academicJournals

Vol. 8(46), pp. 2242-2256, 11 December, 2013
DOI: 10.5897/SRE2013. 5360
ISSN 1992-2248 © 2013 Academic Journals
http://www.academicjournals.org/SRE

Scientific Research and Essays

Full Length Research Paper

Fine-grained federation of geographic information services through metadata aggregation

Ahmet Sayar

Department of Computer Engineering, Kocaeli University,Kocaeli, Turkey.

Accepted 25 June, 2013

Large scale Geographic Information Systems (GIS) require experts from different areas, such as data maintenance and handling, data rendering and displaying, and data processing. However, it is challenging to host data, processing capabilities, and experts in the same geographical location. To enable remote coupling of these data and processing services, interoperable distributed system architectures have been developed. This paper presents a distributed Service Oriented Architecture (SOA) framework for understanding and managing the production of knowledge from distributed observations, simulations, and analyses through integrated data-views. Open GIS standards enable us to develop such a distributed framework by defining the standard common data models and corresponding web service components. The composability nature of these components has inspired us to develop a federated information system framework enabling both application-based hierarchical data definitions and performance enhancing designs.

Key words: Geographic information systems, federation, web services, data models, metadata.

INTRODUCTION

There are large amount of different datasets provided by various specialized repositories. Users and geo-science applications would like to access these distributed heterogeneous data sources from a single access point through uniform service interfaces enabling unified querying. In the literature, these requirements are explained as “federation”, which was initially used by the database community (Sheth and Larson, 1990). A federated database architecture is described in which a collection of independent database systems are united into a loosely coupled federation in order to share and exchange information. A federation consists of components and a single federal dictionary maintaining the topology of the federation. The federated architecture provides mechanisms for sharing data, sharing transactions and combining information from several autonomous components.

Geographic Information Systems (GIS) are systems for creating, storing, sharing, analyzing, manipulating, and displaying spatial data and associated attributes (Peng and Tsou, 2003). The purpose of GIS is for extracting information/knowledge from raw geo-data. The raw data is collected from sensors, satellites or other sources and stored in databases or file systems. The data goes through filtering and rendering services and presents to end-users in recognizable formats, such as images, graphs, charts, and so on. GIS is used in a wide variety of tasks, such as urban planning, resource management, and emergency response planning in case of disasters, crisis management, and rapid responses.

In this paper we propose a view-level federation of data/information provided by domain specific autonomous web services. Geographic Information Systems (GIS) is our selected domain (Peng and Tsou, 2003). Interoperability issues for the federation of services (and/or data) are resolved by adopting domain

E-mail: ahmet.sayar@kocaeli.edu.tr. Tel: +90 262 303-3583. Fax: +90 262 303 30 03.

specific standards (Open Geospatial Consortium (OGC, 1994) and ISO/TC211). These standards basically publish service definitions, data definitions, and metadata about data and services. There are three major data types, raster, vector, and coverage, provided by three major services, map services, feature services, and coverage services respectively. Map services are called Web Map Services (WMS) (Beaujardiere, 2004; Kolodziej, 2004), feature services are called Web Feature Services (WFS) (Vretanos, 2002), and coverage services are called Web Coverage Services (WCS) (Evans, 2003). Domain specific standards and related information are given in “Open Standards and Web Services in GIS” part of this work.

The proposed federator framework is an infrastructure for understanding and managing the production of knowledge from distributed observation, simulation, and analysis through integrated data-views in the form of multilayered map images. Infrastructure is based on a common data model, OGC compatible standard GIS web service components, and a federator. The federator provides one global view over several data sources processed as one source. The framework enables displaying geo-scientific application results on multi-layer map images and also enables scientific analysis and results to be understood and comprehend not only by the scientist but also the public and policymakers from different domains and education level. It gives a lot of advantages in interpreting and analyzing the geosciences application results. The integration of heterogeneous geospatial data offers possibilities to manually and automatically derive new information, which are not available when using only a single data source providing a single layer. For example, data from one information source (e.g. cadastre) can be used to enrich data from another one (e.g. topography). In this way, topographic road data can be enriched with address information, which is used as an indirect geo-reference in many other databases. Pattern Informatics (PI) (Tiampo et al., 2002) application can be given as another example. PI forecasts earthquake happenings and create a heat-map layer showing higher and lower possibilities of earthquake happenings with varying colors. Overlaying this result on a satellite map image, which is enriched with various vector data layers such as city-state boundaries and earthquake seismic data, will help application users better understand and conceive the results. Figure 3 shows an application scenario.

View-level federation can be better explained through an analogy. “Create View” SQL statement in databases creates a virtual table, called view. View is like a real table with its rows and columns but they are actually pointers to the rows and columns in other tables. Views are created by “create view” SQL statement and enable much more complicated queries to be created easily. A view can be used for simplification and customizing the perception each user has of the database. By following the same logic, we thought we can create a virtual map image whose layers come from remote servers. Such a map makes domain specific application developers’ work easier. Some of geosciences applications produce some outcomes which only make sense when they are overlaid on a map. In that way, application users obtain much more perceivable information for the application purposes. Application developers only concern with rendering their application specific data and overlaying it on the multilayer map image returned by the federator. From the application users’ point of view, multilayer map image is conceived as one-layer, but from the federator’s point of view it is multi-layer and each layer is provided by geographically distributed GIS web services. The layer composition is defined in federal schema in federator according to the application specific purposes. Federator handles query distribution and merging the results to create an abstract multilayer map image whose structure is defined in federal dictionary.

Federation work is based on creating a federal dictionary for combining services into a single, logically centralized entity, which we call it federator. Federal dictionary is created by harvesting standard GIS web services’ capabilities metadata. Each standard GIS services is defined with capability metadata. *Capability* is metadata about the data and services together. It includes information about the data and corresponding operations with the attribute-based constraints and acceptable request/response formats. Federal dictionary defines integrated data view in the form of multi-layer map images whose layers are created from spatially related vector, raster, and coverage geo-data sets provided by WFS, WMS and WCS respectively. Throughout the document, federal dictionary is called “capability” or “capability metadata”.

In addition to the usability advantages mentioned earlier, since the federator is a type of central approach built over the distributed autonomous data sources, the proposed architecture enables us to develop performance optimization techniques for distributed data access and query. GIS, 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 populations of human beings), time-consuming rendering processes, and limited network bandwidth, the responsiveness of the system is one of the most challenging issues of the distributed systems In this context, the federator’s aim is to turn open standards’ compliance requirements (such as using XML-encoded data models) into competitiveness and to provide performance enhanced responsive services that still meet the interoperability requirements. In this context, we have added topic-based publish-subscribe paradigm, which is mostly used in P2P systems, to the standard GIS web service communications. These are investigated in

”federator-oriented distributed query optimization” part of this work.

RELATED WORKS

Federation has been used for many purposes in various domains, and in many different contexts. It is initially used by database community (Sheth and Larson,1990; Vermeer and Apers, 1998) to extend an existing database with heterogeneous data that are separately owned. This usage saves maintenance or creation costs for the warehouse. Wide spread use of internet and developments in database technologies lead the federation approach to be used in various area such as digital library federations (Trnkoczy and Stankovski, 2008; Trnkoczy et al., 2006).

The concept of federation is not only used in data integration or database-like technologies, but also used in federation of computation-processing services. This kind of usage is encountered mostly in compositions of web services (Huang et al., 2011; Madsen, 2004; Pautasso, 2009) or composition of grid services (Vázquez et al., 2010; Leal et al., 2009) architectures. Moreover, federation approach is also utilized in various application domains such as social network federation (Chao et al., 2012), software federation (Anh et al., 2003) and wireless or sensor network federation (Al-Turjman et al., 2011).

