Streaming Data Services to Support Archival and Real-Time Geographical Information System Grids

Galip Aydin[1,2], Geoffrey C. Fox[1,2,3], Harshawardhan Gadgil[1,2] and Marlon E. Pierce[1]
[1]Community Grids Laboratory
[2]Department of Computer Science
[3]Department of Physics
Indiana University
{gaydin,, hgadgil,, mpierce}@cs.indiana.edu
{gcf}@grids.ucs.indiana.edu

Abstract— **In this paper we describe our efforts to build a Web Services based architecture to support both real-time and archived geographic data in Geographical Information System (GIS) Grids. We have built and tested Web Service version of OGC Web Feature Service (WFS) that can be used to provide archived geospatial data to various clients. Due to several performance issues described in this paper we also built a streaming version of Web Feature Service for large data sets where high performance is desired. We are also building a Sensor Filter Grid to process and serve real-time GPS messages over a publish/subscribe messaging middleware. We describe several filters built for this purpose and discuss initial performance results. As an example of how we can couple scientific simulation codes with our Grid architecture we describe coupling of a scientific data analysis application with GPS streams.**

Index Terms—**GIS, Grids, Web Services, Real-Time data analysis.**

I. Introduction

Geospatial data can be classified in two major categories in terms of their sources: a) archival data and b) real-time data collected from sensors. Traditional GIS applications such as map interfaces or geo-processing applications require access to spatial databases to extract both geographic data and associated metadata. On the other hand with the emergence of the sensors as an important new technology for real-time data acquisition, a new breed of applications developed to analyze the sensor observations in real-time or near-real time are appearing. These new type of applications require immediate access to data. Consequently a complete GIS Data Grid layer must provide support for data of both kinds.

The organization of the paper is as follows: The rest of the paper is divided into two sections; in Section 2 we review our efforts to build two different versions of Web Feature Service [1] for supporting archival geographic data in GIS Grids. We give summaries for both non-streaming and streaming versions and briefly compare their performances. In Section 3 we describe our initial implementation of a Sensor Filter Grid that supports real-time processing of GPS sensor measurements and coupling scientific applications with real-time data. We discuss the filters built for this purpose and give their initial performance tests results. We also review a sample application integration scenario with our architecture.

II. Supporting Archival Geographic Data in Geographical Information Service Grids

A. Open Standards for Geographical Information Systems

It is estimated that around 80% of all data available to humans contain some sort of a geospatial component [2]. This is the major reason for enormous interest from industry and academia to develop Geographic Information System applications which in turn caused use of a wide variety of software methodologies over the years. Several organizations produced geographic data adhering to proprietary representation schemes and employed different distributed computing practices to serve them. Furthermore because the data sources are owned and operated by individual groups or organizations, geospatial data is in vastly distributed repositories. As a result today the GIS community faces unique challenges to support interoperability between diverse set of data and service standards.

In recent years several organizations have developed data standards and implementation specifications for geospatial and location based services, in a bid to make geographic information and services neutral and available across any network, application, or platform. Two major players in this area are Open Geospatial Consortium (OGC) and the Technical Committee tasked by the International Standards Organization (ISO/TC211). The OGC is an international industry consortium of more than 270 companies, government agencies and universities participating in a consensus process to develop publicly available interface specifications. OGC Specifications support interoperable solutions that "geo-enable" the Web, wireless and location-based services, and mainstream IT. OGC has produced many specifications for web based GIS applications such as Web Feature Service [1] and the Web Map Service (WMS) [3]. Geography Markup Language (GML) [4] is widely accepted as the universal encoding for geo-referenced data. The OGC is also defining the SensorML [26] family of specifications for describing properties of sensors and sensor constellations. On the other hand ISO Standards proposes a standard framework for the description and management of geographic information and geographic information services. ISO/TC 211 did not specify the actual implementation specifications for different platforms and the private software vendors. Instead, ISO/TC 211 defines a high-level data model for the public sector, such as governments, federal agencies, and professional organizations [5].

B. *Web Service implementation of Web Feature Service*

The Open Geospatial Consortium Web Feature Service implementation specification [1] defines interfaces for data access and manipulation operations on geographic features using HTTP as the distributed computing platform. Via these interfaces, a web user or service can combine, use and manage geographic data from different sources by invoking several standard operations. Without having to consider the underlying data stores clients can access and manipulate geographic data via Web Feature Service. Also one Web Feature Service instance can be integrated with various types of data stores at the same time.

