. In addition, a user will have the ability to disconnect from one machine, connect to another, and continue working, since all requests are controlled by topics, not by the specific hardware. Naturally, each client machine registers with NB its hardware capabilities, which will permit NB to control the flow of information to the client to avoid bottlenecks.

Our aim through integration with NB is to maintain all the current advantages of WEB-IS 1, 2 and 3, while overcoming all of their disadvantages. In the integrated WEB-IS, there will be no clear demarcation between WEB-IS1, WEB-IS2 or WEB-IS3. The computational components of WEB-IS will reside on several servers and will be interface with NB through Web services. To request a service from a particular component will require a subscription to a topic whose name is that component. In our case, WEB-IS 1, 2 and 3 provide specific services to the user, so will be accessed by using their names as topics. Services will subscribe to their own topics. For example, a WEB-IS1 server will subscribe to itself, and will therefore receive requests from any client publishing to the WEB-IS1 topic. If several servers have WEB-IS1 installed, a mechanism will be required to ensure that only a single server is activated to provide results to the client. This is the subject of ongoing research. More generally, services can invoke each other by the same topic mechanism. For example, WEB-IS3 could get a cutting plane image generated by Amira through sending a request to WEB-IS2. NB will ensure that a single sign-on will generate security credentials that can be used for the collection of actions (potentially on multiple resources) that may be needed in a WEB-IS Session. The setup of the Integrated WEB-IS, as currently envisioned, is shown in Figure 8.

SOAP was chosen to communicate between WEB-IS components and the clients. SOAP uses XML as the marshalling format and typically adopts HTTP as a firewall-friendly transport protocol. The XML-based protocol is language and platform independent, which implies that the message load can be shared among disparate parties, across different platforms, language and programming environments (Engelen *et al*. 2002). As a result, any application could use our components, as long as it implements an interface to interpret incoming and outgoing soap messages. SOAP also supports binary messages, which facilitates the transfer of, for example, JPEG images among the WEB-IS components.

However, there remains the task of interfacing SOAP with the language used to develop our web service. WEB-IS uses C/C++ to maximize performance. There are some SOAP C++ implementations that adopt a SOAP-centric view and offer SOAP APIs for C++ that require the use of class libraries for SOAP-like data structures. This often forces a user to adapt the application logic to these libraries and is clearly inconvenient when working with legacy codes, since we would like to minimize the number of modifications to them. In contrast, the gSOAP, toolkit provides a unique SOAP-to-C/C++ language binding for the development of SOAP Web services and clients and provides a C/C++ transparent SOAP API. The gSOAP stub and skeleton

compiler automatically maps native and user-defined C and C++ data types to semantically equivalent SOAP data types and vice-versa. Therefore, (mostly) full SOAP interoperability is achieved with a simple API, which relieves the users from the burden of SOAP details and enables them to concentrate on the application-essential logic. The overhead and memory usage of gSOAP is low, which makes gSOAP attractive in high-performance environments. Details can be found at (http://www.cs.fsu.edu/~engelen/soap.html).

Whereas SOAP is an actual protocol, WSDL (Web Services Definition Language) is a XML description of web services (Box *et al*. 2000). This description contains all the information necessary to enable client access to remote services. The most important information includes service location, bindings to a specific message encoding (e.g., SOAP), links between operations and one or more transport protocols (UDP, TCP, etc.). In addition, sufficient information is included about the functionality provided by the service, including method calls, parameter and return types, and complete definitions of any user-defined types. WSDL files play an important role since their information content is sufficient to construct SOAP messages that permit a client to access services remotely. The full specification is available at http://www.w3.org/TR/wsdl. WSDL is the key to building a bridge between our C/C++ SOAP implementation and other language SOAP implementations, for example Java-SOAP. WSDL files are rather complex; fortunately, gSOAP provides a WSDL generator to generate Web service descriptions. In the integrated WEB-IS, the clients are Java Applets to provide friendly, platform-independent and real-time interaction; the services are mostly coded in C/C++ for performance.

