Thesis Proposal

High Performance, Federated and Service-Oriented Geographic Information Systems

Ahmet Sayar
asayar@cs.indiana.edu

1. Introduction

Geographic Information Systems (GIS) [1, 2] is basically a collection of computer hardware and software for capturing, managing, analyzing, and displaying all forms of geographically referenced data.

General purpose of GIS is extracting information/knowledge from the raw geo-data. The raw-data is collected from sensors, satellites or any other ways and stored in databases or file systems. The data goes through the filtering and rendering services and, presented to the end-users in human recognizable formats such as images, graphs, charts etc. A well-known example of GIS is map viewers which process layers of geospatial data to create map images. GIS are used in a wide variety of tasks such as urban planning, resource management, emergency response planning in case of disasters, crisis management and rapid response etc.

Over the past decades, GIS has evolved from traditional centralized systems to distributed systems [3]. Centralized systems provide an environment for stand-alone applications in which data sources, rendering and processing services are all tightly coupled and application specific. Therefore, they are not capable of allowing seamless interaction with the other data or processing/rendering services. These "deficiencies of centralized systems" and "improvements in the internet technologies" encouraged the academia, governments and businesses to start using distributed system approaches. Distributed systems are composed of autonomous hosts that are connected through a computer network. They aim sharing of data and computation resources, and collaboration on large scale applications.

The main challenge in sharing of data and computation resources and integrating GIS data is the heterogeneity of different sources. It has been studied from the perspective of how to resolve the semantic differences between heterogeneous data sources, mapping different schemas, and providing standard service and query interfaces. The adoption of GIS open standards for online service and data model solve the heterogeneity problems to some extent. OGC and ISOTC-211 are the well-known universally standards we use for this respect.

A large variety of different data sets are available in various specialized repositories, and users and geo-science applications would like to access these distributed heterogeneous data sources through uniform service interfaces enabling unified querying from a single access point. In the literature, these requirements are explained as “federation” which is initially used by database community [29, 30] to federate heterogeneous databases. Federation is basically established by working out the interoperability and inter-service communication issues among the distributed heterogeneous sources.

Distributed GIS systems typically handle large volume of datasets. Therefore the transmission, processing and visualization/rendering techniques used need to be responsive to provide quick, interactive feedback. There are some characteristics of GIS services and data that make it difficult to design distributed GIS with satisfactory performance. One of them is that services provided by a GIS typically require heavy CPU usage due to the complex computation involved in the underlying computational geometry. Another one is that GIS services often transmit large resulting datasets such as structured data, images, or large files in tabular-matrix formats.

In this thesis, we will first research architectural design requirements of federated GIS framework (in accordance with Service Oriented Architecture (SOA)) composed of Web Service components of commonly accepted GIS Open standards. Secondly, we propose a novel grid-enabled [10, 11] aggregator map services (federator) for optimizing the performance and responsiveness of the federated Service-Oriented GIS.

2. Motivation

GIS systems require experts from various areas such as data storage and maintenance, data rendering and displaying, and application specific computations. Since it is hard to gather data, hosts (such as high performance computers) and the necessary experts in a geographically same place, it is inevitable to develop interoperable distributed system architecture to enable coupling of these data and rendering/processing services remotely.

GIS is a system of computer software, hardware, and data used to manipulate, analyze, and graphically present a wide array of information associated with geographic locations. The ability to federate different kinds of sources results in more detailed and informatory information. This helps making better informed decisions as well as more effective and timely responses in emergency situations. However, long-standing challenges to data sharing and federation need to be addressed before the benefits of GIS can be fully realized.

GIS which is used in emergency early-warning systems like homeland security and natural disasters (earthquake, flood etc) requires quick responses. However, because of the characteristics of geo-data (large sized and un-evenly distributed such as population of human beings and earthquake seismic data), time-consuming rendering processes and limited network bandwidth, the responsiveness of the system is one of the most challenging issues of the distributed systems.

