Unified Data Access/query over Integrated Data-views for Decision Making in Geographic Information Systems

Ahmet Sayar [a,b.] *, Geoffrey C. Fox [a,b,c], and Marlon E. Pierce [a]

[a] Community Grids Laboratory, Indiana University
501 N. Morton Suite 224, Bloomington, IN 47404 USA
{asayar, gcf, mpierce}@cs.indiana.edu
[b] Computer Science Department, School of Informatics, Indiana University
[c] Physics Department, College of Arts and Sciences, Indiana University

KEY WORDS: Decision making, GIS, federation, Web Services, information rendering, Service Oriented Architecture.

Abstract

Geographic information is critical for building disaster planning, crisis management and early-warning systems. Decision making in Geographic Information Systems (GIS) increasingly relies on analyses of spatial data in map-based formats. Maps are complex structures composed of layers created from distributed heterogeneous data belonging to the separate organizations. This chapter presents a distributed service architecture for managing the production of knowledge from distributed collections of observations and simulation data through integrated data-views. Integrated views are defined by a federation service ("federator") located on top of the standard service components. Common GIS standards enable the construction of this system. However, compliance requirements for interoperability, such as XML-encoded data and domain specific data characteristics, have costs and performance overhead. We investigate issues of combining standard compliance with performance. Although our framework is designed for GIS, we extend the principles and requirements to general science domains and discuss how these may be applied.

* Corresponding author.

1. Introduction

The World Wide Web and its associated Web programming models have revolutionized accessibility to data/information sources. At the same time, numerous incompatible data formats, data heterogeneity (both the data types and storage formats), and machine un-readability of this data have limited data integration and federation. The seamless integration and sharing of data from distributed heterogeneous data sources have been the major challenges of information system communities and decision support systems. In order to be able to integrate and share data/information, data sources need to be in interoperable formats and provide standard service interfaces that interact with standard message formats and transport protocols. The interoperability issues have been studied by many public/private organizations over the last two decades at the data and service levels. Among these are the Web Service standards (WS-I) for cross-language, platform and operating systems, and International Virtual Observatory Alliance (IVOA) and Open Geospatial Consortium (OGC) for defining domain specific data model and online service definitions in Astronomy and Geographic Information Systems, respectively. We are now at the point that we can put this work into practice. Moreover, the merging of GIS standards with Web Service standards enables us to investigate the integration of geophysical application services with geographic data services.

Geographic information is critical to effective and collaborative decision making for building disaster planning, crisis management and early-warning systems. Decision making in Geographic Information Systems (GIS) increasingly relies on analyses of spatial data in map-based formats. Maps are complex structures composed of layers created from distributed heterogeneous data and computation resources belonging to separate virtual organizations from various expert skill levels.

Map-based services are both a crucial application for decision making and an interesting area for researching problems in distributed service architectures. Our proposed federated service-oriented information system framework supports collaborative decision making over integrated data views, described in layer-structured hierarchical data. The users access the system as though all data and functions come from one site. The data distribution and connection paths stay hidden and formulated as hierarchical data defined in federator’s capability metadata. The users access the system through integrated data-views (maps) with the event-based interactive mapping display tools. Tools create abstract queries from users’ actions through action listeners and communicate with the system through federator.

Our framework is based on standard GIS Web Service components that provide standard service interfaces defined by Open GIS standards and are developed in accordance with the Web Service Interoperability Organization’s standards ("WS-I," 2002). The federation service (“federator”) combines standard GIS data services through aggregating their standard capability metadata and enables unified data access/query. Moreover, although the proposed framework is designed for GIS, our experiences with GIS have shown that it can be generalized to many application areas. We provide the overview architecture in Section 3. We give blueprint for this general architecture in terms of principles and requirements in Section 5, with example applications for chemistry and astronomy.

GIS is particularly useful in emergency early-warning, preparedness, and response systems with applications in homeland security and natural disasters (earthquake, flood, etc).

Such applications demand good performance. However, because of the distributed system's nature, interoperability requirements (compliance costs), characteristics of geo-data (large and variable-sized), and time-consuming rendering processes, performance and responsiveness stand as the toughest challenges in distributed modern GIS applications. Thus, despite the advantages of Web Service federation, we have to go beyond naïve implementations to address problems in scalability and performance. This has led us to investigate novel strategies including performance enhancing client-based caching, load balancing, and parallel processing techniques through attribute based query decomposition. We discuss these issues in Section 4.

2. Background

Geographic Information Systems (GIS) (Peng & Tsou, 2003) are systems for creating, storing, sharing, analyzing, manipulating and displaying geospatial data and the associated attributes. GIS introduce methods and environments to visualize, manipulate, and analyze geospatial data. The nature of the geographical applications requires seamless integration and sharing of spatial data from a variety of providers.

The general purpose of GIS is modeling, accessing, extracting and representing information and knowledge from the raw geo-data. The raw data is collected from sources ranging from sensors to satellites and stored in databases or file systems. The data goes through the filtering and rendering services and is ultimately presented to the end-users in human recognizable formats such as images, graphs, charts, etc. GIS is used in a wide variety of tasks such as urban planning, resource management, emergency response planning in case of disasters, crisis management and rapid response.

Over the past two decades, GIS has evolved from traditional centralized mainframe and desktop systems to collaborative distributed systems. Centralized systems provide an environment for stand-alone applications in which data sources, rendering and processing services are all tightly coupled and application specific. Therefore, they are not capable of allowing seamless interaction with the other data or processing/rendering services. On the other hand, the distributed systems are composed of autonomous hosts (or geographically distributed virtual organizations) that are connected through a computer network. They aim to share data and computation resources collaborating on large scale applications.

Modern GIS requires data and computation resources from distributed virtual organizations to be composed based on application requirements, 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.

The primary function of a GIS is to link multiple sets of geospatial data and graphically display that information as maps with potentially many different layers of information (see Figure 1). Each layer of a GIS map represents a particular "theme" or feature, and one layer could be derived from a data source completely different from the other layers (Koontz, 2003). As long as standard processes and formats have been arranged to facilitate integration, each of these themes could be based on data originally collected and maintained by a separate organization. Analyzing this layered information as an integrated entity (map) can significantly help decision makers in considering complex choices.

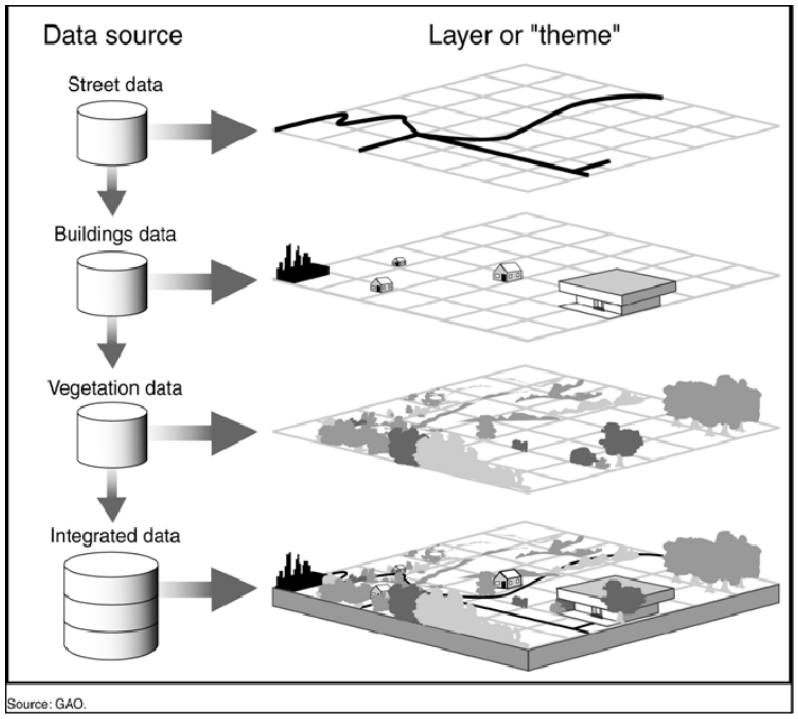

Figure 1: Layered display – a map is composed of distributed multiple set of layers. The figure is from (Koontz, 2003).