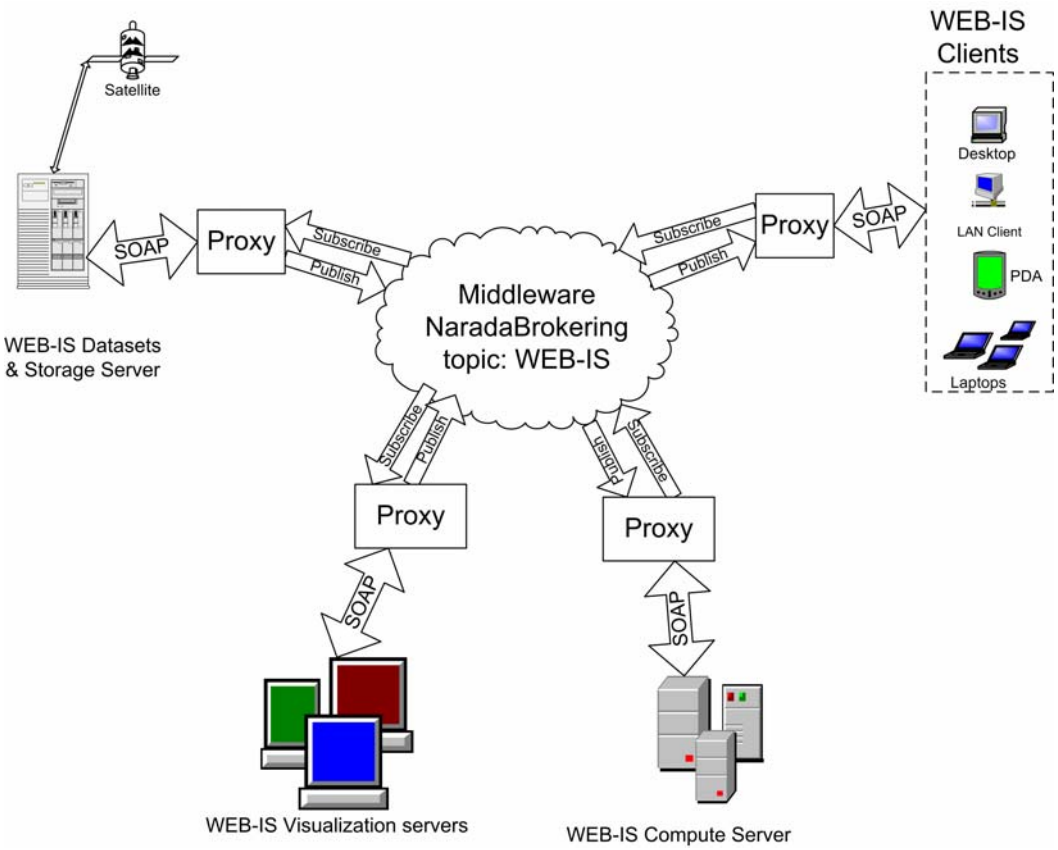

Figure 8. Integrated WEB-IS architecture. Clients (users) submit tasks (visualization, data mining, statistical analysis, file retrieval etc.) to be executed by one or several servers. The particular server is not known to the user. This is accomplished with a collection of Middleware "Brokers" (the NaradaBrokering system) that operate on the publish/subscribe paradigm. Both clients and servers publish information to the middleware, identified by one or more topics. In turn, clients and servers subscribe to these topics to retrieve the topic content. The NaradaBrokering system automatically takes care of routing publisher information to the appropriate subscribers. Information can be metadata (hardware and software resources, task descriptions) or data requested by the client (visualization data) or by the visualization server (files retrieved from data storage). One or more databases can publish information to the system (replies to system queries), or subscribe to particular topics it wishes to place in long-term storage for later retrieval (e.g., various topics that relate to metadata).

The SOAP-based communication between a Java client and a C/C++ server is as follows: On the server side, C/C++ code is compiled with gSOAP to map native and user-defined C and C++ data types to semantically equivalent SOAP data types, and to generate the WSDL to provide a Web service interface for the clients. On the Java client side, a stub that accesses the Web service is generated automatically by the utility "WSDL2Java" (http://www.systinet.com/doc/wasp_jserver/waspj/wsdl2java.html), which generates a Java stub class. This class is used to invoke the remote methods of the Web service. The compatibility between the Java class on the client side and C/C++ Web services is achieved because they are based on the same WSDL. This scheme is showed in Figure 9 as a flowchart.