OGC Web Feature Service implementation specification defines HTTP as the only explicitly supported distributed computing platform which requires use of one of the two request methods: GET and POST. Although SOAP messages are also supported, they are also required to be transported using HTTP POST method. However HTTP significantly limits both service providers and consumers in terms of various capabilities Grid services can provide. These capabilities include providing standard WSDL interfaces to access various services, which can communicate with various databases or remote resources, ability to launch and manage applications remotely, or control collaborative sessions etc. Furthermore complex scientific applications require access to various data sources and run several services consecutively or at the same time. This is not in the scope of HTTP but can be supported using rapidly developing workflow technologies for Web and Grid Services. For these reasons we have based our Web Feature Service implementation on Web Services principals. Our goal is to make seamless coupling of GIS Data sources with other applications possible in a Grid environment.

We have initially implemented a Web Service version of basic Web Feature Service which supports three mandatory operations via a WSDL interface: GetCapabilities, DescribeFeatureType and GetFeature. Request and response of these operations are XML encoded GML documents. We chose to represent these XML documents in <xsd:string> type in our programming implementation. Although ideally we would define these in the <wsdl:types> section of our WSDL service definition, support for complicated, developer-defined types in Apache Axis [6] (our deployment framework) is limited and we wanted to keep our implementation as simple as possible for a wide variety of applications. However since the Web Service returns the XML document as an <xsd:string>, it first has to be constructed in memory and the maximum size of this document will depend on several parameters such as the hardware configuration of the system and memory allocated to the Java Virtual Machine etc. Consequently there will be a limit on the size of the returned XML documents.

Our Web Feature Service implementation has been used in several scenarios such as producing fault maps of Southern California, displaying seismic history of particular regions on the map and providing natural gas and electrical energy components for IEISS, a simulation application used by Los Alamos National Laboratory to analyze interdependency between energy nodes. Another interesting application domain was integrating our Web Feature Service with Pattern Informatics [7] code to forecast future seismic activities in a selected geographic region. This is described in more detail in [8]. Apart from these use cases we have done extensive performance tests to find the limits of our Web Feature Service implementation; these tests are explained in detail in [9]. From these performance tests we draw following conclusions. First, for small data payloads the response time is acceptable. However for larger data sets the performance decreases sharply and the response time is relatively long. Second, there exists a maximum threshold for the amount of data to be transported.

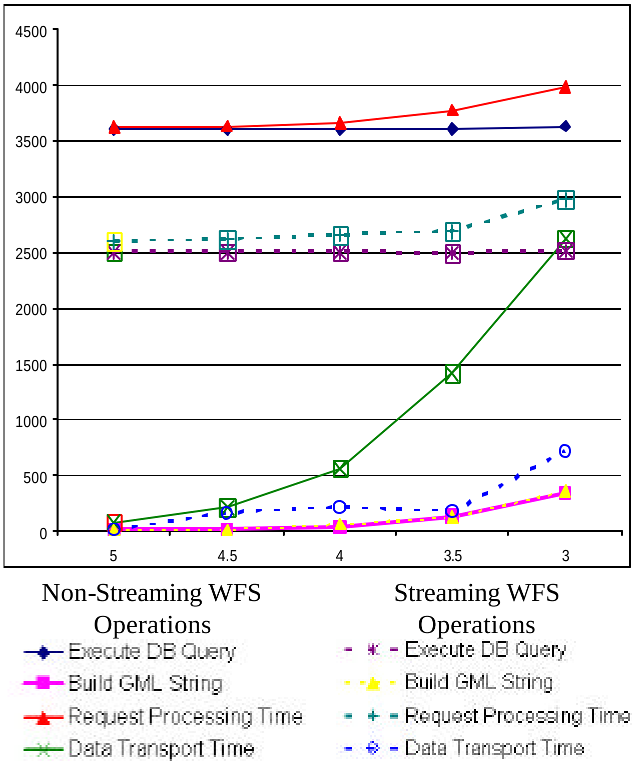

Fig. 1. Performance comparisons of the streaming and non-streaming WFS implementations showing the advantage of streaming the data over a messaging broker.