Here we give a motivating example for federation and performance issues in GIS. Let's consider an earthquake GIS application. In such an application, let's assume users need to display earthquake seismic data together with some other feature data-layers such as state and city boundaries over a satellite map from a single access point. The data sets might come from remotely distributed sources such as seismic sensors (or might be stored in databases), satellite map services and geographic feature services. Furthermore, for this specific example, users might need to ask for further information about the attributes of a specific feature (such as the magnitude of7 the specific seismic record) displayed on the map interactively. Moreover (from the performance point of view), users might need to see all the seismic records which have been recorded over the last 100 years. In such a case, the amount of data might reach to an amount in the order of gigabytes or even terabytes, and handling (transferring, federating and/or displaying) them becomes a challenge.

These requirements motivates us (1) to research federated GIS systems enabling seamless accessing and unified querying of distributed heterogeneous data and processing sources from a single access point, and (2) to develop interoperable and responsive GIS architecture handling large scale data for end users and geo-science applications.

3. Methodology

Our proposed GIS architecture is Web Service-based SOA implemented in JAVA. In order to develop and evaluate our architecture, we chose Apache Axis 1.x [15] version to deploy Web Services. We exploit Apache Tomcat [14] as a servlet container. Apache Tomcat is developed in an open and participatory environment. It implements Java Server Pages (JSP) and the servlet specifications from Sun Microsystems.

In order to test the proposed system for performance and responsiveness, we develop interactive decision making tools enabling interactive display and querying of the data. For the creation of interactive browser-based system, we use AJAX (Asynchronous JAVA +XML), Dynamic HTML, JavaScript and Web Service client tools.

In order to achieve interoperable GIS system we use universally accepted Open Geospatial Consortium (OGC) [25] and ISO-TC211 standards specifications for online services and data model. As online standard services we develop Web Map Services (WMS) [12] and Web Feature Services (WFS) [13] (implemented by Aydin G.), and convert them into Grid-enabled Web Services by extensions.

For the interoperability and federation issues, besides our WMS and WFS developed at CGL Lab, we will be integrating and using other third party OGC compatible WMS and WFS, and Google Map Services with our novel intermediary services.

4. Architecture: Federated Service-Oriented GIS

We first propose an interoperable Service-Oriented GIS architecture as a prototype (triangle in Figure 1), and then introduce innovative architectural framework for federating GIS Web Services. WMS Aggregator (federator) is the key service providing single access point to GIS Web Services and enables unified display and querying through the federation. Federation is based on common data model (GML) and capabilities files of the GIS Web Services. WMS Aggregator has also enhanced capabilities to optimize the performance and the responsiveness of the system (Figure 1).

Interoperability is obtained by using universally accepted OGC and ISOTC-211 standards. These entities define standards both for online services and data models, which have been widely adopted in the GIS community. Even the simplest GIS need at least three major types of online services. These are basically data providing, data processing/rendering and displaying services. According to OGC standard definitions we develop Web Feature Service (WFS) as data service, Web Map Service (WMS) as rendering service (like filters), and interactive user portal as displaying service (see Figure 1). All these services are extended with Web Service standards to make the whole GIS system Service-Oriented. Service-Oriented GIS enable each individual services to be published, located, invoked, and moreover, linked together to create more complex and explanatory information.

The proposed system also includes mediator services to enable inter-service communication between Service-Oriented and non-Sevice-Oriented GIS services such as NASA OnEarth WMS and Google Maps [31].