NB wraps the SOAP messages received from the clients and appends a topic header. Prior to reception by the server, NB unwraps the message, and sends the SOAP payload to the service. However, there is a slight problem with this straightforward approach. Existing clients and services do not know about NB; yet NB requires a topic to forward the message to the correct destination. Our goal is to provide a facility to connect existing services to NB without change to the source code. This is possible by associating with each client and service a proxy, which simply acts as a substitute to the client/service, yet knows about NB. Therefore, the client communicates with a client proxy, while the server communicates with a server proxy. When the client proxy receives a gSOAP message from the client, the message becomes the payload of a NaradaBrokering message, which is published. Conversely, the server proxy (SP) unpacks the NaradaBrokering message, and forwards the soap message to the server.

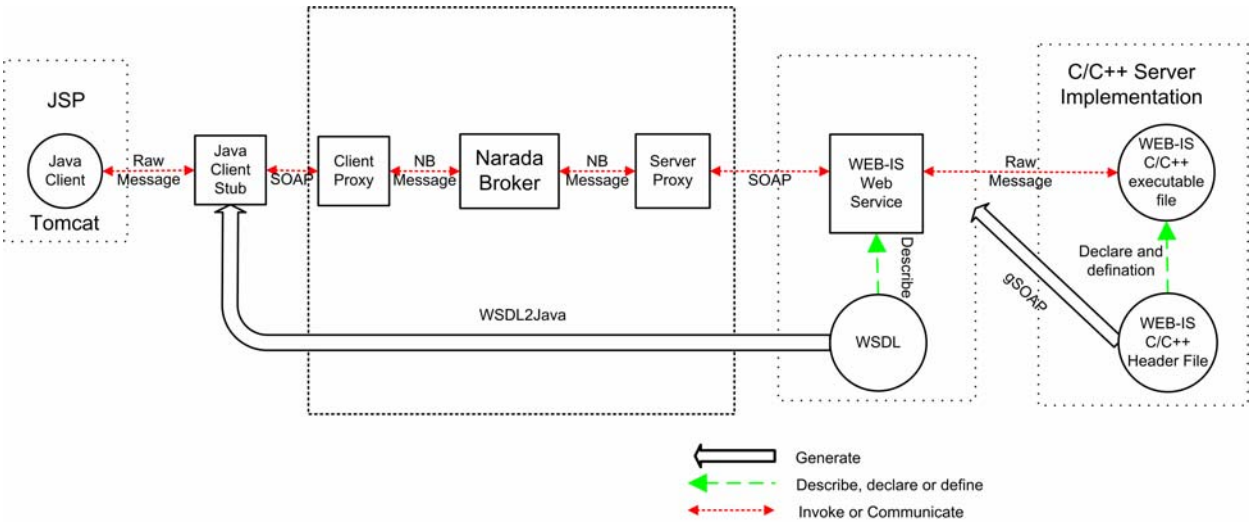

Figure 9. Java Client-NaradaBroker-C/C++ Server. The gSOAP generates platform-independent C and C++ source code for the client stubs, server skeletons and WSDL. WSDL2Java generates the Java Client Stub, which connect the Java Client with C/C++ Client Stub generated by gSOAP. Finally, Java client can communicate with C/C++ server with SOAP through NaradaBroker via the interface of the Web service.

4.3 Demo Project based on NB-gSOAP architecture and Future Implementation

To verify the feasibility of the above picture, a demo project based on Figures 8-9 was developed. In the project, a server renders an image off-screen at the request of the client and returns the images to client for display. Communication between them is channeled through NB. The client is a java applet and the server is implemented in C++. Since the client and server are resident on different machines and are implemented in different languages, some steps have to be taken before they are connected to NaradaBrokering. We also need create Client Stub and Server Skeleton to act as proxies. The actual client and server communicate with their proxies, who in turn wrap the SOAP messages (and unwrap NB messages) and send them to (and receive them from) NB.

The Java Client Stub showed in the Figure 9 is the Java definition and implementation of the Web Service, which is equivalent to the definition and implementation of the Web Service written with C++ on the server side. The utility WSDL2Java generates the Java Client Stub from the WSDL of the Web Service. The main reason for the Java Client Stub is give the client access to the Web Service Object to make remote procedure calls possible. In this setup, the client can call the “Web Service” from its code and translate the user input into the SOAP format through the support of Apache Axis. Then the SOAP messages are sent to the Client Proxy who is in charge of establishing the connection to NB and forwarding it the SOAP message with an appropriate specified topic header. NB routes the message based on the topic. The message arrives at the server proxy, which unmarshals (i.e., translates) the data to its original SOAP format, and forwards it to the server, who receives it and generates the requested image. The return data path is similar. Finally the image is returned as byte stream to the client and rendered by the Web Browser through the intermediary of the Tomcat JSP server. Fig. 10 below shows a plan of the initial configuration of our NaradaBrokering set-up to be used in WEB-IS. Three

institutions will be involved, Univ. of Minnesota, Florida State University and Indiana University.