It can be easily seen that federation approach might have varying application area, and developed for varying purposes. However, the challenge in federation, no matter what area it is used, arises from the heterogeneity of the autonomous components. A federated system can be composed of heterogeneous, that is, autonomous, components but they need to be interoperable. There are two levels of interoperability: syntactical interoperability and semantic interoperability. The former requires that there is a technical connection, that is, the data can be transferred between web services. The latter assures that the contents of data and services are correctly understood when data/services are connected.

The proposed distributed system framework is based on federation approach. Brokering (Tanenbaum, 2008; Erradi and Maheshwari, 2005) is an alternative approach for the similar purposes. In fact, a brokering solution does not impose any common/federal model but is able to implement different federal/common solutions and mediate between them. This can be thought as an advantage of brokering solution over federated approach. On the other hand, developing a federated approach is easier than developing its brokered counterpart. Since we develop a fine grained federation of GIS web services at the “view-level”, developing a federated approach would be more efficient for the purpose of the paper.

EuroGEOSS (Global Earth Observation System of Systems) project (EuroGEOSS, 2013) is an application of brokered approach, implementing multi-disciplinary interoperability and collaborating spatial data and services for both users and data providers. The brokering services in the architecture are grouped into three; discovery broker, access broker and semantic broker. In a brokering framework, application-level services might be possibly provided by the infrastructure to enrich the basic brokering functionalities. A Broker implements added-value functionalities related to its specific scope: Discovery, Access, Semantic expansions, etc.

Ontologies are important for the machines to understand the semantics of exchanged content (Gruber, 1993). W3C (W3C, 2008) recommends a standard called Web Ontology Language (OWL) to represent semantics based on a flexible graph model composed of Resource Description Framework (RDF) triples. It is initially introduced for defining resources on the web. Later, it has been used for some other related purposes in various domains. In GIS domain, Fonseca et al. (2002) analyses ontology based federation of services through Web Ontology Language (OWL). They propose ontologies for both object and field based modeling of geographic datasets, and analysis basics and boundaries of ontology-driven GIS. Morocho et al. (2003) proposes architectures for schema integration on federated spatial databases. They propose a federated schema in GIS framework by using OGC’s Geography Markup Language (GML) standard (Cox et al., 2003) and Spatial Data Transfer Standard (SDTS). They basically define ontology of GML data by means of SDTS. Compared to the work presented in this paper, they are not defining the overall federation system; instead they define only ontology-based semantic data integration. It is not clear how to access and query the data, which is integrated into the federated schema.

Another group of related works to solve semantic heterogeneity for the federation is based on developing application and/or domain specific schema definitions. In this case, since the schema definitions are not created in accordance with the commonly accepted and widely used standards, it is very hard for such frameworks to be adapted and extended by the third party applications. Batcheller (2008) presents a metadata generation approach for integrated data management. They implement an extension to Dublin Core geospatial profile of 23 elements. Dublin Core was originally capable of generating total for 20 basic metadata entries. Butenuth et al. (2007) propose a federated database framework for geospatial data integration. They define geospatial data semantically by using their own schema for predefined and classified object classes.

The proposed federation framework is based on metadata harvesting, similar to Open Archive Initiative’s Protocol for Metadata Harvesting (OAI-PMH) Lagoze and Sompel (2006) in digital library domain. Trnkoczy et al. (2006) work can be given as an example of this approach on a Grid-computing environment. In an application of OAI-PMH, digital libraries to be federated need to be defined earlier. Then, the system (or federator) harvests metadata from the selected digital libraries and creates and stores an index. Index represents the topology of contributing data sources. The focus here is defining federal dictionary in accordance with the domain specific application purposes.

In our work, priority is given on data representation. The syntactic and semantic heterogeneity issues are taken as granted by adopting domain specific open standards. If any two datasets are described with the same spatial reference system and formatted with the same projection system, then they are semantically compatible for the view-level integration. Using open standards enables the proposed federation framework to be possibly used by the third party systems and application developers. The proposed federation is based on defining and creating integrated data-view in the form of multi-layer map image. The federator provides one global view over several data sources processed as one source. Each layer is either rendered from GML data serviced by WFS or provided by WMS in a ready to use image format.

OPEN STANDARDS AND WEB SERVICES IN GIS

GIS are systems for creating, storing, sharing, analyzing, manipulating, and displaying spatial data and associated attributes. Spatial datasets have two kinds of attributes. One is spatial attributes and the other is non-spatial attributes. Spatial attributes carry location information. Non-spatial attributes are any other type of information about the data such as name and gender. Spatial data types are necessary to model geometry and to suitably represent geometric data in database systems. These data types are usually called spatial data types, such as point, line, and polygon. Spatial data types provide a fundamental abstraction for modeling the geometric structure of objects in space, their relationships, properties and operations. These properties of spatial data enable feature-based querying on the display of the real world objects.

The purpose of GIS is extracting information/knowledge from raw geo-data. The raw data is collected from sensors, satellites or other sources and stored in databases or file systems. The data goes through filtering and rendering services and presented to end-users in recognizable formats, such as images, graphs, charts, and so on. 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 responses (Peng and Tsou, 2003).

Over the past decade, GIS have evolved from the traditional centralized mainframe systems to desktop systems to modern collaborative distributed systems. Distributed systems are composed of geographically distributed and loosely coupled autonomous hosts connected through a computer network. They aim to share data and computation resources collaborating on large-scale applications. Modern collaborative GIS require data and computation resources from distributed virtual organizations to be composed based on application requirements, and accessed and queried from a single uniform access point over the refined data with interactive display tools. This requires seamless integration and interaction of data and computation resources. The resources span organizational disciplinary and technical boundaries and use different client-server models, data archiving systems, and heterogeneous message transfer protocols.

Interoperability and distributed services are clear trends that today’s GIS is taking. Standards for interoperability proposed by distributed frameworks such as the Open Geospatial Consortium (OGC) (OGC, 1994) offer advantages for data sharing, for combining software components and for overlaying graphical outputs from different sources. As a result, with a minimum need for adapting data products and software components to each other, standard distributed services offer the possibility to overlay image products coming from multiple data stores and processed by multiple map servers. The standardization efforts cause distributed services to be widely accepted and used in many areas such as governmental agencies and educational institutions. Two well-known and widely accepted standards bodies in the GIS domain are aimed at overcoming the interoperability issues (OGC, 1994). The aims of the standards bodies are to make the geographic information and services neutral and available across any network, application, or platform by defining common data models and online service descriptions. The standards bodies specify methods, tools, and services for data management, accessing, processing, analyzing, presenting, and transferring such data in digital form between different users and systems. ISO/TC211 defines a high-level data model for public sectors, such as governments, federal agencies, and professional organizations (Peng and Tsou, 2003). On the other hand, the OGC is interested in developing both abstract definitions of Open GIS frameworks and technical implementation details of data models, and to a lesser extent, services. Web Map Service (WMS) and Web Feature Service (WFS) are two major services defined by the OGC for creating a basic GIS framework enabling information rendering of heterogeneous data sources as map images. Web Coverage Service (WCS) Evans (2003) is another OGC defined data service. WCS provide coverages representing space/time-varying phenomena that relate a spatio-temporal domain to a (possibly multidimensional) range of properties. WCS provides available data together with their detailed descriptions; defines a rich syntax for requests against these data; and returns data with its original semantics. WMS are the key services to the information rendering and visualization. WMS produces maps from the standard geographic data encoded in the Geography Markup Language (GML) Cox et al. (2003) obtained from various WFS instances, and from coverages obtained from WCS instances. It also enables attribute and feature-based data querying (through WFS) over the data display from its standard service interfaces. This general approach is similar to the SkyServers Gray et al. (2002) defined by the National Virtual Observatory (NVO) community. The OGC's WFS implementation specification defines interfaces for data access and manipulation operations on *geographic features*. Geographic features are basically earth-related data definitions, such as rivers, lakes, earthquake seismic records, and so on.