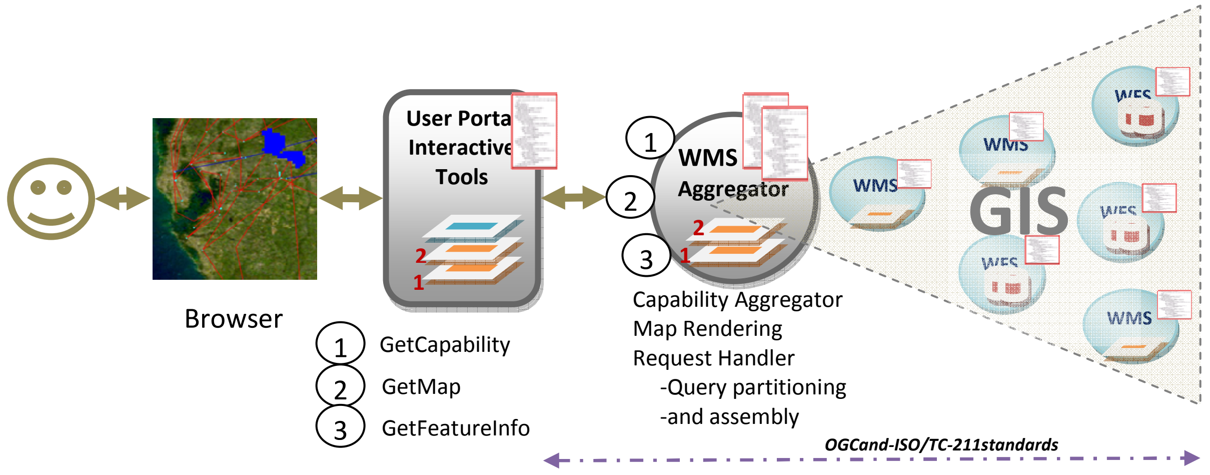

Figure 1: Proposed responsive and interoperable GIS architecture for end-users and Geo-science applications

As common data model we use semi-structured GML [4] data defined by widely accepted OGC and ISOTC-211 standards. GML is basically an XML encoding for the modeling, transport and storage of geographic information including the spatial and non-spatial properties of geographic features. Feature is an entity related to earth with a geographic location such as road, river, states etc. GML has separate schema standards for both content part and presentation part. That enables display and query of the data. Display is related to presentation part of the data schema and query is related to attributes of the data schema

There are two types of data flying around in the below GIS. One is for vector data encoded in GML common data model and provided to the system through WFS. Another is binary data encoded in map images. These are handled differently. GML encodes earth related features such as rivers, earthquake faults, seismic records, state boundaries etc in structured format.

Responsiveness and performance issues are mostly related to using structured common data model to solve the interoperability issues and limited bandwidth of the current internet. The key service proposed to handle these issues is WMS Aggregator (AWMS) (see Chapter 4.2) located at the top of GIS. It provides hierarchical data/information composition through its capability metadata. The general structure of the hierarchy (from top to bottom): project-> map -> layer -> {vector, raster and binary data}. It is actually a WMS with capability extensions such as: (1) map rendering in image layers, (2) information layering and layer-composition through capability federation (each service has capability in accordance with the schema defined by OGC), (3) request and response handlers covering query partitioning and assembling of the results returned to partitioned sub-queries. These are all handled according to the definitions and configurations in AWMS's capability metadata. User portal GUI is dynamically upgraded according to AWMS's capability metadata obtained through inter-service communication.

Streaming data transfer for GML data traveling from WFS to WMS is implemented by using publish-subscribe based messaging middleware called Naradabrokering [18]. The clients make the requests with standard SOAP messages but for retrieving the results, a NaradaBrokering subscriber class is used. Through first request to Web Service (called *getFeature)*, WMS gets the topic (publish-subscribe for a specific data), IP and port to which WFS streams requested data. Second request is done by *NaradaBrokering* Subscriber.

Parsing and rendering of the structured GML data we use Pull Parsing technique. Pull parser only parses what is asked for by the application rather than passing all events up to the client application as SAX parsing does. The pull approach of this parsing model results in a very small memory footprint (no document state maintenance required – compared to DOM), and very fast processing (fewer unnecessary event callbacks - compared to SAX). You see the article where pull parsing is compared with other leading Java based XML parsing implementations [26].

4.1. Components

The proposed GIS (triangle in Figure 1) is composed of WFS and WMS services. All these services are described by their capability metadata (RDF-WSDL like semi structured schema defined by OGC) enabling inter-service communications.

WFS (implemented by Aydin G.) provides interfaces for data access on geographic data kept in geospatial database to retrieve and present the data to the clients in a standard data model (GML feature collections). WFS provide three major interfaces. These are "GetCapabilities", "GetFeature" and "DescribeFeatureType". WMS interact with WFS by submitting standard structured queries in compliance with OGC's Filter Encoding and OGC Common Query Language.