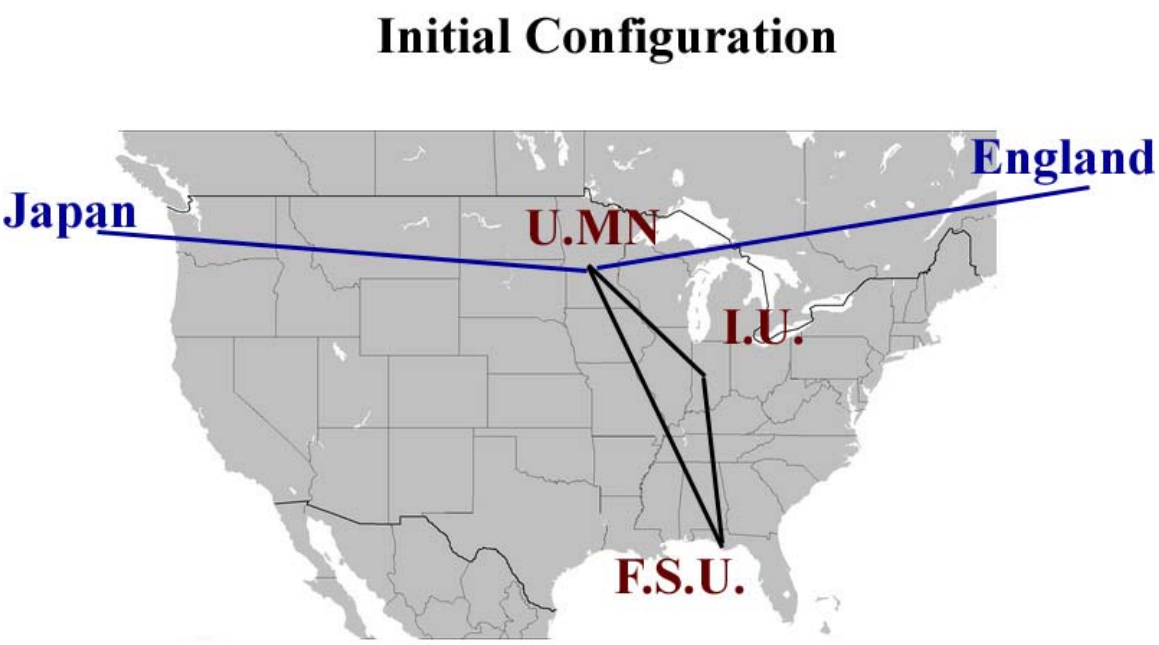

Figure 10. The NB-gSOAP architecture over the Internet connecting the three institutions, U of Minnesota (UMN), Indiana University (IU) and Florida State University (FSU)

5. Concluding Remarks and Future perspectives

Without a doubt, geoscientists are confronted with increasingly larger datasets, which appear to be growing at an exponential rate. This relentless drive forces the need for visualization, feature extraction and more compact data representations (Erlebacher and Yuen, 2003). In this paper we have focused our attention on geophysical data sets. However, we are cognizant of the potential of WEB-IS in the areas of bioinformatics and medicine. A particularly good example is the display of digital mammograms, which can benefit greatly from the use of WEB-IS. A goal of this archive is to study and understand epidemiologic issues related to breast cancer (http://sprojects.mmi.mcgill.ca/mammography/anat.htm) by searching through this vast database of high-resolution images with a 40 Mbytes capacity. Figure 11 shows the original picture of the 2D mammogram, the picture with the color map, the 3D picture with the height field or topography, and a zoomed-in view of the topography, which is similar to topography found in geological terrains (e.g., Sandwell and Smith, 1997)

Our proposed architecture, based on NB will eventually allow multiple users to collaborate visually regardless of the location of the server resources. Different users might even access different servers. However, in this case, some form of synchronization between servers would be necessary to ensure that multiple users share the same view. Users may also use WEB-IS for submitting jobs and monitoring jobs in progress. The only requirement imposed on the clients will be the availability of a browser capable of displaying Java applets. We hope that WEB-IS will soon be integrated into a GRID-based remote visualization and monitoring environment and used by the geoscientists to enhance his working experience.