Open GIS Standards and GIS Web Services

In order to achieve such a layered display (Figure 1) whose layers come from autonomous, heterogeneous data resources provided by various virtual organizations, the domain-specific common data models, standard service functionalities and interfaces need to be described and widely adopted. There are two well-known and accepted standards bodies in the GIS domain aiming at these goals. These are Open Geospatial Consortium ("OGC," 1994) and the Technical Committee tasked by the International Standards Organization (ISO/TC211). The standards bodies' aims 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/electronic 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 & Tsou, 2003). On the other hand, 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. They are compatible with each other. ("JAG," 1999)

OGC's standards definition for data model (Geographic Markup Language -GML) (Cox, Daisey, Lake, Portele, & Whiteside, 2003) and online data services are well-known and widely adopted. As more GIS vendors are releasing compatible products and more academic institutions use OGC standards in their research and implementations, OGC specifications are becoming de facto standards in GIS community, and GML is rapidly emerging as the standard XML encoding for geographic information.

The Web Map Service (WMS) (Beaujardiere, 2004; Kolodziej, 2004) and the Web Feature Service (WFS) (Vretanos, 2002) are two major services defined by OGC for creating a basic GIS framework enabling information rendering of heterogeneous data sources as map images. WMS is the key service to the information rendering/visualization in GIS domain. WMS produces maps from the geographic data in GML provided by WFS. It also enables attribute/feature based data querying over data display by its standard service interfaces. OGC's WFS implementation specification defines interfaces for data access and manipulation operations on geographic features. Via its standard service interfaces, a web user/client can combine, use and manage geo-data from different sources by invoking several standard operations (Vretanos, 2002). By creating an interoperable standard GIS framework as a result of adopting Open GIS standards (using GML and standard online services WMS and WFS), we open the door of interoperability to this growing community.

In addition to the domain-level interoperability and extensibility, information systems need cross-language, operating system and platform interoperability to enable data sharing/federating and analysis over autonomous heterogeneous resources provided by various virtual 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. Grid computing (Foster & Kesselman, 2004; Fox, 2004) has a converging Web Service-based architecture. By implementing Web Service versions of GIS services, we can integrate them directly with scientific application Grids (Atkinson et al., 2005; Aydin et al., 2008).

A Web Service is an interface that describes a collection of operations that are network accessible through standardized XML messaging (Kreger, 2001). Web Services collectively are a software system designed to support interoperable machine-to-machine interaction over a network. A typical service has an interface described in a machine-processable format called the Web Service Description language (WSDL) (Christensen, Curbera, Meredith, & Weerawarana, 2001). 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. Representational State Transfer (REST) (Fielding & Taylor, 2002; Khare & Taylor, 2004) is a variation of this architecture that replaces WSDL with standard HTTP operations (GET, POST, PUT, DELETE). REST can be used to transmit SOAP messages as well as other formatted transmissions such as RSS, ATOM, or JSON.

The major difference between Web Services and other component technologies is that Web Services are accessed via the ubiquitous Web protocols such as Hypertext Transfer Protocol (HTTP) and Extensible Markup Language (XML) instead of object-model-specific protocols such as Distributed Component Object Model (DCOM) (Redmond, 1997) or Remote Method Invocation (RMI) ("RMI," 2004) or Internet Inter-Orb Protocol (IIOP) (Kirtland, 2001). One typically builds services to be stateless and places the distributed system state in a single state machine that aggregates clients to services. This simplifies several well-known problems in distributed object systems (such as fault tolerance), enabling Web Service-based systems to have better scalability.

Adopting GIS Open Standards to Web Service standards and implanting Web Service versions of standard GIS services permit applications to span programming languages, platforms and operating systems. It also enables application developers to integrate the third party geospatial functionality and data into their custom applications easily.

3. Federating GIS Web Service Components: Distributed Service Architecture

Our federator framework provides an infrastructure for understanding and managing the production of knowledge from distributed observation, simulation and analysis through integrated data-views in the form of multi-layered map images. Infrastructure is based on a common data model, OGC compatible standard GIS Web-Service components and a federator. The federator is actually an extended Web Map Server (WMS) federating GIS services and enabling unified data access/query and display over integrated data-views.

By federation, we mean providing one global view over several data sources that are 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 to the queries posed on the global schema); and the third is the common presentation model of data sources, i.e. mapping of 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 we take them as granted by applying Open Geographic Standards specifications for data models (GML) and online services (WMS and WFS).

Our extended standard GIS Web Service components are integrated into the system through a federator, which is actually a WMS that is extended with capability-aggregating and stateful service capabilities to enable high performance support for responsive GIS applications. This section presents view-level information presentation through federation of standard GIS Web Service components. The framework is designed for GIS domain; however we present the generalization architecture in terms of principles and requirements in Section 5.

Geo-data and Integrated Data-view

Geo-data is provided by geographically distributed services from many different vendors in different formats, stored in various different storage systems and served through heterogeneous service API and transport protocols. The heterogeneity of geographic resources may arise for a number of reasons, including differences in projections, precision, data quality, data structures and indexing schemes, topological organization (or lack of it), set of transformation and analysis services implemented in the source.

The OGC and ISO/TC-211 have tried to address these issues. The specifications for data models and online service descriptions define compliance requirements at data and service API level. In brief, according to the standard specifications there are three general groups of data services: Web Map Services, Web Feature Services, and Web Coverage Services (Evans, 2003). WMS provides rendered data in maps in MIME/image formats; WFS provides annotated feature-vector data in XML-encoded GML, and WCS provides coverage data as objects or images. Since they have standard service API and capability metadata about their services and data, they can be composed, or chained, by capability exchange and aggregation through their common service method called *getCapability*.

This idea has inspired us to develop an infrastructure for creating and managing the production of knowledge from distributed observation, simulation and analysis through integrated data-views in the form of multi-layered map images (see Figure 2) enabling unified data access/query/display from a single access point. As shown in the figure, the geo-data is accessed through a federator service, and data is always kept in its originating resources. They are integrated into the system with user's on-demand querying (just-in-time federation). This enables easy data maintenance and autonomy.

There is a three-level hierarchy of data. At the top layer, there is a federator service providing human comprehensible data display in multi-layered map images. The federators compose the data from the standard data services located at the middle level (WMS and WFS). The bottom levels are consisted of heterogeneous data sources integrated into the system through standard data services at the middle level. WMS are rendering and displaying services, and WFS are mediator/adaptor services providing heterogeneous data in common data model, and provide resource and data specific query/response conversions. They provide heterogeneous data in common data model with standard service interfaces as defined in Open GIS standards.