WMS renders the raw binary or structured data captured from WFS and, creates comprehensible information in map images (MIME types JPEG etc). WMS defines a request/response protocol for web-based client/web-server interactions. It has three standard services, "GetCapabilities", "GetMap" and "GetFeatureInfo". Communications between WMS and WFS is achieved by exchanging the capability documents through their GetCapabilities service interface. Capability document include all the required information to make successive request to use other service interfaces such as GetFeature, GetMap and GetFeaturInfo.

GetCapabilities request allows the server to advertise its capabilities such as available layers, supported output projections, supported output formats and general service information. After getting this request, WMS returns an XML document with the metadata about the WMS Server. This capability file is kept in the local file system.

The getMap service interface allows the retrieval of the map. getMap request is consist of attribute value pairs. Attributes are predefined in a standard form but values are assigned in accordance with the capability document of the server invoked. The most of the WMS clients are browser based and GUI enabled. Values for the attributes are assigned with the help of user interface and interactive map tools such as zoom-in, zoom-out and panning.

GetFeatureInfo is for making unified querying over the feature data used for creating the map layers. It is used for querying the attributes (content) of the data. The common data model used to encode feature data in the system (GML) enables this. GML has two parts, one is content and the other is presentation. The presentation part is for rendering and displaying geometry elements such as points, line-strings, polygons etc. Content part is related to the attributes of the data such as the magnitude values of seismic records and/or length of a fault segment.

Interactive user portal: Interactive user portal provides an independent browser based GUI enabling interaction with GIS Web Services while hiding system complexity from the users. It has capability of inter-service communication with other WMS, WFS and more importantly AWMS. It gets the GIS Web Services' capabilities files and according to the capabilities of the communicated service, it dynamically upgrades its user interfaces. Initially communicated services such as AWMS address is defined in the user portal's properties file. The portal is also capable of being extended with some application specific functionalities such as coupling with geo-applications and creating map animations and movies.

4.2. WMS Aggregator (AWMS)

AWMS is actually a WMS with some extensions providing enhanced map rendering services by using innovative pre-fetching, parallel processing and caching techniques. AWMS aggregates, composes and orchestrates WMS and WFS services and, express the layer level compositions in its capabilities file by federating other services' capabilities metadata, and present its capability to the user portal (or client) with "GetCapabilities" Web Service interface.

In summary, pre-fetching is purely for overcoming the natural bandwidth problem, caching helps system prevent redoing the jobs of querying and rendering before, and parallel processing helps workload sharing and parallel job run. Depending on the data characteristics, AWMS uses only one or the combination of these techniques.

Pre-fetching is for rendering of the data - not changing often. Pre-fetching is getting the data before it is actually needed and keeping it in local server for the successive requests done for the same data. In our framework we use archived data and they don't change frequently. Therefore, it is not reasonable to transfer and render the same data repeatedly for every request with minor changes in their request criteria. We use pre-fetching as a solution to this type of problem. Pre-fetching will be done for GML data transfers between WMS and WFS based on the predefined periodicity. Periodicity is defined depending on the data's characteristics, ex. once a day.

Sometimes pre-fetching is not efficient because of the characteristics of the data such as changing frequently and limited storage capacity of the initiating server. In these cases, we use parallel processing approach applied together with caching.

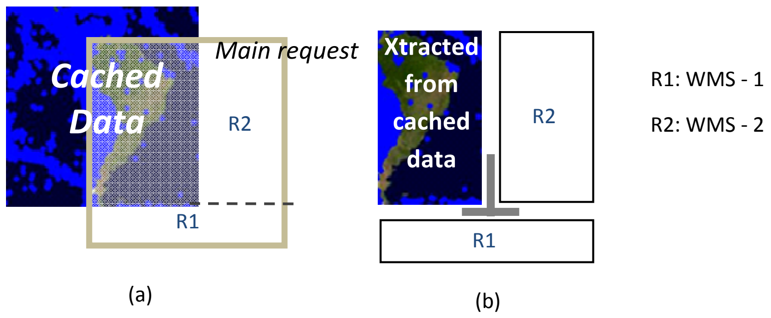

Figure 2: (a) cached data extraction (b) parallel processing through caching, query partitioning and rectangulation