C. *Streaming Web Feature Service*

Our Web Service implementation of Web Feature Service proved to be useful for transporting relatively smaller data sets with transmission rates allowed by HTTP. However scientific applications such as RDAHMM (a time series data analysis program useful for mode change detection in a given series of data) [10], Pattern Informatics and IEISS may require larger amounts of data to be transferred between servers and clients with high transmission rates. Fast transfer of data payloads is also very important in crisis management and early warning systems where GIS applications are often used. For these reasons we have researched publish/subscribe based messaging systems as alternative to HTTP. Our research showed that NaradaBrokering [11] [12] can be used to transfer large amounts of data between publishers and subscribers without significant overhead.

NaradaBrokering is a distributed publish/subscribe messaging system which supports many-to-many messaging between entities such as clients, resources, services and proxies. It allows us to choose between different transport protocols such as UDP, SSL, HTTP, RTP, GridFTP etc. and between different qualities of service such as guaranteed delivery, once-only delivery, replayed delivery, secure delivery, and so forth. For more information about NaradaBrokering capabilities see [13]. Such capabilities already inherent in NaradaBrokering led us to develop a novel Web Feature Service that integrates OGC Web Feature Service specification with Web Service-SOAP calls and NaradaBrokering messaging system.

The difference between our streaming and non-streaming Web Feature Service implementations is that the streaming service employs NaradaBrokering topics to publish query results. The user queries are made with standard SOAP messages however instead of returning the results over HTTP, they are published to a pre-determined topic. In addition to the transport method the streaming-WFS has another important advantage: we utilize MySQL's ability to stream results from database row by row thus create and publish the GML feature members as they become available. This allows us to start publishing the results after a short query processing time without waiting for the whole result set to be returned from the database as in the conventional implementation.

Initial performance test results for out streaming-WFS implementation is discussed in brief in [14]. A more detailed discussion is given in [9]. Figure 1 shows the performance comparison of two Web Feature Service implementations. Note the 'Request Processing Time' plots the total amount of time it takes for the Web Feature Service to process the user requests. Overall the streaming-WFS outperforms non-streaming version by a significant margin for large data payloads and demonstrate an equal or better performance for smaller data sizes. Another important point is that there is no size limit for the data to be transported between Web Feature Service server and the client in streaming version which is a major advantage.

III. STREAMING SUPPORT FOR REAL-TIME SENSOR DATA

A. *Background*

Sensors are changing the way we acquire data about various entities. Recent advancements in sensor technologies such as micro-circuitry, nano-technology and low-power electronics allowed sensors to be deployed in a wide variety of environments [15] [16] [17] [18] [19]. The trend in this field shows us that in the near future we will see thousands of sensor nodes being deployed either individually or as part of sensor networks. For instance Southern California Integrated GPS Network (SCIGN) [20] has deployed 250 continuously-operating GPS stations in Southern California whereas several hundred more operating elsewhere in the United States.

Sensor networks constitute several sensors and advanced communication and computation infrastructures. They can be used to collect fine-grain information about the entity individual devices sense such as soil moisture or weather quality of a particular area. Several aspects of sensors and sensor networks such as power consumption, efficient data routing and security are the topics of much research. In this section we describe a service oriented approach to support coupling real-time sensor messages with scientific applications in a Grid environment. We discuss an interesting use case of scientific sensors, the GPS station networks and our efforts to make GPS messages available to clients such as RDAHMM and displacement filters.

GPS stations deployed alongside the active fault lines [20] are a good example of distributed scientific sensors. Instantaneous measurements from GPS stations are continuously collected and archived by several organizations throughout the world. These measurements are valuable for understanding the mechanics behind long term tectonic movements and seismic activities.

B. GPS Networks

The Global Positioning System stations deployed alongside the seismically active fault lines have been used to identify long-term tectonic deformation and static displacements. Continuous GPS has proven very effective for measurement of the interseismic, coseismic and postseismic deformation [21]. GPS Stations are effectively independent sensors that calculate and broadcast their instant geographic positions. They can run for long periods of times without need for frequent maintenance and can communicate with the data collection points using various connection types such as Wi-Fi, modems and phone lines or fiber-optic lines. Scripps Orbit and Permanent Array Center (SOPAC) [22] in University of California, San Diego is one of the several organizations that maintains several sub-networks of Southern California Integrated GPS Network (SCIGN) [20]. The raw measurements from the GPS sensors are continuously collected and locally stored by a Common Link Proxy (RTD) Server and later made available to public via FTP sites. The GPS networks provide real-time position data (less than 1 sec latency) and operate at high rate (1 – 2 Hz). The RTD server also broadcasts real-time positions in a proprietary binary format called RYO. Each RYO message contains the positions of the stations that reported for that epoch.