In addition to the domain-level interoperability and extensibility mentioned above, information systems need cross-language, operating system, and platform interoperability to enable data sharing/federating and analysis over autonomous heterogeneous resources provided by various organizations. Web service standards Booth et al. (2004) are a common implementation of Service-Oriented Architectures (SOA) ideas, giving us a means of interoperability between different software applications running on a variety of platforms. A web service is an interface that describes a collection of operations that are network accessible through standardized XML messaging Kreger (2001). Collectively, web services are a software framework designed to support interoperable machine-to-machine interactions over a network. Other systems interact with the web services in a manner prescribed by its description using SOAP-messages (Simple Object Access Protocol), typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Adopting GIS and web service standards and implementing web service versions of standard GIS services permits applications to span programming languages, platforms, and operating systems Pierce et al. (2008). It also enables application developers to integrate the third party geo-spatial functionality and data into their custom applications easily.

FINE-GRAINED FEDERATION ARCHITECTURE

View-based data federation is a framework that solves the data integration problem for structured data by integrating sources into a single unified view. This integration is facilitated by a declarative mapping language that allows the specification of how each source relates to the unified view. In GIS domain, such unified views are used in many geo-science and geo-physics applications by using ad-hoc solutions. Figure 3 can be given as a motivating scenario in which earthquake seismic data records and LandSat satellite map image are combined at view-level, and as a result, new information is obtained. Users from different expert levels can easily see that south west region of Turkey has higher earthquake records than the other regions.

The proposed federator framework is an infrastructure for understanding and managing the production of knowledge from distributed observation, simulation, and analysis through integrated data-views in the form of multilayered map images. Infrastructure is based on a common data model, OGC compatible standard GIS web service components, and a federator. The precondition to be able to define a unified view (view-level data integration) is that all heterogeneous datasets representing a layer in the view needs to be created and stored by using the same spatial reference system and as well as the same projection system. If the data sets satisfy this condition they are said to be semantically compatible, that is, they can be overlaid. If a data provider is not OGC compatible, their data can be possibly integrated to the system through mediator services (Wiederhold, 1995). Mediators enable interoperability between heterogeneous data sources and the proposed system by performing resource specific conversions. The federator provides one global view over several data sources processed as one source (). There are three general issues here. The first is the data modeling (how to integrate different source schemas); the second is their querying (how to answer the queries posed on the global schema); and the third is the common presentation model of data sources, that is, the mapping of a common data model to a display model, enabling integration/overlaying with other data sets (integrated data-view). The first two groups of research issues are related to lower level (database and files) data format/query/access heterogeneities, summarized as semantic heterogeneity. In the proposed framework, OGC specifications for data models (GML) and online services (WMS and WFS) define these. The following parts of the work "federation framework" and "federation through capability aggregation" present the proposed solution approach to the third issue.

Federation framework

Figure 1 shows a three-level hierarchy data system, which is the essential framework of the proposed system. Heterogeneous data sources, which form the bottom layer of the hierarchy, are integrated into the system through mediators (WMS and WFS). Mediators provide an interface for the local data sources and play the roles of connectors between the local and the global sources Wiederhold (1995). The mediators not only enable data sources integrated into the system to conform to the global data model, but also enable the data sources to maintain their internal structure. In the end, the whole mediator system provides a large degree of autonomy (Figure 1).

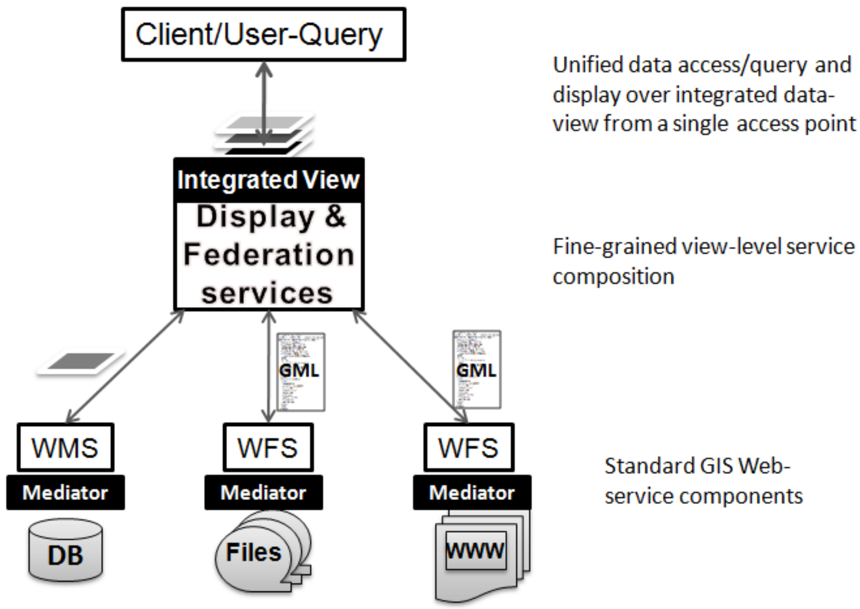

Figure 1. Data life-cycle and integrated data-view creation.

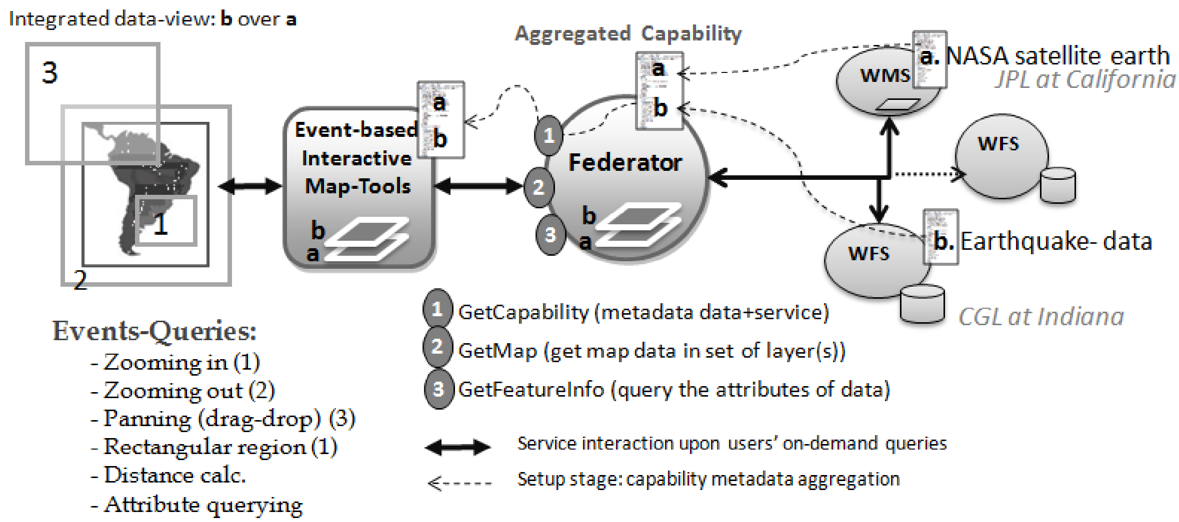

Figure 2. Federated GIS framework.