AWMS apply the **parallel processing** after cached data extraction and rectangularion. Since all the data in the system is geo-referenced, and queried in ranges defined by bounding boxes (defining coordinates of rectangles in the form of (minx, miny, maxx, maxy)), we do range query partitioning to implement parallel processing. Partitioning the queries for parallel processing can be considered in two ways. One is layer-based (horizontal) another is partitioning the bbox criteria of the queries. These issues are taken cared of at the AWMS and layer compositions and partitioning are configured in its capability metadata.

As it is mentioned earlier, parallel processing is done for handling frequently changing data. We will research what techniques give better results for what type of data and in what conditions by doing benchmarking.

4.2. Work Plan

(1) Developing streaming and non-streaming versions of Web Map Services in accordance with universally accepted standards. (2) Extending WMS as Web Services –grid-enabled (3) Developing WMS Aggregator from basic WMS. (4) Federating GIS Web Services through

WMS Aggregator (5) Extending OGC standards to enable pre-fetching, parallel processing and caching. (6) Integrating the system with real geo-science applications such as PI and VC applications and other CGL lab projects such as WS-Context and HPSearch. It covers developing the interactive web-based map tools to interact with the coupling of real Geo-Science applications and map animation tools such as streaming map movies. (7) The performance and responsiveness tests and evaluation of the proposed system.

4.3. Benchmarking

After we are done with the architecture, we integrate the system with the real Geo-science applications such as Pattern Informatics (PI) [16] and Virtual California (VC) [17] through other CGL Lab projects such as WS-Context [27] and HPSearch [28].

In order to address research issues we will perform performance-responsiveness tests. We are planning to do below performance tests which are illustrated in Figure 3:

Each test will be done on a single machine, local area network and wide area network separately.

For different data size

- Timing for rendering and data display ***(T)***
    - Parsing and rendering a map from GML data locally in WMS
    - Caching and parallel processing (for the frequently changing data)
    - Pre-fetching (for the rarely changing data)
- Timing for data (GML) transfer issues (from WFS to WMS –see Figure 3) ***($t_{1.1}$-$t'_{1.3}$)***
    - Streaming data transfer (through pub/sub based messaging middleware [18])
    - Non-streaming data transfer
- Stress tests:
    - Streaming map movies requiring high performance data transfer and rendering.
    - Querying, rendering and displaying data in GBs (take hours – need in seconds)

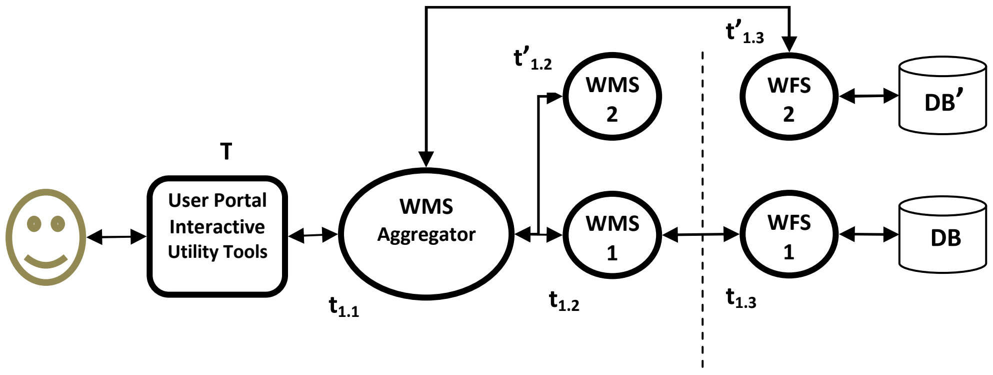

Figure 3: Performance test illustrations.

Measurement of success: We will compare the performances of the system with the other well-known deegree project [19] and possibly UMN MapServer [20] map-rendering tools. These are the two best well-known scientific map rendering services developed with the universally accepted standards.