TABLE I
SAMPLE NARADABROKERING TOPICS FOR PARKFIELD GPSNETWORK.

Filter	Topic
ryo2ascii	/SOPAC/GPS/PARKFIEL/ASCII
ascii2pos	/SOPAC/GPS/PARKFIELD/POS
ascii2gml	/SOPAC/GPS/PARKFIELD/GML
Single Station	/SOPAC/GPS/PARKFIELD/HOGS

C. Filter Chains

To process GPS sensor streams in real-time we have developed several filters and Web Services to make real-time position messages available to scientific applications. In summary, the core of the system is to implement filter chains that convert or otherwise process the incoming data streams. These filters serve as both subscribers (data sinks) and publishers (data sources). NaradaBrokering topics are used to organize different data stream sources into hierarchies as shown in Table 1. Currently we are testing our system for 8 networks with 85 GPS Stations maintained by SOPAC. SOPAC RTD servers broadcast one message per network per second.

In our architecture filters are small applications designed to realize simple tasks such as transforming or aggregating messages. We have developed an abstract filter interface which can be extended to create new filters. A basic filter is consisted of three parts: a NaradaBrokering subscriber, a publisher and a data processing unit. The abstract filter interface provides subscriber and publisher capabilities. Typically a filter subscribes to a specified NaradaBrokering topic to receive streaming messages, process the received data and publishes the results to another topic. However outputs need not be always published, for instance a Database Filter may only receive the station positions to insert into the database. Furthermore filters can be connected in parallel or serial for realizing more complicated tasks.

The first filters we have developed are format converters that present original binary messages in different formats since GIS applications require different representations of geographic data. Since the original RYO messages contain multiple stations we also developed filters to extract and process individual station messages. Figure 2 shows our system architecture and current deployment of several filters. Here we give a brief overview of the filters we use in our architecture:

ryo2nb Filter: This is a simple message forwarding application that opens a TCP socket connection to the RTD server to receive the RYO messages and publishes to a NaradaBrokering topic (i.e. /RYO).

ryo2ascii filter: Subscribes to the RYO topic to receive binary messages, converts them to simple ASCII format and publishes to another topic (i.e. /ASCII).

ascii2gml filter: Geography Markup Language is perhaps today's most popular geographic data format produced by OGC. We have developed a GML Schema conformant with the latest Observations and Measurements [23] extension to describe GPS station messages. This filter converts the ASCII position messages into GML and publishes to a new topic (i.e /GML). We expect that in the near future most GIS applications will be developed to conform to OGC standards and presenting GPS messages in GML will help us easily integrate scientific applications.

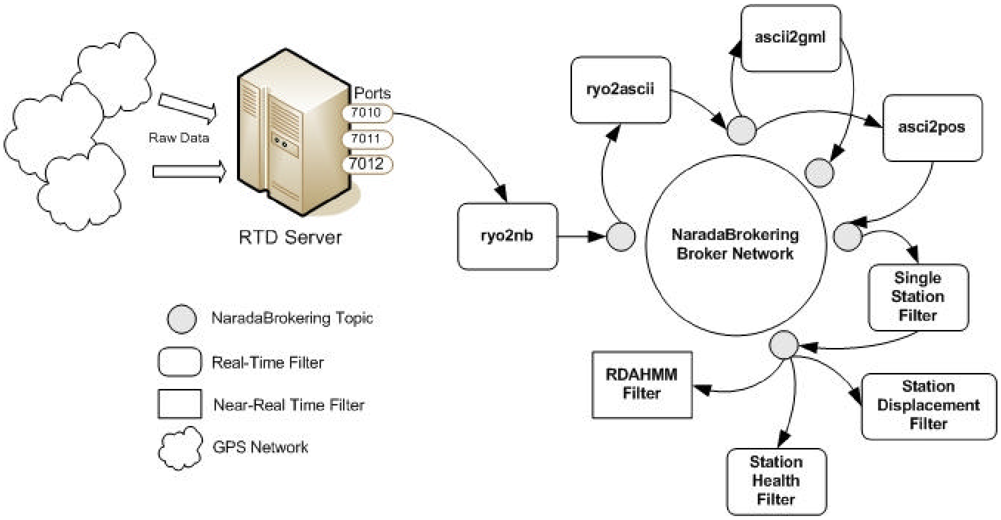

Fig. 2. Real-Time GPS messages are processed and served using various filters connected via NaradaBrokering messaging substrate.

ascii2pos filter: The RYO message type contains several sub parts other than physical position of the station such as position quality and several optional blocks. However most of this extra information is not required by the applications. This filter eliminates optional blocks and unnecessary information from the ASCII messages to create concise position messages which only include a time stamp, station id and position measurements.