The proposed federation framework is a 2-stage process. The stages are compile-time (or setup time) and run-time. The compile-time stage defines integrated data-view and its components in terms of layers and corresponding web services. In the first stage, the federator searches for the standard GIS web service components (WMS or WFS) providing required data layers and organizes them into one aggregated capability file as indicated in the dotted lines in Figure 2. There is no client/user interaction with the system in this first stage. This stage is the core of the proposed framework and explained in "federation through capability aggregation" part of the work in detail. In the run-time stage, there is client/user interaction with the system through a browser that provides an event-based interactive display and query tools, such as map displaying tools as indicated in the solid arrows in Figure 2. Zooming in/out, distance calculation, dragging and dropping are given as examples of event-based queries in Figure 2.

The proposed federation framework does not support automated service registry or sign off. It does not have automatic service discovery capability. The topology of contributing services (aggregated capability metadata) is

Figure 3. Integrated data-view. Layers are Nasa-Satellite (raster) and Earthquake-seismic (vector).

defined at the beginning at compile/setup time manually by the developer. Contributing components are supposed to define their standard services and data in accordance with the publicly available open standards (OGC). These definitions are done through capability metadata. Each type of component (WMS and WFS) has its own type of schema to define its capability metadata in which available information/data and related operations are defined and binding information (information about how to access those data) is provided. In GIS, each component has its own global schema. Underlying data accesses are manipulated and mediated by their global schema. All these services and their data are publicly available and autonomous. We have nothing to do with those services except using their data through their standard service interfaces.

Figure 2 represents the proposed federation framework on a sample scenario. A WMS provides NASA satellite map images in raster formats with corresponding capabilities metadata. Its metadata is called “a”. A WFS provides earthquake seismic data records in GML format with its corresponding capabilities metadata are called “b”. At compile time federator collects their metadata and creates a aggregated metadata carrying information about satellite map images and earthquake seismic data records (a and b). The federator serves these datasets as if they are its own. Figure 2 also shows an event-based interactive map tool displayed on browsers and enable user interaction with the system. The client tools convert users’ event-based actions into standard web service queries sent to the federator.

The focus of this paper is on combining information from several components to form a view as a map. Federation is based on a hierarchical data definition as multilayer maps. This definition is done in the federator. Hierarchical data is described as below. The more detailed illustration of this is given in “federation through capability aggregation” part of this work as a capability metadata.

Map -> Layer -> Data [vector (GML) / raster (binary images)] --->Raw data

A sample scenario illustrated in Figure 2:
[Map]
- *[Layer]*
  - *[Data] - raster*
    - *Nasa Satellite Earth Images (binding information for WMS)*
- *[Layer]*
  - *[Data] - vector*
    - *Earthquake-Seismic-Data (binding information for WFS)*

A map is an application-based, human-recognizable, integrated data display and is composed of layers. A layer is data rendering of a single homogeneous data source. Layers are created from the structured XML-encoded common data model (GML) or binary map images (raster data). Heterogeneous data sources (*raw data*) are integrated into the system as GML or binary map images through the resource specific mediators. The mediators have resource specific adaptors for request and response conversions and appropriate capability metadata describing the data and resources.

To illustrate service federation, we give a real geo-science application as an example. In the Pattern Informatics (PI) application (Tiampo et al., 2002), decision makers need to see earthquake forecast values and seismic data records plotted on satellite map images. Satellite map images are provided by the NASA OnEarth server (a WMS) located at the NASA Jet Propulsion Laboratory (JPL) (OnEarth, 2007), and the WFS at the Visualization Laboratory at Kocaeli University provides the earthquake seismic data records. The federator aggregates these services’ standard capability metadata and creates an aggregated one as if those data sets were its own. The output of this federation is an integrated data view given in Figure 3. The users access the system as though all the data and functions come from the federator. The data distribution and connection paths stay hidden and formulated in the federator’s aggregated capability metadata.

Federation through capability aggregation

Capabilities are metadata about the data and services and have an XML schema defined by the OGC. Capability descriptions include information about data and its corresponding operations with the attribute-based constraints and acceptable request/response formats. It supplements the Web Service Description Language (WSDL) (Christensen et al. 2001), which specifies key low-level message formats but does not define information or data architecture. These are left to domain specific capabilities metadata and data description languages (such as GML). Capabilities also provide machine and human readable information that enables integration and federation of data/information. It also aids the development of interactive, reusable client tools for data access/query and display. We use the open standard specifications' definitions and present the required extensions for the federation through hierarchical data creation.

Let's assume we are federating two datasets as shown in Figure 2. These datasets are "Nasa Satellite" and "Earthquake-Seismic-Data". "Earthquake-Seismic-Data" is also called earthquake data. As mentioned earlier in "federation framework" part of this work, these datasets are actually defined as layers and federation outcome is defined as a map. The remaining of this part of this work describes our approaches to view-level federation of those datasets and creating a data representation model through a capability metadata (that is, federal dictionary).

There are two possible ways to bind the services to the federator to be able to create an application specific hierarchical data in an integrated data-view. One is borrowing some definitions of Web Map Context's (WMC) standards (Sonnet, 2005) and extending them for the federation purposes. WMC do not include WFS binding information in its service composition definition which is called context-document. It needs to be extended in accordance with federator's aims. Another alternative is extending the WMS' standard capability schema definitions by giving the reference to the service access points providing the required layer (WMS) and/or feature data (WFS). These approaches are explained below.

In the first approach, this paper utilizes the context document specifications of the OGC. The OGC's WMS and WFS services are inherently capable of being cascaded and chained in order to create more complex data/information. To standardize these issues, the OGC introduced the Web Map Context (WMC) standard specifications. A WMC is actually a companion specification to WMS. It is one of OGC specifications for describing how a specific grouping of maps from distributed Web Map Servers can be described in a portable, platform-independent format to store or transmit between clients. WMC define project-based contents of layers possibly be used as a map in a GIS application. WMC do not define overlay layers, which are rendered from GML provided by WFS. It only describes layers from WMS. In this paper, we extend context-document schema definition for the federation purposes, and enable integration of layers both from WMS and WFS.

This description of federal schema is done in a portable and platform-independent format (XML) and called "context document" or "context". A *context* document contains information about the composed layers to create an overall map. Each layer is described with domain specific attributes such as map projection system and bounding boxes in which layers are available. The *context* document can provide default startup views for particular classes of users. An example of a context document is given below. It describes the integrated data view displayed in Figure 3. The unnecessary details at the above context file are truncated. We just use related elements and tags for the data cascading and service binding. The standard schema for a context document is given in Sonnet (2005).

```
<ViewContext version="1.0.0" id="OGCContext"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <General>
  <Window width="500" height="400" />
  <BoundingBox srs="EPSG:4326" minx="-180.00" miny="-90.00" maxx="180.00"
maxy="83.62" />
  <Title>Maps for Pattern Informatics Application</Title>
  <Abstract />
 </General>
  ….
 <LayerList>
  <Layer queryable="1" hidden="0">——————→ WFS Binding
   <Extension infoFormat="text/xml" ID="4e4b-83e" editable="0" local="1" />
   <Server service="WFS" version="1.1.0" title="CGL_WFS">
    <OnlineResource xlink:href=" http://toro.ucs.indiana.edu/cgi-bin/wms0.cgi?" />
   </Server>
   <Name> Earthquake-Seismic-Data </Name>
   <Title>Earthquake Seismic Data</Title>
   <Abstract>Sample WMS to WFS layer cascading</Abstract>
   <DataURL format="text/xml">
    <OnlineResource xlink:href=" http://toro.ucs.indiana.edu/cgi-bin/wms0.cgi?" />
   </DataURL>
   <SRS>EPSG:4326</SRS>
   <FormatList>
    <Format current="1">image/png</Format>
   </FormatList>
```

```
   …..
  </Layer>
  <Layer hidden="0">──────────► WMS Binding
    <Extension infoFormat="text/html" ID="1fc-4e4b-83e" editable="0" local="1" />
    <Server service="WMS" version="1.1.1" title="CGL_WMS">
      <OnlineResource xlink:href=" http://wms.jpl.nasa.gov/wms.cgi" />
    </Server>
    <Name> Nasa Satellite Earth Images </Name>
    <Title>Nasa Satellite Data</Title>
    <Abstract>Sample WMS to WMS layer cascading</Abstract>
    <DataURL format="text/xml">
      <OnlineResource xlink:href=" http://wms.jpl.nasa.gov/wms.cgi" />
    </DataURL>
    <SRS>EPSG:4326</SRS>
  </Layer>
  …..
 </LayerList>
 …
</ViewContext>
```