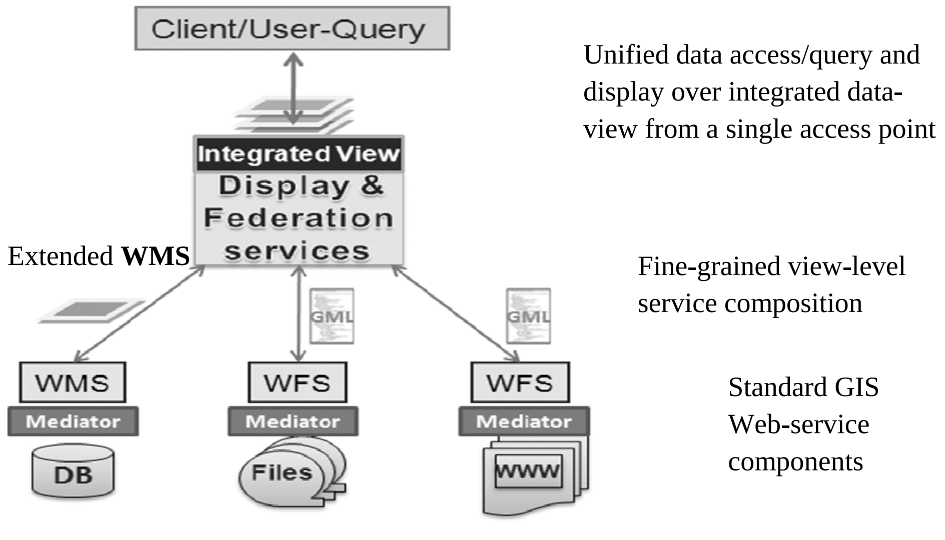
Client/User-Query
Unified data access/query and display over integrated data-view from a single access point
Integrated View
Display & Federation services
Extended **WMS**
Fine-grained view-level service composition
GML
GML
WMS
WFS
WFS
Mediator
Mediator
Mediator
DB
Files
WWW
Standard GIS Web-service components

Figure 2: Data life-cycle and integrated data-view creation.

Heterogeneous data sources, which form the bottom layer of the hierarchy, are integrated into the system through mediators. Mediators provide an interface for the local data sources and play the roles of connectors between the local source and the global one. The principle of integration is to create non-materialized view in each mediator. These views are then used in the query evaluation. Mapping rules that express the correspondence between the global schema

(GML) and the data source ones are essential. The problem of answering queries is another point of the mediation integration – a user poses a query in terms of a mediated schema (such as *getFeature* to WFS), and the data integration system needs to reformulate the query to refer to the sources. Therefore, an information integration architecture emerges based on a common intermediate data model (GML) providing an abstraction layer between legacy storage structures and exposed interfaces. In our system, we use OGC to enable these interfaces. GML provides a semantic abstraction layer for data files and is exposed through a higher level data delivery service called WFS.

There are several advantages in adopting the approach shown in Figure 2. The mediators not only enable data sources integrated into the system 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. The integration process does not affect the individual data sources' functionality. These data sources can continue working independently to satisfy the requests of their local users. Local administrators maintain control over their systems and yet provide access to their data by global users at the federation level.

The remainder of the chapter focuses on upper levels (view-level) of dataflow and query refinements illustrated in Figure 2. Since the OGC's standard services are developed as Web Service components, they can be chained/orchestrated with Web Service workflow tools, such as Kepler (Ludäscher et al., 2006) and Taverna (Turi, Missier, Goble, Roure, & Oinn, 2007), but we do not attempt to delve into those issues in this chapter. We instead focus on the definition of service compositions and integrated data views as presented in the following sections. Workflow execution abstraction is a higher-level abstraction than the capability metadata federation that we investigate.

Hierarchical Data Definition as Multi-layer Maps

Hierarchical data is defined as an integrated data-view in the federator's capability metadata. It actually defines a static workflow starting from the federator and ending at the original data sources (WFS serving GML or WMS serving map layer images). The services are linked through the reference-tags defined in their capability metadata. Decision makers' interactions with the system are carried over the integrated data views through event-based interactive map tools. Integrated data-views are defined in the hierarchical data format as explained below:

Map -> Layer -> Data {GML / binary images} ->Raw data (any type).

A map is an application-based, human-recognizable, integrated data display and is composed of layers. A layer is a 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.

Different applications need different maps that are composed of different data layers in different numbers and combinations (Figure 3). Maps are multi-layered, complex structures whose layers come from distributed heterogeneous resources and are rendered from any type of data. This type of multi-layered map image is defined and managed in the federator with utilization of its cascading WMS properties and inter-service communication between the components.

Federation Framework

Our federation framework is built over a service-oriented GIS framework and its components (WMS and WFS). Federation is based on federating capabilities metadata from the GIS Web Services components. Capabilities are aggregated through inter-service communication using standard service interfaces. We do not define common data models, online standard service components and their capability metadata definitions in GIS. These are already defined by Open Geographic Standards (OGC). We instead have developed the components according to the open standard specifications, and applied them to our proposed information system framework by defining required extensions at implementation and application levels in compliance with WS-I Web Service standards (Sayar, Pierce, & Fox, 2005). They also serve as a test bed for implementing and testing general concepts in service architectures.

This section presents a federation framework based on common data models (GML), standard Web Service components, federator and event-based interactive decision making tools over integrated data views in the form of multi-layered map images. The general architecture is illustrated in Figure 3. This figure presents the proposed federation framework with a sample application using earthquake seismic data (from WFS) and NASA satellite map images (from WMS). WMS is the NASA OnEarth server located at the NASA Jet Propulsion Laboratory (JPL) ("OnEarth," 2007) and WFS is located at Community Grids Labs (CGL) at Indiana University.

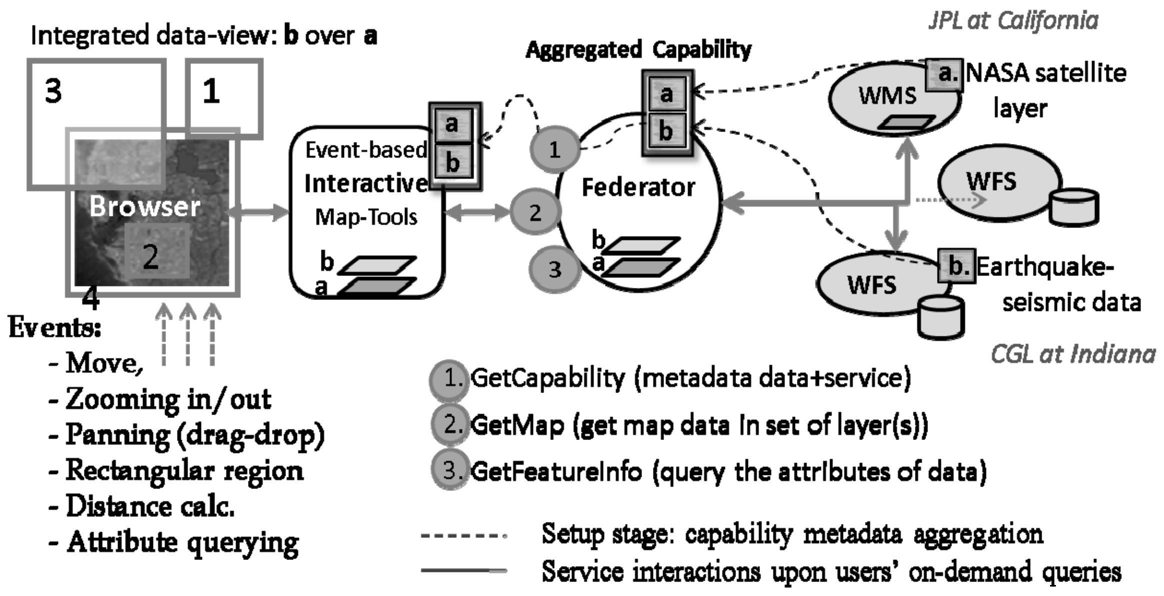
Integrated data-view: b over a
3
1
Browser
2
4
Events:
- Move,
- Zooming in/out
- Panning (drag-drop)
- Rectangular region
- Distance calc.
- Attribute querying
Event-based Interactive Map-Tools
a
b
Aggregated Capability
Federator
1
2
3
JPL at California
WMS
a. NASA satellite layer
WFS
WFS
b. Earthquake-seismic data
CGL at Indiana
1. GetCapability (metadata data+service)
2. GetMap (get map data in set of layer(s))
3. GetFeatureInfo (query the attributes of data)
Setup stage: capability metadata aggregation
Service interactions upon users' on-demand queries

Figure 3: Federated GIS framework.