5. Research Issues

In order to create federated Service-Oriented GIS and optimize its performance and responsiveness, we will investigate the following research issues.

The first one is creating interoperable Service-Oriented GIS. The interoperability challenges mostly result from data and service heterogeneity, and lack of wide-spread adopting of the universal standards. Service standardization includes first, defining the required standard online services for a complete system, and second, defining corresponding interfaces and message formats.

The second research issue is developing capability file-based federated GIS system. Federation is built on top of the proposed interoperable SOA-based GIS.

The third research issue is addressing the requirements for the performance and responsiveness of the proposed federated GIS involving large scale data transfer and processing. In most cases, the amount of collected data reaches to an amount in the order of gigabytes or even terabytes. Therefore, the GIS services must enable accessing and processing these large data sets in a reasonable time period. It even gets worse when the map animations and map movies (requiring many static map images to be created successively) need to be created.

Furthermore, due to the limited bandwidth and network speed, a GIS system faces the same performance problem as all the large scale distributed applications do. This challenge prevents making large scale geo-science applications feasible. Therefore, we will research the software level techniques such as parallel processing and caching on GIS domain.

In addition to aforementioned main research issues we will couple geo-applications with the proposed GIS system, explore high-performance streaming data services through pub/sub based messaging system, and develop architectural framework for map animations and streaming map movies based on time-series data over the proposed system.

We identify the following research questions:

- How to build Web Service-based Service-Oriented GIS.
- How to incorporate widely accepted geospatial standards with GIS Web Services.
- How to develop capability file-based federated GIS.
- How to optimize the performance and responsiveness of the proposed GIS.
- How to make responsive and unified querying of distributed data via their integrated views.
- How to apply parallel processing in order to enhance the GIS's responsiveness.
    - -How to make range query partitioning and assembling the hierarchical data.

6. Related Work

Linked Environments for Atmospheric Discovery (LEAD) [21] is a large scale project funded by NSF Large Information Technology (IT) Research grant for addressing fundamental IT and meteorology research challenges to create an integrated framework for analyzing and predicting the atmosphere [23]. On the other hand we aim same things for the earth-related Geo-science. LEAD and our architecture both use SOA for the utilization of distributed sources.

At the application level, LEAD supports for adaptive analysis and prediction of large scale meteorological events. They call it forecast mode using available observations or model generated data and manages necessary resources. They mostly focus on automated data management, scalable data archiving system and easy search and access interfaces via GUI and underlying ontology. LEAD has MyLEAD concept for enabling users to process their own data. Users can interactively explore the weather as it evolves, create custom scenarios or acquire and process their own data. Our approach is based on accessing querying and displaying the data not management.

GEON (Geo-science Network) [22] is large scale NSF-funded SOA based project adopted with a portal developed for end-users involving the development of a distributed, services-based system that enables geoscientists to publish, integrate, analyze, and visualize their data. Through its Service-Oriented architecture, GEON provides as single access point to geological data repositories and software tools. They deal with sharing, publishing and integrating the heterogeneous data sources, tools for the user portals and knowledge representations through their semantics.

Like us, GEON and LEAD both provide user portal and tools as the entry point for students, users or advanced researchers to the data, services, models, and workflow, analysis and visualization tools.

7. Expected contributions

From our proposed research work, we expect GIS to benefit from the application of techniques in distributed programming, and distributed programming to benefit from our analysis of capability file-based federation. We can also add a couple of minor contributions such as (1) introducing the innovative techniques for high performance and responsive map-data rendering, (2) interactive and seamless data access and querying architectures, and (3) developing interoperable Service-Oriented GIS with universally accepted standards (OGC and ISO-TC211)

References

[1] GIS Research at Community Grids Lab, Project Web Site: http://www.crisisgrid.org.

[2] Ahmet Sayar, Marlon Pierce, Geoffrey Fox OGC Compatible Geographical Information Services Technical Report (Mar 2005), Indiana Computer Science Report TR610

[3] Peng, Z.R. and M. Tsou, Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks. 2003: Wiley

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

[5] ESRI, ArcIMS, 9 Architecture and Functionality, J-8694. ESRI White Paper, http://downloads.esri.com/support/whitepapers/ims_/arcims9-architecture.pdf. 2004.