In the second approach, WMS are extended with the federator capabilities. Data providing in the WMS are called “layers” and defined in layer tags in the capability metadata with attributes and features according to the standard WMS capability schema (Beaujardiere, 2004). Service binding is accomplished through the cascaded layer definition. A layer is regarded to have been “cascaded” if it was obtained from an originating server and then included in the capabilities XML of a different server. The second server may simply offer an additional access point for the layer, or may add value by offering additional output formats or spatial reference systems. If a WMS cascades the content of another WMS, then it must increment the value of the cascaded attribute for the affected layers by 1 (see the example below). If that attribute is missing from the originating WMS’s capabilities XML (that is, the layer has not been cascaded before), then the cascading WMS inserts the “cascade” attribute to the layer tag and sets it to 1. The default value of cascading is 0 (Kolodziej, 2004). Federator is not supposed to provide any layer by itself, it is a uniform access point for the registered data services. Therefore each layer in its capability metadata needs to be defined with the attribute cascaded and set to 1. A small part of the federator’s capability metadata is displayed below, unnecessary details are truncated. It is developed for the sample case illustrated in Figures 2 and 3. WMS’s standard capability metadata schema is available at OGC (2012).

```
<?xml version="1.0" encoding="UTF-8"?>
<WMS_Capabilities xmlns="http://www.opengis.net/wms"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wms>
……
….
<Layer cascaded="1">
        <Name> Earthquake-Seismic-Data </Name>
        <Title> Earthquake-Seismic-Data layer</Title>
         <LatLonBoundingBox minx="-180" miny="-90" maxx="180" maxy="90" />
        …
        <DataURL>
                <Format>image/gif</Format>
                <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
                  xlink:type="simple" xlink:href=" http://toro.ucs.indiana.edu/cgi-
bin/wms0.cgi?" />
        </DataURL>
</Layer>
…..
```

```xml
<Layer cascaded="1">
        <Name>NASA Satellite</Name>
        <Title> NASA Satellite </Title>
          <LatLonBoundingBox minx="-180" miny="-90" maxx="180" maxy="90" />
        <SRS>EPSG:4326</SRS>
        …
        <DataURL>
                <Format>image/gif</Format>
                <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
                  xlink:type="simple" xlink:href=" http://wms.jpl.nasa.gov/wms.cgi?" />
        </DataURL>
</Layer>
…..
</Capability>
</WMS_Capabilities>
```

FEDERATOR-ORIENTED DISTRIBUTED QUERY OPTIMIZATION

Distributed GIS systems typically handle large volume of datasets. Therefore, transmission, processing, and visualization/rendering techniques need to be responsive to provide quick, interactive feedback. Some characteristics of GIS services and data make it difficult to design the distributed GIS with a satisfactory performance. One is that processes and analysis in GIS necessitate heavy CPU usage. This mostly stem from the complexities in underlying computational geometry. Another is that processes and analysis in GIS often require transmitting large datasets, such as annotated-structured data (due to the interoperability requirements), images, or large files in tabular-matrix formats. In most cases, the amount of collected data reaches an amount in the order of gigabytes or even terabytes. Therefore, the GIS services must enable accessing and processing of these large data sets in a reasonable time period. This scenario is even worse when map animations and map movies are created (requiring many static map images to be created successively). 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.

In such an application framework (fine-grained federation architecture and Figure 2), in which you know what data sets need to be used and from where, you can apply some well-known performance improving techniques. Federator is a kind of central approach built over distributed autonomous data sources. The central approaches give the better performance but in most cases, due to the application requirements, it is inevitable that distributed approaches will be used. In the proposed framework, we take advantage of both centralized and distributed approaches. The following part of this work "Pre-fetching" presents the architecture details.

Pre-fetching

Pre-fetching is briefly defined as retrieving the data before it is needed and used to overcome the performance bottleneck of the distributed systems. The bottleneck is transferring large-sized data from source (database) to destination in the common data model, GML. On-demand accesses to the originating databases through WFS are very costly, not only because of the requirement of moving the large data, but also due to query and response conversions. To solve that bottleneck problem, the federator periodically fetches and updates whole data in databases into GML sets and stores it locally. The successive queries are served from pre-fetched data in a federator's local disk. To reduce the inconsistency problem, the fetching module fetches and updates the data periodically. It indirectly enables getting over the query/response transformation overhead at WFS.

Datasets to be pre-fetched are predefined by application developer. Developer needs to be experienced about the contributing servers and their performance. If the contributing servers' performances are good enough, then their data do not need to be pre-fetched.

The proposed architecture has two processes. The first process is pre-fetching, which is independent of application run time, and it does not affect on-demand data accesses from the clients. The second process is users' on-demand access, which is served from the pre-fetched data. The federator behaves as proxy for other distributed data sources by collecting their data in a ready to use common data model (GML) at periodic intervals, and successive queries are responded centrally from a federator's disk space in which the pre-fetched GML data

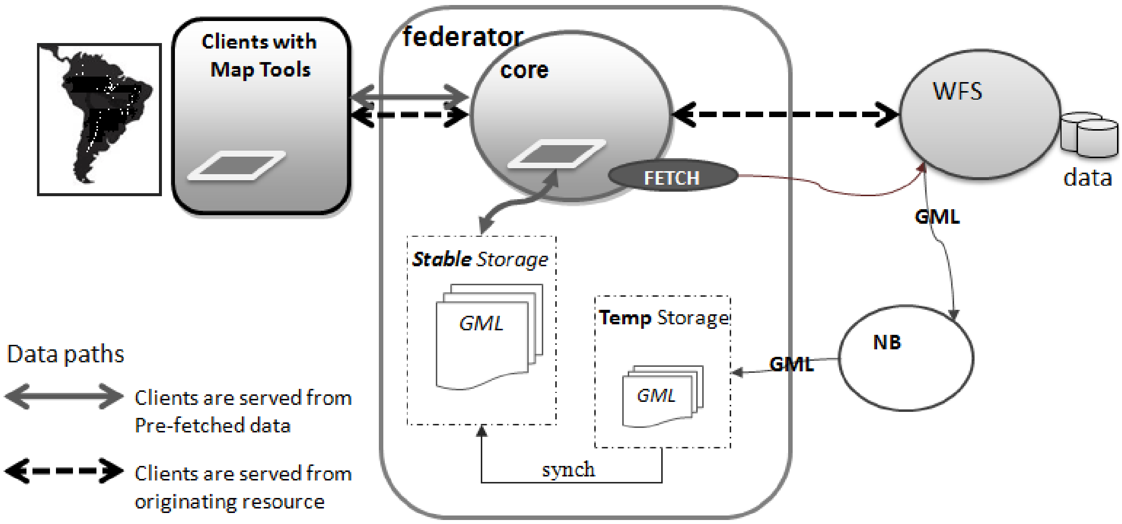

Figure 4. Pre-fetching architecture.

sets are stored. Figure 4 presents the pre-fetching approach.