The framework enables users (i.e., decision-makers) to access the system as though all the data and functions come from one site. The data distribution and connection paths stay hidden and formulated as hierarchical data defined in federator's capability metadata. The users access the system through integrated data-views (maps) with the event-based interactive mapping display tools (Sayar, Pierce, & Fox, 2006). These tools transform the users' actions into abstract queries through action listeners and enable client interaction with the system via the federator.

As shown in Figure 3, the federator is actually a WMS (Kolodziej, 2004) with extended capabilities and functionalities. These can be summarized as aggregating capability metadata from distributed standard GIS services and orchestrating/synchronizing requests and responses over the composition of data services referenced in aggregated capability metadata. The federator enables stateful service access over the stateless GIS Web Service components, and results in a better performance for responsive GIS systems. These issues are addressed in Section 4.

The federation framework is based on a two-stage process. The first stage is the setup (or initialization) stage. The second stage is the application run-time stage. In the setup stage, an integrated data-view (in the form of multi-layered map image) is defined in the federator's aggregated capability metadata. The federator searches for standard GIS Web Service components (WMS or WFS) providing required data layers and organize them into one aggregated capability file (see the following section). This is shown as dotted lines in the Figure 3. There is no client/user interaction with the system in this first stage. In the second stage (*run-time stage*), a user/client interacts with the system through a browser that provides event-based interactive display and query tools over the integrated data-view. The second stage is illustrated with solid arrows in the figure.

Interactive information visualization tools provide researchers with capabilities to support discovery. We developed these tools for interacting with standard WMS providing OGC compatible online services such as *getMap*, *getFeatureInfo* and *getCapabilities*. Since the federator is also a WMS, clients still use *getMap* service interface to display multi-layered map images and/or query it through *getFeatureInfo* service interface. The system removes the burden of accessing each data source with ad-hoc query languages such as SQL for MySQL source, and enables interactive feature based querying besides displaying the data. It also enables easy data-maintenance and high degree of autonomy.

Service Federation through Capability Aggregation

Capabilities are metadata about the data and services and have an XML schema that is defined by Open Geospatial Consortium (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, re-usable 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 by service chaining.

The integrated data-view in multi-layered map images is defined in the federator's aggregated capability metadata. There are two major issues here: a) definition of aggregated capability metadata and b) definition of multi-layered map images.

As mentioned earlier, the federation framework is built over the standard GIS Web Service components, and the federator concept is inspired from OGC's cascading WMS definition (Beaujardiere, 2004). In this respect, the federator is actually a cascading WMS with extended capabilities. In the following sections, we describe how we apply OGC's ideas related to the service chaining and aggregation, and define multi-layered map images in the aggregated capability metadata.

Extending WMS as a Federator Service

The federator is actually a cascading Web Map Server. A cascading Web Map Server is a WMS that behaves like a client to other WMSs and like a WMS to other clients. It can receive input from other WMS (and WFS) and display layers from them. For example, a cascading Web Map Server can aggregate the contents of several distinct map servers into one service. Furthermore, it can even perform additional functions such as output format conversion or coordinate transformation on behalf of other servers.

There are two possible ways to chain the services to be able to create a federator framework and application specific hierarchical data in integrated data-view. One is extending the WMS capability file by giving the reference to the service access points providing the required layer (WMS) and/or feature data (WFS). Another way is using Web Map Context's standards defining chaining in a context document (described below). In any case, we utilize the cascading WMS definitions to develop a federator providing information/knowledge in multi-layered map images.

i. Federating through context document:

OGC's WMS and WFS services are inherently capable of being cascaded and chained in order to create more complex data/information. In order to standardize these issues, OGC has introduced the Web Map Context (WMC) (Sonnet, 2005) standard specifications. Before that, OGC recommended application developers to extend their services' capabilities for cascading. WMC is actually a companion specification to WMS.

The present context specification states how a particular grouping of one or more maps from one or more map servers can be described in a portable, platform-independent format for storage in a repository or for transmission 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. However, extensibility is envisioned for binding to other services.

A *context* document is structured using XML, and its standard schema is defined in the WMC specifications (Sonnet, 2005). A *context* document includes information about the server(s) providing layer(s) in the overall map, the bounding box and map projection shared by all the maps, sufficient operational metadata for client software to reproduce the map, and additional metadata used to annotate or describe the maps and their provenance for the benefit of end-users.

There are several possible uses for *context* documents besides providing chaining and binding of services. The *context* document can provide default startup views for particular classes of users. For example specific applications require a specific list of layers. The context document can store not only the current settings but also additional information about each layer (e.g., available styles, formats, spatial reference system, etc.) to avoid having to query the map server again once the user has selected a layer. Finally, the *context* document could be saved from one client session and transferred to a different client application to start up with the same context. In this document, we just focus on its binding functionalities.

ii. Federating through aggregated WMS capability:

This is another alternative approach to extend the WMS as a federator. It is based on extending the standard WMS capabilities file.

Data providing in WMS is called “layer” and defined in layer tags in capability metadata with attributes and features according to the standard WMS capability schema (Beaujardiere, 2004). Service chaining 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. 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 set it to 1. The default value of cascading is 0 (Kolodziej, 2004).

In order to illustrate service federation, we give a real geo-science application as an example. In the Pattern Informatics (PI) application (Tiampo, Rundle, Mcginnis, & Klein, 2002), decision makers need to see earthquake forecast values and seismic data records plotted on satellite map images. Satellite map images are provided by NASA OnEarth project’s WMS at the NASA Jet Propulsions Laboratory, and earthquake seismic data records are provided from WFS at the Community Grids Labs (CGL) at Indiana University. The federator aggregates these services’ standard capability metadata and creates an aggregated one as if those data sets are its own. 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 federator’s aggregated capability metadata.

4. High-performance Support in Distributed Geo-Data Rendering

General Performance Issues in Interoperable Service-oriented Geographic Information Systems

Distributed GIS systems typically handle a large volume of datasets. Therefore the transmission, processing and visualization/rendering techniques need to be responsive to provide quick, interactive feedback. There are some characteristics of GIS services and data that make it difficult to design distributed GIS with satisfactory performance.

In order to provide interoperable and extensible framework, we have adopted domain-specific standard specifications for data model (GML) and online services from OGC, and Web Services specifications from WS-I ("WS-I," 2002). However, these adoptions degrade the performance even more for large-scale applications because using XML-encoded data models and Web Services' XML-based SOAP protocol introduces significant processing overhead. These issues and proposed enhancement approaches are presented in the following sections. The aim is to combine compliance requirements with competitiveness and to create a responsive information system framework providing map images for interactive decision making tools.