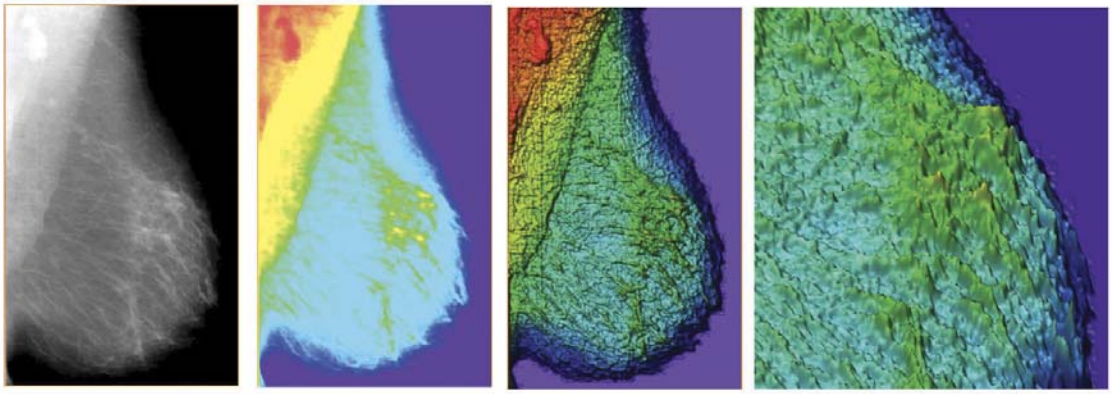

Figure 11 The original 2D picture of mammogram (left 1), the picture with colormap (left 2), the 3D picture with height-field (right 1) and zoomed-in 3D picture.

6. Acknowledgments

The authors would like to thank Cesar DaSilva for useful feedback on the contents of this paper, and fruitful discussion with Shrideep Pallickara, the author of NaradaBrokering. They also acknowledge the support of the geophysics and mathematics-geoscience programs of the National Science Foundation.

7. References

Berman, F., Fox, G., and Hey, A.J.G. (Editors), GRID Computing: Making the Global Infrastructure a Reality, Wiley Series in Communications Networking & Distributed Systems, 1013 pp., John Wiley & Sons Ltd., 2003.

Bulut, H., Fox, G., Pallickara, S., Uyar, A., and Wu, W., Integration of NaradaBrokering and Audio/Video Conferencing as a Web Service. IASTED International Conference on Communications, Internet, and Information Technology, 2002, pp. 401-406, 2002.

Cruz-Neira, C., Sandin , D.J. and T.A. DeFanti, Visualization in a Cave –environment, Proceedings of SIGGRAPH '93, pp. 135-142, 1993.

Dubuffet, F., Yuen, D.A., Murphy, M.S., Sevre, E.O., and Vecsey, L., Secondary Instabilities developed in upwellings at high Rayleigh number convection, in EOS, TRANS, AGU, 2001, Vol. 82, No. 47, pp. F210.

Dzwinel, W., Yuen, D.A., Kaneko, Y.J.B.D., Boryczko, K., and Ben-Zion, Y., Multi-resolution clustering analysis and 3-D visualization of multitudinous synthetic earthquakes, Geosciences, Vol. 8, pp. 12-25, 2003. http://link.springer.de/link/service/journals/10069/contents/tfirst.htm .

Erlebacher, G., Yuen, D.A., and Dubuffet, F., Current trends and demands in visualization in the geosciences. Electronic Geosciences 4. http://link.springerny.com/link/sevice/journals/-10069/technic/erlebach/index.htm, 2001.

Erlebacher, G. Yuen, D.A., and Dubuffet, F., Case Study: Visualization and Analysis of High Rayleigh Number -- 3D Convection in the Earth's Mantle. Proceedings of IEEE Visualization, pp. 529-532, 2002.

Erlebacher, G. and Yuen, D., A Wavelet Toolkit for Visualization and Analysis of Large Data Sets in Earthquake Research, Pure and Applied Geophysics, 2003. http://www.csit.fsu.edu/~erlebach/publications/erlebach_yuen_maui2_2002.pdf .