Data transfers between standard GIS web services are done over the standard HTTP protocol. In our earlier works, we have extended OGC's transfer protocol in accordance with web service standards. Here, we go one step further and integrate topic based publish-subscribe paradigms (which is mostly used in P2P systems) into the communication between WFS and federator to transfer feature collections. NaradaBrokering Pallickara and Fox (2003) is one of the well-known applications of that approach enabling streaming data transport, reliable delivery, and recovery from network failures at the application level. In this approach, standard web service interfaces are used as a handshake protocol and the actual data is transferred over publish-subscribe based messaging system (NaradaBrokering, in Figure 4). This approach has some advantages over using pure web services. The system gets rid of the SOAP message creation overheads, and enables creation of map layers with partially returned data.

As shown in Figure 4, in the pre-fetching, requests requests for data are done through standard SOAP messages. On the other hand, to retrieving the results, NaradaBrokering system is used in which WFS (server) become publishers and the federator (client) becomes subscriber. Through the "*getFeature"* interface of WFS web services, the pre-fetching module gets the topic name (publish-subscribe for a specific data), IP, and port on which WFS stream the requested data. The

NaradaBrokering subscriber does the second request using the returned parameters. GML data is provided by streaming WFS (Vretanos, 2002). It uses standard SOAP messages for receiving queries from the clients; however, the query results are published (streamed) to a NaradaBrokering topic as they become available. To do that, we define the "task" and "timer". Task defines the pre-fetching job, and timer defines the running periodicity of the task. Aydin et al. (2008) for more details about the streaming data transfer between GIS web services by using NaradaBrokering.

As Figure 4 illustrates, there are two separate storage locations for the data: temporary storage and stable storage. The former is for pre-fetching, and the latter is for serving the clients' on-demand queries. Even if the system is busy with pre-fetching, it keeps itself up and running for the clients by using the stable storage. When the data transfer to the temporary location is complete, all data at that location will be moved to stable location. Reading and writing the data files at the stable locations are synchronized to keep the data consistent. This cycle is repeated at some time intervals predefined by the periodicity parameter of the pre-fetching module. Since the data fetching is done independent of real application time, it does not affect the application performance. In order for the pre-fetching algorithm to work properly, the pre-fetching module fetches the data as a whole; the query should not define any constraints. On the other hand, the requests from clients contain some query constraints. The federator side handles these queries and their constraints. Queries are processed by using parser techniques and XPATH (Clark and DeRose, 1999) queries over the pre-fetched data.

The pre-fetching module is composed of two components. One is "timer", defining the periodicity that pre-fetching will be running on, and the other is "task", defining what to fetch. The periodicity should not be less than the data transfer time. The periodicity for data fetching is defined under the considerations of data characteristics and the developer's experience on the domain specific application. Here is the pre-fetching task defined in a pseudo code:

```
public void task() {
        //List all the data sets to be pre-fetched
        dataList = getDataNamesTobePrefetched();
         // Define the storage locations
        String tempDatastore = applpath + "/whereDataTobePlaced";
        String stableDatastore = applpath + "/whereDataTobeServedFrom";
        //Fetching all the data in CDM format (GML)
        fd.FetchDataWithStreaming( NBip,NBport,NBtopic,
                          wfs_address,tempDatastore,CDMdataList );
        // move the data to the stable storage
        fd.moveData(tempDatastore, stableDatastore);
}
```

Here is the sample timer: timer. schedule (task, 0, 40000);

Timer schedules the specified task for repeated fixed-delay execution, and subsequent executions take place at regular intervals.

There are three concerns in developing an efficient pre-fetching architecture. The first one is regarding the limited storage capacity for a node. In other words, the federator's storage capacity constrains the size of the pre-fetched data. In the future, we plan to use Apache Hadoop (distributed data storage framework) (Apache Hadoop Project, 2007) to overcome the storage limitations for the federator. The second concern is that system might possibly have inconsistent datasets in a short period of time. This is also related to the characteristics of data. Some archived data are updated so often that they look like real-time data. In that case, pre-fetching becomes unfeasible and cannot be benefited. Our criterion whether this technique is applicable or not depends on two measurements. One is the minimum time required to fetch a whole critical data from the source and another is the time periodicity in which dataset is updated in its storage. If the dataset changes less than a time period in which whole critical dataset is fetched, then the dataset is called frequently changing. Therefore, it can be concluded that this technique is not applicable for the frequently changing datasets. The third, and final, concern is about the availability of the data sources. When the originating data server is down and the federator cannot perform synchronization regularly, then the federator will have to serve its clients from the last pre-fetched, possibly outdated, data sets. When the data server comes to live, the federator starts performing its pre-fetching.

Performance evaluation of pre-fetching

The proposed pre-fetching technique is tested on real-world Pattern Informatics earthquake geo-science application . Pattern Informatics is an earthquake forecasting application developed at University of California at Davis and uses archived earthquake seismic records stored at the WFS as feature collections encoded in GML.

Every machine (on which servers are deployed) in the test setup has 2 Quad-core Intel Xeon processors with 8 GB of memory and running at 2.33 GHz. They are operating Red Hat Linux ES. WFS, WMS, federator and Map Client tools are deployed in separate machines in Kocaeli University's Local Area Network (LAN). Figure 4 illustrates the pre-fetching technique and can be considered as a test setup. In case of serving client requests from original resources (on-demand fetching approach), one end is the database and other end is the user. This is shown with the broken arrows in the Figure 4. In case of serving client requests from pre-fetched data (pre-fetching approach), one end is the federator and other end is the user, which is shown as thick solid arrows in the Figure 4. Thin solid lines are pre-fetching module which is independent of the application runtime and does not affect the response times of clients' on demand requests.

The pre-fetched data size does not change as long as the data in the originating source remain the same. For the test case scenario, pre-fetching is applied on earthquake seismic data records. Seismic data records are fetched and stored in the federator's local file system in GML form and periodically synchronized with the originating data source. Every time a request comes (even for 1 KB of data) from a client, federator scans whole GML file to extract the requested part. The size of GML is about 127 MB, which represents the whole data in its originating source (WFS). For the 127 MB of GML data, parsing and data extraction time is average 5.69 s. This value is a part of overall response time is case of using pre-fetched data. As the query size increases the ratio of this value in overall response time decreases. In case of serving clients from pre-fetched dataset, response time does not include data transfer time. This explains why, in, response time does not change considerably depending on the increasing query payload size as indicated in the dotted line in Figure 5.