Station Displacement Filter: One of the use cases of GPS stations is to detect seismic activities. We have developed a simple filter that analyzes position information of a GPS Station and outputs its real-time physical displacement. The filter allows displacements to be calculated based on different time intervals, i.e. actual displacement of the station in last hour or in last 24 hours.

Station Health Filter: One advantage of dealing with the real-time measurements is that we can instantly see if any of the sensors in a network is not publishing information. We have developed a filter that logs the down times of the stations and (potentially) alerts administrator if a threshold value is reached. For instance it can be tolerable for a GPS station to be down for a few minutes due to network problems etc. but if a station has not been publishing position information for over an hour a maintenance call may be required. Currently this filter is under development however we are planning to add alerting capabilities using e-mail etc.

Single Station Filter: As mentioned above the original messages imported from the RTD server contains position information for multiple stations. However most of our filters and applications analyze data for a particular station. For this reason we have developed this filter to strip the original message of unwanted parts and publish only the position information.

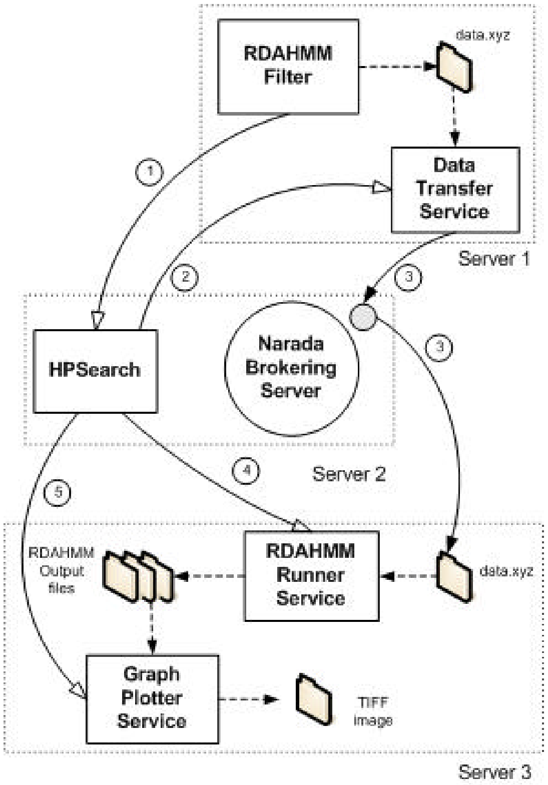

Fig. 3. System Architecture diagram for coupling RDAHMM with real-time GPS messages.

D. Application integration Use Case: Coupling RDAHMM with streaming data

RDAHMM is a data analysis program that employs Hidden Markov Models to identify different modes of the system and their probabilistic descriptions. An earlier, non-streaming version of this application was discussed in [24]. RDAHMM has been used to identify mode changes in GPS time series data. With the development of our real-time GPS data support architecture a new version of RDAHMM has been under development to analyze streaming data. Current version operates in two phases: Training and Evaluation. In our case first the application is trained on a set of data for a particular station. Then it can be run continuously on accumulated data once a pre-determined time window is reached. Although this version is not completely real-time we can run it near-real time by keeping the time window relatively small.

To integrate RDAHMM with our architecture we use HPSearch [25], a scripting based management interface to manage publish-subscribe systems. HPSearch also provides tools to wrap existing codes as Web Services and provides a scripting based workflow management interface [8] to connect different components of the workflow. HPSearch uses NaradaBrokering's publish/subscribe based messaging architecture to stream data between various services. Ref. [25] describes an initial version of RDAHMM using HPSearch. Figure 3 illustrates newer architecture for RDAHMM integration. As shown in the figure, the system consists of 3 Web Services, a NaradaBrokering server and an HPSearch node.