Distributed Nature of Data

The data ownership issues (that is, various data provided by geographically distributed various virtual public/private organizations) and large data volumes make it infeasible to put all geospatial data into one large data center. In addition, the computational resources associated with those data centers are naturally distributed. Furthermore, decision making requires these distributed heterogeneous data sources to be shared, and represented/rendered to extract useful knowledge giving sense to anybody joining the decision making process. Although we concentrate on the performance issues related to compliance requirements such as using XML-encoded data model GML and Open GIS compatible Web Service components, throughout the section we touch upon the general issues briefly mentioned above.

Using Semi-structured Data Model

GML is the data modeling language for OGC specifications. GML carries content and the presentation tags together with the core data. This enables the data sources to be queried and displayed together (i.e., map images interactively query-able through interactive map tools). Querying and displaying data in the GML format requires parsing and rendering tools to extract requested tag elements such as geometry elements to draw map features or non-geometry elements to answer content-related queries.

Structured data representations enable adding some attributes and additional information (annotations) to the data. Those resulting XML representations of data tend to be significantly larger than binary representations of the same data. The larger document size means that the greater bandwidth is required to transfer the data, as compared to the equivalent binary representations.

In addition, due to the architectural features (integration of autonomous resources), the system spends a lot of time on query/response transformations for relational database-to-GML mappings. WFS enable mediation of autonomous databases and serving the data in common data model through the standard service interfaces and message formats. However, it is often time consuming because of the requirements for query and response conversions (getFeature to SQL and relational tables to GML). In summary, advantages of using structured/annotated data come with its costs.

Geo-Data Characteristics

Geo-data is described with its location on the earth. A location in a 2-dim surface is formulated as (x, y) coordinates. Based on the location attribute, geo-data is unevenly distributed (consider human populations, earthquakes, and temperature distributions) and variably sized. In addition, geo-data collected from sensors are dynamically changed and/or updated over time.

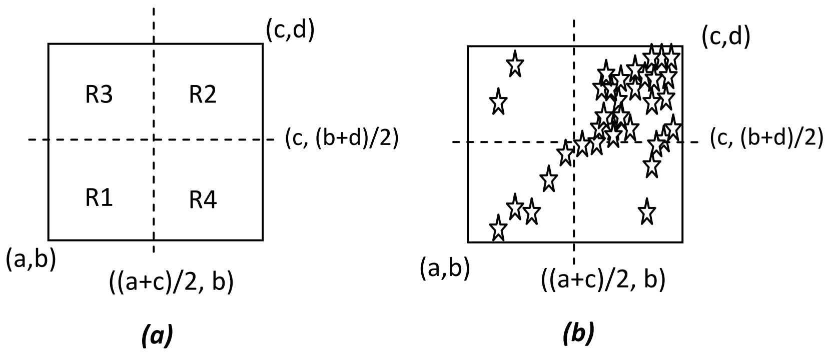

Figure 4: ***Unbalanced load sharing. Server assigned R2:"((a+b)/2, (b+d)/2), (c, d)" gets the most of the work***

Because of these stringent characteristics of data, it is not easy to make load balancing and parallel processing over the unpredictable workload. Figure 4 illustrates this problem. The work is decomposed into independent work pieces, and the work pieces are of highly variable-sized. In the following two sections, we present our solution approaches.

Extending Open GIS Standards with Streaming Data Transfer Capability

The OGC's initial standard WMS and WFS specifications were based on HTTP Get/Post methods, but this type of services have several limitations such as the amount of data that can be transported, the rate of the data transportation, and the difficulty of orchestrating multiple services for more complex tasks. Web Services help us overcome some of these problems by providing standard interfaces to the tools and applications we develop (Aydin, 2007).

Our experience shows that although we can easily integrate several GIS services into complex tasks by using Web Services, providing high-rate transportation capabilities for large amounts of data remains a problem because the pure Web Services implementations rely on SOAP (Gudgin et al., 2007) messages exchanged over HTTP. This conclusion has led us to an investigation of topic-based publish-subscribe messaging systems for exchanging SOAP messages and data payload between Web Services. We have used NaradaBrokering (Pallickara & Fox, 2003), which provides several useful features such as streaming data transport, reliable delivery, ability to choose alternate transport protocols, security and recovery from network failures. This allows us to provide higher level qualities of service in GIS services.

NaradaBrokering is a message oriented middleware (Tran, Greenfield, & Gorton, 2002) system that facilitates communications between entities through the exchange of messages. This also allows us to receive individual results and publish them to the messaging substrate instead of waiting for the whole result set to be returned. In case of using streaming, the standard Web Service interfaces are used for handshaking, and the actual data transfer is done between subscriber and publisher deployed in proposed GIS Web Service components respectively. Besides giving better performance in general, the streaming data transfer technique enables data rendering and processing even on partially returned data. It can even be applied to the real-time data rendering.

Federator-oriented Design Approaches

The system supports fully distributed, decentralized, just-in-time, on-demand data fetching and rendering in which data sources are heterogeneous and autonomous. Autonomy in this context means keeping the data in their originating sources at all times. The originating sources are autonomous and control their own data definitions. Autonomy indirectly results in scalability and enables decentralized data maintenance. On the other hand, it has performance and reliability drawbacks coming from accessing and querying the heterogeneous data sources through WFS-based mediations. Mediators perform time-consuming query and response conversions to provide GML data in standard service interfaces.

In the following sections we present techniques to reduce the negative effects of time-consuming query and data response conversions and data transfer latencies. We focus on the issues at the upper level of data handling, which is view-level data handling at the federator.

The main idea behind the performance enhancement design is enabling stateful service capabilities by developing client-based caching, and parallel data fetching and processing for un-cached data queries. Since the data are kept only in their originating sources and not stored in intermediary places, this architecture provides consistency and strong autonomy.

Adaptive Client-based Caching

OGC Open Standard's GIS services are inherently stateless and based on on-demand processing. They don't maintain the state information between message calls. Introducing a federator service over the OGC's WMS and WFS data services enable stateful service capabilities. The federator's stateful service capabilities also have inspired us to develop novel caching, load balancing and parallel processing approaches in distributed data access/query/display from a single access point. This is presented in the following section.

Client-based caching keeps records about the previously requested layer-data and corresponding query and data attributes, and stores them as session-class objects. The client's cache is kept up-to-date as in working window concept in operating systems. The server differentiates the clients based on their IDs defined in the request. Adaptive client-based caching helps make efficient load balancing over the unpredictable workload by utilizing the locality (Denning & Schwartz, 1972) and nearest neighborhood (Dasarathy, 1991) principles. By the "locality principle," we mean that if a region has a high volume of data, then the regions in close neighborhood are also expected to have high volume of data. The human population data across the earth can be given as an example: Obviously urban areas have higher human population than the rural areas. Differentiating dense data regions from sparse regions enables us to find the most efficient number of partitions for parallel processing and reduces the overhead timings for handling an unnecessary number of partitions. Clustering techniques (Buyya, 1999; Pfister, 1998) provides a more precise way for determining this if one has access to data, but in our architecture we must treat the data servers as black boxes.

Processing from the cache gives better performance results over going to the remote data resources or even to the local disk. However, for large scale applications it might be impossible to cache whole data at intermediary servers because of the physical storage limitations. Client-based caching addresses this issue by proposing a way to get high performance with the limited storage capacities.