Figure 5 shows that the performance results for pre-fetching and on-demand fetching seem very close until a specific threshold data size (around 500 KB). After that threshold value, performance difference increases significantly. For example, for 10 MB of data, the pre-fetching is about 7 times faster than on-demand fetching. also shows that the larger the data size the higher the performance gains when the pre-fetched dataset is used by means of the federator. Figure 5 also shows that on-demand fetching curve linearly increases by the increasing data sizes. This is because of WFS used in the test scenario. When the request comes; WFS parse the query, access the databases, get the feature data and convert the result sets to GML data. Since OGC services are stateless and WFS used in the system do not use any performance enhancing techniques,

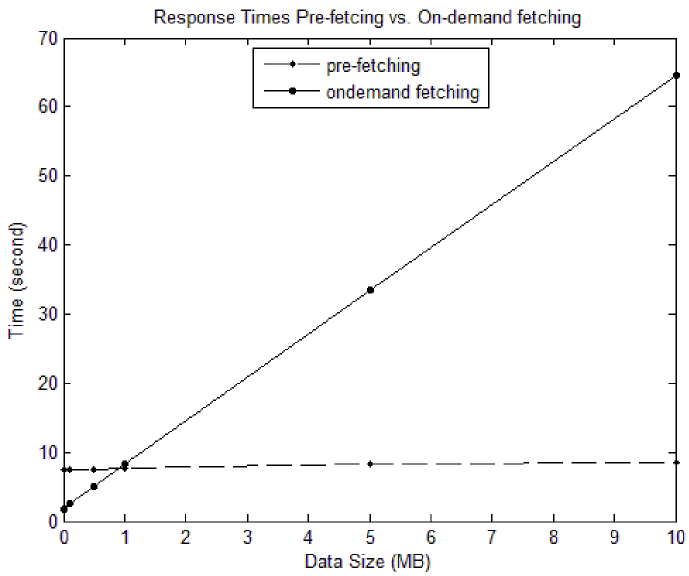

Figure 5. Performance comparison of pre-fetching and ordinary way.

response time increases linearly.

SUMMARY AND FUTURE WORKS

Open GIS standards define standard data services that provide data in standard formats (common data models) with the corresponding capability metadata (about the data+services) and the standard service API. These service properties and standardization make them composable. We have introduced a federator, which federates the standard GIS web services components through aggregation of their capabilities' metadata and presents a single database image to the user, which is defined in federator's aggregated capability metadata.

The proposed framework federates service-oriented GIS web services and addresses interoperability issues by integrating web services with Open Geographic Standards. The framework provides interoperability at data, service, and application levels, and integrates geo-data sources into geo-science grid applications seamlessly. We have also outlined our research and implementations to build a distributed geophysical SOA enabling fine-grained information/knowledge presentations in multilayered map images through a novel federator architecture. The proposed architecture is based on a XML-encoded common data model, standard GIS web service components, and a federator service. We have addressed several issues related to archival data access and processing from a single access point, and investigated novel techniques to federate distributed GIS web services.

Federator approach is a kind of centralized approaches over distributed web services. It has some advantages in terms of performance through the applications of caching, load balancing, sessionful service policies, etc. However, it might cause data inconsistencies due to having centralized characteristics and keeping data at multiple geographically distributed places simultaneously. Another drawback of the federated systems is single point of failure issue. However, this risk is present in most of the "ultimate" and successful platforms, including: GoogleMap/Earth; Cloud Computing platforms, etc. The system reliability will be enhanced by adopting backup servers and modular software components.

Even though it has some performance and easy to use advantages for the users, federated approach is not going in the direction of an efficient and distributed environment. Brokering approach on the other hand can be considered for further developments of the project. It offers a greater level of flexibility than other architectural solutions. A broker can implement added-value functionalities related to data discovery, access and semantic expansions.

In the proposed system, we use a static approach to create application-specific hierarchical data layers in the federator's aggregated capability metadata. That is, the federated capabilities file defining the data and corresponding data sources are not allowed to be changed or updated after the application runs. It would be useful for the system to automatically create, deploy, and update the required layers and to add the corresponding services dynamically. To enable such quality of services, we plan to enhance the system through data registry and discovery services. Federator automatically looks up the registry service and finds out the required data layers and updates its aggregated metadata automatically. To do that registry services need to keep metadata for the data sets, as well as service API. In addition, we plan to enhance the system with Web 2.0 (Shuen, 2008) standards. The concept of services has been changing in recent years. The new generation service concept is letting people collaborate and share information online. This concept is represented with the term Web 2.0. It would be useful to see if the proposed framework can be extended with Web 2.0 and what kind of outcomes can be obtained.

The pre-fetching technique gives the best performance outcomes for archived data, but may cause inconsistency depending on the fetching and data updating periodicity in their originating sources. It is not easy to apply the pre-fetching approach on dynamically changing data sets used in some applications, such as early warning systems. The criterion for selecting the technique to apply depends on two measurements. One is the minimum time required to fetch a whole data from the source and another is the time periodicity in which the data are updated in its storage. If the former is smaller than the latter, it is not feasible to apply the proposed pre-fetching technique. Another concern with the pre-fetching is the limited storage capacity of the federator. In the future, this problem can be solved by using a distributed file system such as Apache Hadoop. The work presented in this paper was aimed towards problems in geo-science, and we believe it can be adopted for other scientific domains, if the data in those domains can be spatially defined. The data space may be real space (such as astronomy), or takes a part in a parameter space (e.g. chemical spaces). However, the effects of domain-specific requirements are not well understood. We think that it is important to explore how the common data standards, such as GML, and service standards, such as WFS or WMS, can be adapted to these different domains.

REFERENCES

Al-Turjman FM, Hassanein HS, Oteafy SMA (2011). Towards Augmenting Federated Wireless Sensor Networks. Procedia Computer Science. 5:224-231.

Anh TL, Villalobos J, Estublier J (2003). Multi-Level Composition for Software Federations. Electronic Notes in Theoretical Computer Science. 82(5):164-173.

Apache Hadoop Project (2007). The Apache Software Foundation. http://hadoop.apache.org/. Accessed 24/12/2010 2010.

Aydin G, Sayar A, Gadgil H, Aktas MS, Fox GC, Ko S, Bulut H, Pierce ME (2008). Building and Applying Geographical Information Systems Grids. Concurrency and Computation: Practice and Experience. 20(14):1653-1695.

Batcheller JK (2008). Automating geospatial metadata generation—An integrated data management and documentation approach. Computers & Geosciences. 34(4):387–398

Beaujardiere Jdl (2004). OGC Web Map Service Interface. 1.3 edn. Open GIS Consortium Inc. (OGC).

Booth D, Haas H, McCabe F, Newcomer E, Champion M, Ferris C, Orchard D (2004). Web Services Architecture. World Wide Web (W3C),

Butenuth M, Gösseln Gv, Tiedge M, Heipke C, Lipeck U, Sester M (2007). Integration of heterogeneous geospatial data in a federated database. ISPRS Journal of Photogrammetry & Remote Sensing. 62(5):328–346.

Chao W, Guo Y, Zhou B (2012). Social networking federation: A position paper. Computers and Electrical Engineering. 38(2):306-329.

Christensen E, Curbera F, Meredith G, Weerawarana S (2001). Web Services Description Language (WSDL) World Wide Web Consortium (W3C),

Clark J, DeRose S (1999). XML Path Language (XPath) Version 1.0.

Cox S, Daisey P, Lake R, Portele C, Whiteside A (2003). OpenGIS® Geography Markup Language (GML) Encoding Specification. 3.0 edn. Open Geospatial Consortium (OGC),

Erradi A, Maheshwari P A(2005). Broker-Based Approach for Improving Web Services Reliability. In: IEEE International Conference on Web Services, 11-15 July 2005. IEEE, pp. 355-362.

EuroGEOSS (2013). http://www.eurogeoss.eu/default.aspx. Accessed June 20, 2013

Evans JD (2003) Web Coverage Service (WCS), Version 1.0.0.

Fonseca FT, Egenhofer MJ, Agouris P, Cámara G (2002). Using Ontologies for Integrated Geographic Information Systems. Transactions in GIS. 6(3):231-257.