The Web Services in this architecture are as follows:

1- *DataTransfer Service*: This service transfers position messages accumulated by the RDAHMM Filter via NaradaBrokering to the server where RDAHMM actually runs.

2- *RDAHMMRunner Service*: Invokes RDAHMM to run on the transferred data set.

3- *GraphPlotter Service*: Runs Matlab to plot RDAHMM results as TIFF files and copies figures to a Web accessible folder.

Additionally HPSearch kernel also has a WSDL interface which is used by RDAHMM Filter to start the flow.

The system components are distributed over three servers. RDAHMM Filter and Data Transfer Service runs on Server-1. HPSearch kernel and NaradaBrokering server are installed on Server-2, whereas RDAHMM application, RDAHMM Runner Service and Graph Plotter Service run on Server-3. We also run an Apache Tomcat Web Server on Server-3 to present the generated TIFF images online.

The system uses following real-time filters described above: ryo2nb, ryo2ascii, ascii2pos and Single Station Filter. Additionally the RDAHMM Filter subscribes to a single station topic to save that station's position information.

The experimental system works as follows: The RDAHMM Filter is a part of the architecture discussed previously and shown in Figure 2. It accumulates the position messages of a particular station in a file (*data.xyz*) for a certain amount of time (i.e. 10 minutes for 600 lines, or 30 minutes for 3600 lines). Once the time threshold is reached it invokes HPSearch to start the process. HPSearch starts executing the script that defines the service locations and the order of the services to be executed. It first invokes the DataTransfer Service to start transferring the *data.xyz* file created by RDAHMM Filter to Server-3. Once this transfer is completed HPSearch engine invokes RDAHMMRunner Service and waits until it finishes the evaluation. Then it invokes GraphPlotter Service to read the RDAHMM outputs and plot the resulting graphic. This cycle is repeated every time the RDAHMM Filter reaches the time threshold.

For this system we have created a simple application that acts as the RTD server to publish RYO messages once per second. We used an RYO data set collected by 13 Parkfield GPS Network sensors for a 24-hour period between 09-27-2004, 06:59:47 PM and 09-28-2004, 06:59:46 PM. The latest major Parkfield earthquake occurred on 09-28-2004 at 10:15:24 AM.

The RDAHMM outputs tell us the number of different states detected in the input and information useful for plotting these states. Previous versions of RDAHMM were used to analyze archived GPS daily time-series and successfully detected state changes in the inputs which correspond to seismic events.

Our tests show that the real-time filters used in this architecture do not introduce any overhead. Since the GPS messages are received every second it is expected from the real-time filters to complete processing under one second not to skip the next message. According to our timing measurements all of the four real-time filters finish message processing under 100ms.

Currently we have successfully deployed and tested the system for one GPS station in the Parkfield test data. We tested RDAHMM using two different methods. First we used a sliding window method and run RDAHMM for every 1000, 3000, 5000 etc. lines of data. Next we applied an increasing window method by transferring every 1000 lines to RDAHMM server add appending this to previous data file. Thus RDAHMM was run on increasing data sizes.

IV. Conclusion and Future Work

In this paper we described several applications built to support two different layers of GIS Data Grids. Although most of today's GIS applications are based on consuming archival data, it is clear that sensors will be generating large amounts of data in the near future. For this reason real-time processing and evaluation of the sensor measurements is critical. We believe that our approach is fundamental and can be generalized to other problem domains besides GIS applications. The concept of distributed filters communicating via a messaging broker supports natural scaling.

We are currently working on to expand our system to test all of the stations included in the original RYO messages. Later we will plug-in to several real-time GPS networks.

We are also planning to use the sensor filter architecture to experiment whether we can use RDAHMM with real-time sensor data collected once per second and pick any patterns about the seismic activities.

ACKNOWLEDGMENT

This work is supported by the NASA Advanced Information Systems Technology program. We thank Dr. Yehuda Bock and Michael Scharber at SOPAC for help developing and deploying GPS sensor grid services.

REFERENCES

[1] Editor: Panagiotis A. Vretanos, Web Feature Service Implementation Specification, OGC document reference number: OGC 04-094, May 2005

[2] Liping Di, "Distributed Geospatial Information Services-Architectures, Standards, and Research Issues." ISPRS, Istanbul 2004

[3] Jeff De La Beaujardiere, OpenGIS Consortium Web Mapping Server Implementation Specification 1.3, OGC document reference number: 04-024, August 2002.