In Figure 3, cached data is shown as browser, which the user interacts with through the interactive map tools. The query-regions not overlapping with cached data (boxes numbered as 1, 3 and 4 in the figure) are processed from remote resources through novel distributed load-balancing and parallel processing techniques mentioned in the following section. On the other hand, the queries overlapping with cached data (such as box-2 in the figure and some parts of boxes numbered 1, 3 and 4) are processed from the federator's cache.

Load-balancing and Parallel Processing Through Query Decomposition

A federator inherently makes workload sharing by fetching the different data layers from separate resources to create multi-layered map image. We call this as vertical load balancing. This is a natural load balancing and parallel processing result from the architectural features.

In addition to the layer-based natural load-balancing, a layer (in the multi-layered map image) itself can be split into smaller bounding box tiles and each tile can be farmed out to a worker WFSs/WMSs. Layer-based partitioning is based on attribute-based query decomposition in which the attribute is the bounding box defining the requested data's range in a rectangular shape. This section focuses on individual layer partitioning and gives the architectural details in Figure 5.

In our federator framework, the load balancing and parallel processing techniques are applied over the un-cached regions of the main query. Queried regions overlapping with cache is met from the users' cached data.

The main idea is to decompose an un-cached region's bounding box (defined in the main query) and to create small sub-regions (defined again as smaller, constituent bounding boxes) (see Figure 5). After having partitions in small bounding boxes, each partition is assigned to a separate thread of work, and the results to partitions are merged to create a final response for the main query. The partitions are assigned to threads in a round-robin fashion.

Figure 5 shows a sample case in which there are 3 WFS worker nodes and 5 partitions. BBtotal is the main query's bounding box. Bb1, bb2, bb3, bb4 and bb5 are the partitions obtained from partitioning BBtotal.

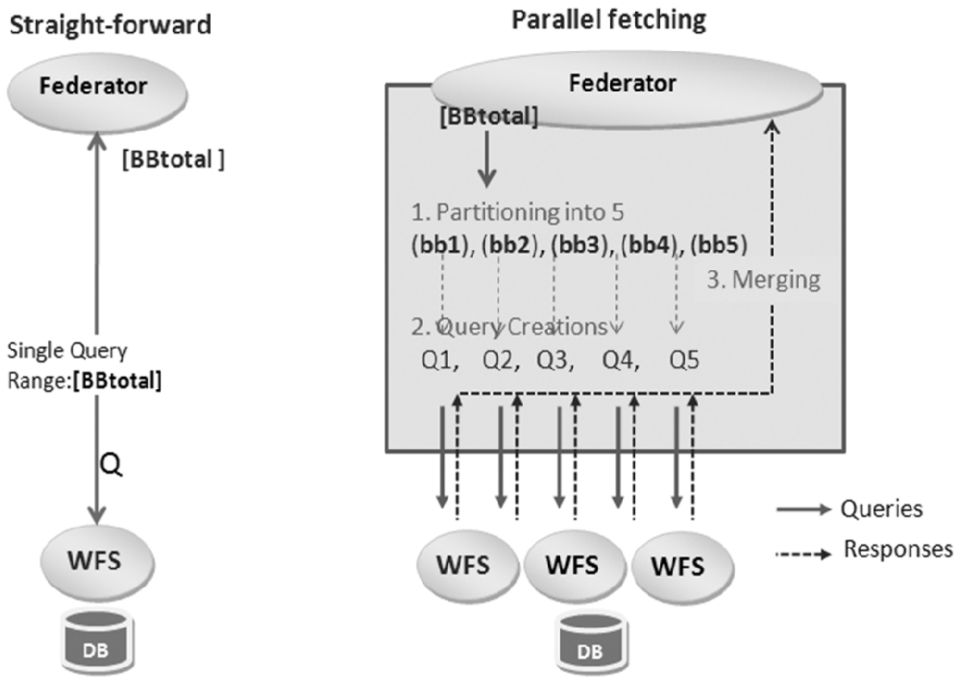

Figure 5: Architectural comparisons of parallel fetching (5-partition) with straightforward single thread fetching. BBtotal=bb1+bb2+bb3+bb4+bb5.

Overall Evaluation of the System Performance

The overall performance results change considerably depending on the cached data available and queried region, and their positioning to each other. Here we analyze those affects on overall performance results. Figure 3 is the test setup. In Figure 3, “browser” shows the user the previous request’s answer (cached data in federator) and boxes 1, 2, 3 and 4 illustrate the user’s successive query region.

Here we analyze end-to-end response times based on all possible cases of query and cached data ranges positioning (see Figure 3). Overall performance changes depending on cached data and main query positioning. Queries and cached data can be positioned in three possible ways. These are:

1. Main query range falls outside of cached data ranges [worst case]
2. Main query range falls in cached data ranges [best case]
3. Query range partially overlaps cached data ranges [in between]

The first group of requests is typically first-time queries. The federator does not have prior state information about them. In such cases, federator fetches whole data falling in the requested ranges from remote databases through WFSs.

The second group of requests results typically from users zooming-in or panning (cutting a rectangular region over the display map) actions over the event-based interactive map tools. In such cases, the query falls in the cached data ranges and gives the best performance result. In this case the federator does not need to make successive queries to fetch the required data from the remote distributed databases through WFS-based mediators. The cache meets the whole requested ranges. Cached data is GML data kept in federator from the previous request. It is used to create the map image shown as "browser" in Figure 3.

The third group of requests comes primarily from moving (drag and drop) and zooming-out actions over the event-based interactive map tools displaying the previous map images. In such cases, cache cannot meet the whole requested range. This is called partial overlapping with cache. These three groups of requests are illustrated in Figure 3.

We now evaluate the performance of the system. The test setup is shown in Figure 3: The system is evaluated with a Geo-science application, Pattern Informatics (PI) (Nanjo, Holliday, Chen, Rundle, & Turcotte, 2006) at its core. Performance results are obtained with two-layer map images for simplicity. The bottom layer is from NASA's satellite map images provided by OnEarth project's WMS ("OnEarth," 2007), and the top layer is earthquake seismic data provided by WFS for Geo-science application.

We run the test in Local Area Network (LAN). We have used machines having 2 Quad-core Intel Xeon processors running at 2.33 GHz with 8 GB of memory. For the partitioning process we have used 6 WFS workers to which partitions are assigned in a round-robin fashion.

Table 1: Average response times and standard deviations for response times in different possible cases. The values are in seconds and the test setup was run 50 times to get each value in the table.

Data MB	(No-Cache) (No-PRT) Single process	StdDev	(No-cache) (PRT) multi-process partition to 10 (1)	StdDev	(Cache) (No-PRT) Cache meets whole query (2)	StdDev
0.01	1.81	0.14	2.33	0.13	1.04	0.23
0.1	2.64	0.31	2.76	0.10	1.15	0.23
0.5	5.00	0.24	3.46	0.12	1.37	0.44
1	8.23	0.20	4.64	0.11	1.69	0.42

The systematic uncertainty for our timer is in 10's of milliseconds.