[6] Autodesk. MapGuide http://usa.autodesk.com. [cited.

[7] MapServer, W. http://www.wthengineering.com/GIS/web_gis.htm. [cited.

[8] de La Beaujardiere, J., Web Map Service, OGC project document reference number OGC 04-024. 2004.

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

[10] Fox, G. and M. Pierce. Web Service Grids for iSERVO. in International Workshop http://www.eps.s.u-tokyo.ac.jp/jp/COE21/events/20041014.pdf on Geodynamics: Observation, Modeling and Computer Simulation University of Tokyo Japan October 14 2004. 2004

[11] Fran Berman, Geoffrey C, Fox, Anthony J. G. Hey., Grid Computing: Making the Global Infrastructure a Reality. John Wiley, 2003.

[12] de La Beaujardiere, J., Web Map Service, OGC project document reference number OGC 04-024. 2004.

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

[14] Apache Tomcat, http://tomcat.apache.org/.

[15] Apache Axis, http://ws.apache.org/axis/.

[16] Tiampo, K. F., Rundle, J. B., McGinnis, S. A., & Klein, W. Pattern dynamics and forecast methods in seismically active regions. Pure Ap. Geophys. 159, 2429-2467 (2002).

[17] Rundle, P.B, J.B. Rundle, K.F. Tiampo, A. Donnellan and D.L. Turcotte, Virtual California: Fault Model, Frictional Parameters, Applications, PAGEOPH, submitted

[18] Pallickara, S. and G. Fox. NaradaBrokering: A Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. in Lecture Notes in Computer Science. 2003: Springer-Verlag.

[19] Deegree projects home page http://deegree.sourceforge.net/

[20] UMN MapServer project home page http://mapserver.gis.umn.edu/

[21] Beth Plale, Dennis Gannon, Dan Reed, Sara Graves, Kelvin Droegemeier, Bob Wilhelmson, Mohan Ramamurthy, "Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD", *To appear ICCS workshop on Dynamic Data Driven Applications*, Atlanta, Georgia, May 2005.

[22] GEON (Geosciences Network): A Research Project to Create Cyberinfrastructure for the Geosciences. http://www.geongrid.org

[23] Kelvin K. Droegemeier, et al. Linked environments for atmospheric discovery (LEAD): A cyberinfrastructure for mesoscale meteorology research and education. in 20th Conf. on Interactive Information Processing Systems for Meteorology, Oceanography, and Hydrology, . 2004. Seattle, WA.

[24] Sosnoski, D. "XML and Java Technologies", performance comparisons of the Java based XML parsers. Available at http://www-128.ibm.com/developerworks/xml/library/x-injava/index.html

[25] OGC (Open Geospatial Consortium) official web site http://www.opengeospatial.org/

[26] Sosnoski, D. "XML and Java Technologies", performance comparisons of the Java based XML parsers. Available at http://www-128.ibm.com/developerworks/xml/library/x-injava/index.html

[27] Bunting, B., Chapman, M., Hurlery, O., Little M., Mischinkinky, J., Newcomer, E., Webber J, and Swenson, K., Web Services Context (WS-Context) Specification, Version 1.0, July 2003.

[28] Harshawardhan Gadgil, Geoffrey Fox, Shrideep Pallickara, Marlon Pierce, Robert Granat, Proceedings of the IEEE/ACM Cluster Computing and Grid 2005 Conference, CCGrid 2005, Cardiff, UK

[29] McLeod and Heimbigner (1985). "A Federated architecture for information management", ACM Transactions on Information Systems Vol 3, Issue 3: 253-278.

[30] Sheth and Larson (1990). "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases". ACM Computing Surveys Vol 22, No.3: 183-236.

[31] Ahmet Sayar, Marlon Pierce, and Geoffrey C. Fox, Integrating AJAX Approach into GIS Visualization Web Services., IEEE International Conference on Internet and Web Applications and Services, ICIW'06, February 2006.