[4] Simon Cox , Paul Daisey, Ron Lake, Clemens Portele, Arliss Whiteside, Geography Language (GML) specification 3.0, OGC document reference number: 02-023r4., January 2003.

[5] Zhong-Ren Peng and Ming-Hsiang Tsou, Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks, available from http://map.sdsu.edu/gisbook

[6] Apache Axis project web site: http://ws.apache.org/axis/

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

[8] Galip Aydin, Mehmet S. Aktas, Geoffrey C. Fox, Harshawardhan Gadgil, Marlon Pierce, Ahmet Sayar SERVOGrid Complexity Computational Environments (CCE) Integrated Performance Analysis Proceedings of Grid Computing Conference, 2005. The 6th IEEE/ACM International Workshop 13-14 Nov. 2005. Page(s): 256 - 261 DOI

[9] Galip Aydin et al, "Building and Applying Geographical Information System Grids", Community Grids Lab Technical Report, submitted to special issue of Concurrency and Computation, Practice and Experience.

[10] Robert A. Granat: Regularized Deterministic Annealing EM for Hidden Markov Models, Doctoral Dissertation, University of California, Los Angeles, June, 2004

[11] Shrideep Pallickara, Geoffrey Fox: NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. Middleware 2003: 41-61.

[12] Geoffrey Fox, Shrideep Pallickara, Marlon Pierce, Harshawardhan Gadgil, Building Messaging Substrates for Web and Grid Applications to be published in special Issue on Scientific Applications of Grid Computing in Philosophical Transactions of the Royal Society of London 2005

[13] NaradaBrokering project web site: http://www.naradabrokering.org

[14] Geoffrey C. Fox, Mehmet S. Aktas, Galip Aydin, Hasan Bulut, Harshawardhan Gadgil, Sangyoon Oh, Shrideep Pallickara, Marlon E. Pierce, Ahmet Sayar, and Gang Zhai Grids for Real Time Data Applications Invited talk in proceedings of PPAM 2005 – 6th International Conference on Parallel Processing and Applied Mathematics Poznan Poland September 11-14 2005

[15] Kevin A. Delin and Shannon P. Jackson, "The Sensor Web: A New Instrument Concept", SPIE's Symposium on Integrated Optics, 20-26 January 2001, San Jose, CA

[16] Kevin A. Delin, "The Sensor Web: A Macro-Instrument for Coordinated Sensing.", Sensors, Vol. 2, (2002), pp. 270-285

[17] K. Martinez, J.K Hart and R. Ong, "Environmental Sensor Networks", Computer, Vol. 37, No. 8, pp 50-56, 2004.

[18] Chee-Yee Chong, Kumar, S.P, "Sensor networks: evolution, opportunities, and challenges", Proceedings of the IEEE, Volume 91, Issue 8, Aug. 2003 Page(s): 1247 – 1256

[19] Akyildiz, I.F., W. Su, Y. Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks", IEEE Communications Magazine, August, 102-114 (2002).

[20] Southern California Integrated GPS Network web site: http://www.scign.org/

[21] Bock, Y., Prawirodirdjo, L, Melbourne, T I. : "Detection of arbitrarily large dynamic ground motions with a dense high-rate GPS network" GEOPHYSICAL RESEARCH LETTERS, VOL. 31, 2004

[22] "SOPAC: Scripps Orbit and Permanent Array Center" web site: http://sopac.ucsd.edu/

[23] Open Geospatial Consortium Discussion Paper, Editor Simon Cox: "Observations and Measurements". OGC Document Number: OGC 03-022r3

[24] Harshawardhan Gadgil, Geoffrey Fox, Shrideep Pallickara, Marlon Pierce, Robert Granat, "A Scripting based Architecture for Management of Streams and Services in Real-time Grid Applications", Proceedings of the IEEE/ACM Cluster Computing and Grid 2005 Conference (CCGrid 2005). Cardiff, UK May 2005 Pages 710-717. No DOI

[25] Harshawardhan Gadgil, Jin-Yong Choi, Bernie Engel, Geoffrey Fox, Sunghoon Ko, Shrideep Pallickara, Marlon Pierce: "Management of Data Streams for a Real Time Flood Simulation", Community Grids Lab Technical Report, June 2004

[26] Botts, Mike, Sensor Model Language (SensorML) for In-situ and Remote Sensors, OGC document reference number 04-019r2