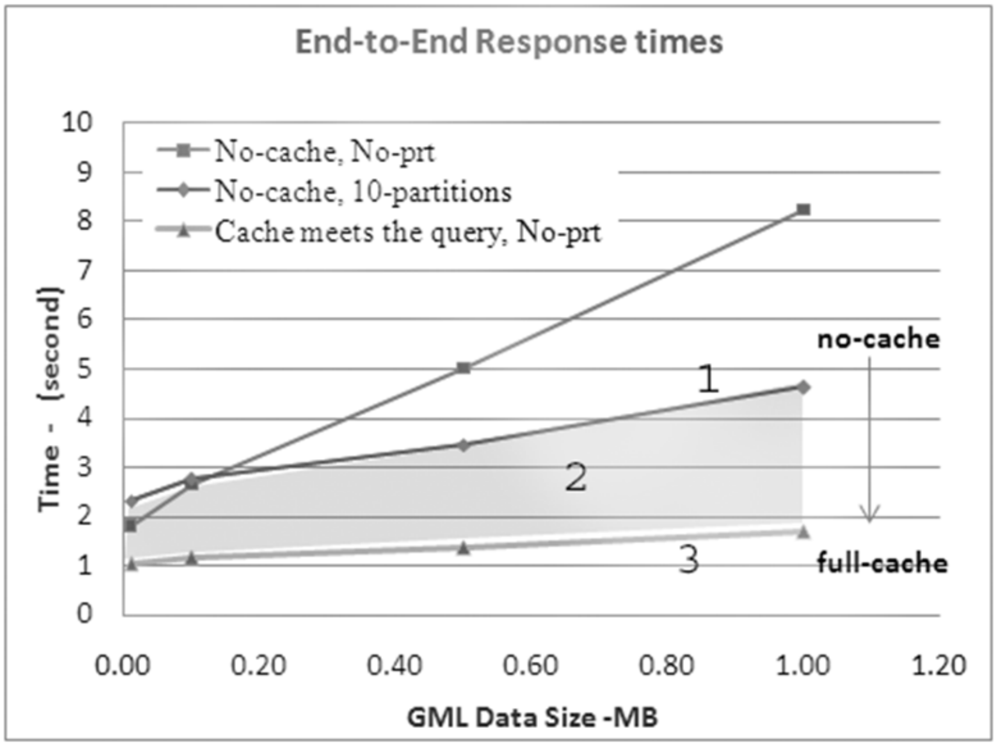

Figure 6: End-to-end response times according to possible query cases. Figure 3 shows the test setup.

Figure 6 shows that stateful access to OGC's stateless data services through the federator and resulting parallel processing through query decomposition and caching techniques enhanced the system responsiveness to a great extent. The performance results are end-to-end response times in which one end is the database and the other end is the user.

Depending on how many partition of the main query is met from the cache, the performance changes in region-2 between the lines tagged as no-cache and full-cache. As the overlapped partition increases, the performance gain increases and gets close to the full-cache line. The full-cache line shows the performance results in case of that the queries are served fully from cache. The performance results also show that as the data size increases, the performance gain from the proposed techniques increases.

5. Abstraction of the Framework for the General Domains

Our experiences with GIS have shown that a federated, service-oriented, GIS-style information model can be generalized to many application areas and scientific domains. We call this generalized framework Application Specific Information System (ASIS), and provide a blueprint architecture in terms of principles and requirements. Developing such a framework requires first defining a core language (such as GML) expressing the primitives of the domain; second, key service components, service interfaces and message formats defining services interactions; and third, the capability file requirements (based on core-language) enabling inter-service communications to link the services for the federation (see Figure 7).

GIS is a mature domain in terms of information system studies and experiences. It has standards bodies defining interoperable online service interfaces and data models such as OGC ISO/TC211, but many other fields do not have this. In order to see the applicability of the GIS-style information model given in Figure 3, we have surveyed two science domains (Astronomy and Chemistry). Table 2 presents the results briefly in terms of service counterparts (ASIS vs. science domains).

Astronomy has a standards body, the International Virtual Observatory Alliance (IVOA), for defining data formats and online services that are somewhat analogous to the OGC standards. FITS (Flexible Image Transfer), Images and VOTable (Williams et al., 2002) are the data models. SkyNodes are database servers with an ADQL (Astronomy Distributed Query Language) based SOAP interfaces that return VOTable-encoded results. VOPlot and TopCat are two services to visualize the astronomy data in the format of VOTable, FITS and images. VOResource and UCD are the metadata definition and standards for the service descriptions (Yasuda et al., 2004).

Chemistry, although a vastly different field, does provide a common data model (CML (Holliday, Murray-Rust, & Rzepa, 2006)) that can be used to build up Web Services. Although many research groups have investigated service architectures for chemistry and chemical informatics, the field has (to our knowledge) no Web Service standards-defining body equivalent to the OGC or IVOA.

This chapter presents a high level architecture that consists of abstract components and explains their data flow and components interactions. In this section, we focus on the principles and requirements to generalize GIS-like architecture to any other information system domains. It should be noted that this abstract architecture is intended to be domain-specific. That is, it may be realized in chemistry or astronomy, for example, but we are not suggesting cross-domain interoperability.

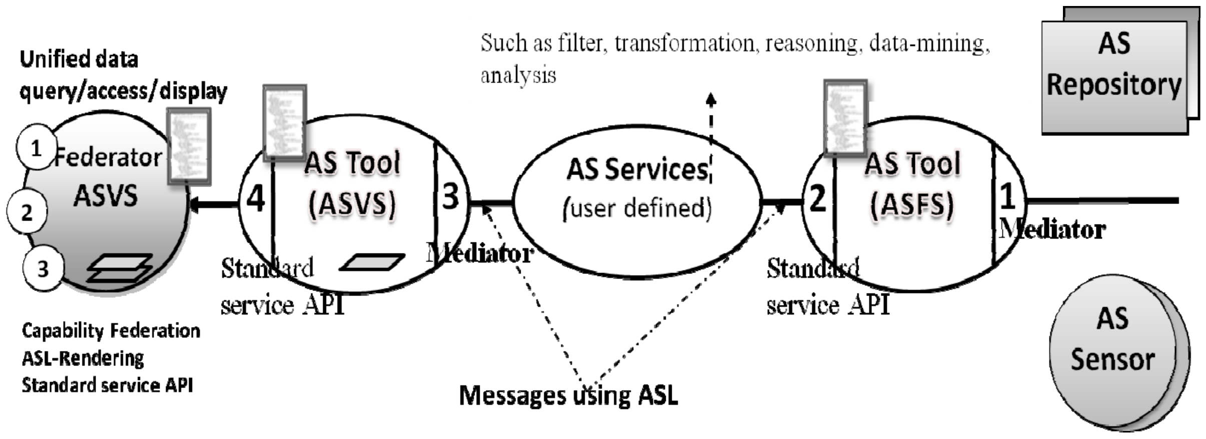

Figure 7: Application Specific Information System (ASIS)

ASIS is a proposed solution to heterogeneous data integration. This solution enables inter-service communication through well-defined service interfaces, message formats and capabilities metadata. Data and service integration is done through "capability" federation of these services, which are implemented in Web Services. In ASIS approach, there are two general groups of services. These are Application Specific Feature Service (ASFS) and Application Specific Visualization Service (ASVS), and each service is described by corresponding generic metadata descriptions that can be queried through Web Service invocations. In addition to allowing service discovery, this approach also enables at least three important qualities of services. First, services of the same type that provide a subset of the request can be combined into a "super-service" that spans the query space and has the aggregate functionality of its member services. Second, the capability metadata can be used to determine how to combine services into filter chains with interconnected input-output ports. Third (and building on the previous two), capabilities of super-services can be broken into smaller, self-contained capabilities that can be associated with specific services. This enables performance gains through load-balancing.