Gray J, Szalay AS, Thakar AR, Kunszt PZ, Stoughton C, Slutz D, vandenBerg J (2002). Data Mining the SDSS SkyServer Database. Microsoft.

Gruber TR (1993). A translation approach to portable ontology specification. Knowledge Acquisition. 5(2):199-220.

Huang Y, Cai G, Wang G (2011). Dynamic Service Composition Based on Federated Agents. Energy Procedia. 13:5536-5543.

ISO. (2008). http://www.isotc211.org/. Accessed 03/27/2008 2008.

Kolodziej K (2004). OpenGIS Web Map Server Cookbook. Open Geospatial Consortium Inc. (OGC).

Kreger H (2001). Web Services Conceptual Architecture (WSCA 1.0). IBM.

Lagoze C, Sompel HVd (2006). The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH).

Leal K, Huedob E, Llorente IM (2009). A decentralized model for scheduling independent tasks in Federated Grids. Future Generation Computer Systems. 25(8):840 -852.

Madsen P (2004). Federated identity and web services. Information Security Tech Report. 9(3):133-121.

Morocho V, Saltor F, Perez-Vidal L (2003). Schema Integration on Federated Spatial DB across Ontologies. In: Proceedings of the 5TH International Workshop on Engineering Federated Information Systems (EFIS2003), Coventry, UK, Jul 2003. pp. 63-72.

OGC (1994). http://www.opengeospatial.org/. Accessed 02/14/2008.

OGC (2012). WMS capability schema.

OnEarth (2007). NASA - Jet Propulsion Laboratories. http://onearth.jpl.nasa.gov. Accessed 03/15/2008.

Pallickara S, Fox G (2003). Narada Brokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. Paper presented at the ACM/IFIP/USENIX, Rio Janeiro, Brazil, June 2003.

Pautasso C (2009). RESTful Web service composition with BPEL for REST. Data and Knowledge Engineering. 68(9):851-866.

Peng ZR, Tsou MH (2003). Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks. John Wiley & Sons, New Jersey, USA.

Pierce ME, Fox GC, Aktas MS, Aydin G, Qi Z, Sayar. A (2008). The QuakeSim Project: Web Services for Managing Geophysical Data and Applications. Pure and Applied Geophysics (PAGEOPH). 165(3-4):635-651.

Sheth A, Larson J (1990). Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Computing Surveys. 22(3):183-236.

Sonnet J (2005). Web Map Context Documents (WMC). Open Geospatial Consortium Inc. (OGC).

Tanenbaum AS (2008) Modern Operating Systems. Third edn. Pearson Prentice Hall, NJ, USA.

Tiampo KF, Rundle JB, Mcginnis SA, Klein W (2002). Pattern Dynamics and Forecast Methods in Seismically Active Regions. Pure and Applied Geophysics. 159(10):2429-2467.

Trnkoczy J, Stankovski V (2008). Improving the performance of Federated Digital Library services. Future Generation Computer Systems. 24(8):824-832.

Trnkoczy J, Turk Ž, Stankovski V (2006). A grid-based architecture for personalized federation of digital libraries. Computer and Information Science. 30(3/4):139-153.

Vázquez C, Huedo E, Montero RS, Llorente IM (2010). Federation of TeraGrid, EGEE and OSG infrastructures through a metascheduler. Future Generation Computer Systems 26(7):979-985.

Vermeer MWW, Apers PMG (1998) Specifying Global Behaviour In Database Federations. Information Systems 23 (3/4):217-233.

Vretanos PA (2002). Web Feature Service Implementation Specification.

W3C. (2008). http://www.w3c.org.

Wiederhold G (1995). Mediation in Information Systems. ACM Computing Surveys (ACM). 27(2):265-267.

Glossary

API: Application Programming Interface

Bbox: (Bounding box) (OGC-defined) A geo-data attribute to define 2-dimensional ranges in rectangular shapes (minx, miny maxx, maxy).

Capability metadata: A metadata about the data and services together. It includes information about the data and corresponding operations with the attribute-based constraints and acceptable request/response formats.

Context document: A context document is an XML-encoded description of map layer compositions. It is defined as a part of WMC specifications. It includes information about layer descriptions in terms of server bindings and URLs, projection system, available bounding boxes etc.

Fine-grained: Smaller components of which the larger ones are composed. It can also be described as dense or compact in structure or texture.

Geographic features: Earth-related data definitions, such as rivers, lakes, earthquake seismic records, and so on

GIS: (Geographic Information Systems) represent the main technology motivating interest in developing spatially enabled system. GIS also provide convenient mechanisms for analyzing and visualizing geographic data.

GML: (Geographic Markup Language) (OGC-defined) An XML grammar defined by OGC to model geo-data in commonly accepted widely used standard. It enables datasets to be easily accessed, integrated and analyzed across the heterogeneous and autonomous organizations.

HTTP: (Hyper Text Transport Protocol) A stateless internet protocol for transferring hyper text data between server and client.

ISO/TC211: A standard technical committee formed within ISO for geographic information.

Metadata: Simply described as data about data. Metadata describes other data.

NaradaBrokering: A P2P overlay network developed at Indiana University, Community Grids Labs. It consists of broker nodes and based on topic-based publish subscribe paradigm developed as an overlay network.

OGC: (Open Geospatial Consortium) is an international standards (not-for-profit) development consortium. It has 365+ industry, government, and university member. It’s standards are publicly available and widely-used in GIS domain.

PI: (Pattern Informatics) An earthquake Geo-science application developed at UC-Davis. It defines method using observational data to identify the existence of correlated regions of seismicity.

Raster data: The raster data model is used to model spatial phenomena that vary continuously over a surface and that do not have discrete dimension. Examples of this are elevation, temperature, rainfall and noise levels.

Rendering: Rendering is a process for creating an image from raw datasets having geometric attributes. Rendering is done by using computer programs.

Service Oriented Architecture: SOA

SOA: (Service Oriented Architecture) SOA is basically a collection of communicating services, and organized as distributed systems. The communication is accomplished through message passing or some other means.

SOAP: (Simple Object Access Protocol) A simple protocol specification for exchanging messages between clients and servers in web services. It consists of three parts: an envelope, encoding rules and principles for representing calls and responses.

Vector data: Vector data uses points and their (X,Y) coordinates to represent spatial features. Point sets come together in a mathematical model and form lines, line-strings and polygons to represent spatial data.

Web Services: Web services are basically services available on the web. They are identified with URI and their binding information and interfaces (such as types and numbers of request and response parameters) are defined by using XML. It also uses XML-encoded protocol called SOAP for client-server communications.

WFS: (Web Feature Service) (OGC-defined) provides standards for creating a service to serve any data in an XML-encoded standard data format (GML) with standard service interfaces. Geo-data is described with its various attributes in GML and WFS allows attribute-based queries with standard.

WSDL: (Description Language for web service) is s a language to define web services. It is XML-based, and service descriptions can be accessed through HTML.

WMC: (Web Map Context) (OGC-defined) One of OGC specifications for describing how a particular overlaying (combination) of map images from distributed Web Map Servers can be described in a portable, platform-independent format to store or transmit between clients. This description is known as a "Web Map Context Document," or simply a "Context." Presently, context documents are primarily designed for WMS bindings.

WMS: (Web Map Service) (OGC-defined) Creates digital maps from abstract datasets retrieved from Web Feature Services (WFS). Abstract datasets carry some geometric attributes to be drawn as digital images.

XML: (Extensible Markup Language): XML is a W3C-proposed standard, and enables interoperable way to represent documents for flexible processing. It is based on SGML (ISO 2008), which is a standardization for markup languages.