Foster and C. Kesselman (Eds.), The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kauffman, 2003.

Fox, G. and Pallickara, S., The Narada Event Brokering System: Overview an dExtensions. Proceedings of the 2002 *International Conference on Parallel and Distributed Processing Techniques and Applications (*PDPTA '02), CSREA Press (2002) ed. H.r. Arabnia, Vol. 1, pp. 353-359.

Fox, G. and Pallickara, S., NaradaBrokering: An Event Based Infrastructure for Building Scaleable Durable Peer-to-Peer Grids. Chapter 22 of *Grid Computing: Making the Global Infreastructure a Reality Grid.* Published by John Wiley, England, 2003.

Garbow, Z.A., Olson, N.R., Yuen, D.A., and Boggs, J.M., Interactive Web-Based Map: Applications to Large Data Sets in the Geosciences, Electronic Geosciences, Vol. 6, 2001. http://link.springer.de/link/service/journals/10069/papers/webbased/index.htm, 2001.

Garbow, Z.A., Erlebacher, G., Yuen, D.A., Boggs, J.M., and Dubuffet, F., Web-Based Interrogation of Large-Scale Geophysical Datasets from Handheld Devices, Visual Geosciences, Vol. 8, 2003a.

Garbow, Z.A., Erlebacher, G., Yuen, D.A., Bollig, E. and Kadlec, B., Remote Visualization and cluster Analysis of 3-D Geophysical Data over the Internet Using Off-Screen Rendering, in press, 2003b.

Gerya, T., and D.A. Yuen, Rayleigh-Taylor instabilities from hydration and melting propel `cold plumes' at subduction zones, Earth Planet. Sci. Lett., 212, 47-52, 2003.

Ito, T., Yoshioka, S., A dike intrusion model in and around Miyakejima, Niijima and Kozushima. Tectonophys. 359, 171-187, 2002.

Livingston, Dan, Advanced SOAP for web professionals, 331-335, 2002.

Rasmussen, M., Deshpande, M., Karypis, G., Johnson, J., Crow, J., and Retzel, E., wCLUTO: A Web_enabled Clustering Toolkit, Plant Physiology, vol. 133, No. 2, p. 511, 2003.

Robert A. van Engelen, Kyle A. Gallivan, The gSOAP Toolkit for Web Services and Peer-To-Peer Computing Networks, In Proceedings of IEEE CC Grid Conference, 2002.

Robert Englander, Java and SOAP, O'Reilly, May 2002.

Rudolph M. L., Gerya T. V., Yuen D. A., and DeRosier S., Visualization of Multiscale Dynamics of Hydrous Cold Plumes at Subduction Zones, Visual Geosciences, 2003.

Rudolph, M., Gerya, T., Yuen, D.A. and S. De Rosier, Visualization of Complex Multiscale Phenomena at Subduction Zone, Visual Geosciences, in press, 2004.

Sandwell, D.T. and W.H.F. Smith , Marine gravity from Geosat and ERS-1 altimetry, J. Geophys. Res., 102, 10,039-10,054, 1997.

Stegmaier, S., Magallon, M. and T. Ertl, A Generic Solution for Hardware-Accelerated Remote Visualization, Joint Eurographics – IEEE TCVG Symposium on Visualization, 2002.

Vasilyev, O.V., Gerya, T. and D.A. Yuen, The application of multidimensional wavelets to unveiling multi-phase diagrams and in situ properties of rocks, Earth Planet. Sci. Lett., in press, 2004.

Wang, Y., Erlebacher, G., Garbow, Z. A., & Yuen, D. A., 2003, Web-based service of a visualization package “Amira” for the geosciences, Visual Geosciences, 2003.

Yang, X. L., Wang, Y., Bollig, E. F., Kadlec B. J., Garbow Z. A., Yuen D. A., Erlebacher G., WEB-IS2: Next Generation Web Services Using Amira Visualization Package, American Geophysics Union (AGU) Fall Meeting, 2003

Yuen, D.A., Bollig, E.F., Kadlec, B.F., Dzwinel, W., Ben-Zion Y., Yoshioka, S., Data-Mining Analysis & Visualization of Earthquake Clusters in a GRID-like Interactive Environment, American Geophysics Union (AGU) Fall Meeting, 2003.