ASIS must consist of filter-like Web Services components (ASFS and ASVS) having common interfaces and communicating with each other through a capability metadata exchange interface. Being a Web Service enables filter services to publish their interfaces, locate each other and chain together easily. Filters have inter-service capabilities and are chainable. If the filter is capable of communicating and obtaining data from other filters, and updates (or aggregates) its capability metadata with these data (after capability files exchange), then it can claim that it serves these data. Filter Services are information/data services that enable distributed data/information access, querying and transformation through their predictable input/output interfaces defined by capability document. Filter located in the same community network can update their capability metadata dynamically through "getCapabilities" service interface of the filters. Dynamically updating capabilities of filters enable removal of obsolete data or down filters.

Abstract Components and Matching to Sample Science Domains

-Chemistry and Astronomy

In ASIS, there are two groups of filter services, ASVS and ASFS, which correspond to the OGC's WFS and WMS, respectively. Since they have different service APIs and provided data, they have different schema of capabilities. The capability metadata defines service and data attributes, and their constraints and limitations to enable clients to make valid queries and get expected results. Capabilities metadata and Application Specific Language (ASL) are closely related to each other. One defines the domain-specific data and other defines the query and response constraints over the service and data provided.

ASVS must visualize information and provides a way of navigating ASFS and their underlying database. ASVS must provide human readable information such as text and graphs (scalable vector graphic (SVG) or portable network graphic (PNG)) images. An ASFS is an annotation service providing heterogeneous data in common data model with an attribute-based query capability. ASFS serves data in ASL, which must be realized as a domain specific XML-encoded common data model containing content and representation tags. Heterogeneity in queries and data formats is handled through resource specific mediators.

User defined services in ASIS (Figure 7) provide application specific data and services. These can include transformations, reasoning, event-detection, and data-mining tools for extraction knowledge from the feature data provided by ASFS in ASL format. For example, we can provide the Pattern Informatics application as an example once again. In Pattern Informatics, ASIS needs to overlay multi-layered map images with earthquake forecast values (Rundle, Turcotte, Shcherbakov, Klein, & Sammis, 2003) as hot-spot plots in colored boxes showing magnitudes of expected earthquake seismicity.

Table 2: Components and common data model matching for generalization of GIS to ASIS. Two selected domains are Astronomy and Chemistry.

ASIS / Science Domains	Common data Model ASL	Components ASFS	ASVS	Metadata
GIS	GML	WFS	WMS	capability.xml schema
Astronomy	VOTable, FITS	SkyNode	VOPlot TopCat	VOResource
Chemistry	CML, PubChem	None	NO standard JChemPaint, JMOL	None

Inter-service Communications

Inter-service communication is achieved through common service interfaces and capability metadata exchange. The standard service interfaces can be grouped into three types: a) capability metadata exchange: inter-service communication (set-up stage); b) interactive data display: selecting layer composition and bounding box regions; and c) querying of data itself over the display, getting further information about the data content and attributes.

As mentioned before, capability helps clients make valid requests for its successive queries. Capability basically provides information about the data sets and operations available on them with communication protocols, return types, attribute based constraints, etc. Each domain has different set of attributes for the data and it is defined in ASL common data model. For example, in GIS domain, attributes might be bounding box values (defining a range query for data sets falling in a rectangular region) and coordinate reference system.

Standard requests/query instances for the standard service interfaces are created according to the standard agreed-on request schemas. These are defined by open standards bodies in corresponding domains. The request instances contain format and attribute constraints related to the ASL common data model. For example in the GIS domain, *getMap* request defines a map images' return format (JPEG, PNG, SVG, etc.), height, width, bounding box values, and so on. Format, height and width are related to display, but bounding box values are related to the attributes of the data defined in its ASL representation provided by ASFS. In this specific example of the *getMap* request, ASVS must both visualize information through the *getMap* service interface and provide a way of navigating ASFS services and their underlying database. ASVS make successive queries to the related ASVSs to get the ASL data and render it to create final display for its clients.

In ASIS, the task of mediators is to translate requests to the standard service interfaces to those of the information/data sources', and transform the results provided by the information source back to the ASIS's standard formats. For ASFS, the returned data is ASL, and for ASVS the returned results can be any kind of display format such as images.

Acting as a proxy of information source, the mediators communicate with an information source in its native language and API. They communicate with ASIS in a commonly agreed language (ASL) and Web Service API calls (such as *getCapabilities*, *GetFeature* and *DescribeFeatureType* in GIS). Because of the obvious heterogeneity between different science domains, each will need to extend and create its own service interfaces corresponding WMS and WFS, as well as XML-based queries for those services. Using common data model-ASL and common services make autonomous resource to be integrated into the system in a manageable way.

The mediators-wrappers enable data sources integrated to the system conform to the global data model (ASL) but enable the data sources to maintain their internal structure. At the end, this whole mediator system provides a large degree of autonomy. Instead of actual physical data federation, system makes distributed querying and response composition on the fly.

6. Conclusions

We have presented a service-oriented architecture for understanding and managing the production of knowledge from the distributed observation, simulation and analysis data through integrated data-views in the form of multi-layered map images. The infrastructure is based on a common data model, standard GIS Web-Service components, and a federation service. The federator integrates GIS data service components and enables unified data access and query over integrated data-views through event-based interactive display tools. Integrated data-views are defined in the federator's capability metadata, which consists of composition of layers provided by standard GIS Web-Services. The framework applies just-in-time (late-binding) federation in which the data is kept in its originating sources all the time. This enables autonomy and easy data maintenance.

Creating a GIS in accordance with OGC and Web Services standards, and the compatibility nature of open standard GIS services and their capability definitions, inspired us to develop an information system enabling both a unified data access/query and a display from a single access point. Open standards and Web Service technologies also enable integrating the third party geospatial functionality and data into the custom applications easily.

We have developed a framework for federated service-oriented Geographic Information Systems and addressed interoperability issues by integrating Web Services with open geographic standards. This enables us to provide interoperability at data, service and application levels. We have enhanced the standard GIS Web Service components with the streaming data-transfer capability by using a publish/subscribe-based messaging middleware. We have investigated performance efficient designs for the federator, data transfer, and distributed rendering to support responsiveness in GIS requiring interactive response times.

Open standard GIS services are inherently stateless and based on on-demand processing. They do not maintain state information between message calls, and that causes poor performance. Our federator architecture enables stateful access to stateless GIS data services from a single access point through client-based caching technique.

The federator architecture inherently enables workload sharing by fetching the different data layers from separate resources to create a multi-layered map image. This is a natural load balancing and parallel processing resulting from the architectural features. However, we can take this general idea further. In addition to layer-based natural load-balancing, a layer (in the multi-layered map image) itself can be split into smaller bounding box tiles and each tile can be farmed out to a worker WFSs/WMSs. Layer-based partitioning is based on attribute-based query decomposition.

Although the framework is fine-grained for GIS, we have also defined the principles for generalizing federated service-oriented GIS to the general science domains. We have defined two general service types (ASFS and ASVS) with limited number of service interfaces. Service interfaces enable metadata exchange and data querying. Data flows from databases to users through ASFS and then ASVS. Due to the domain specific data heterogeneity, each domain should define its own ASL and corresponding queries.

REFERENCES

Atkinson, M., DeRoure, D., Dunlop, A., Fox, G., Henderson, P., Hey, T., et al. (2005). Web Service Grids: An Evolutionary Approach *Concurrency & Computation: Practice&Experience, 17*(Number 2-4, February/April 2005), 377-389.

Aydin, G. (2007). *Service Oriented Architecture for Geographic Information Systems Supporting Real Time Data Grid.* Unpublished Doctoral dissertation, Indiana University, Bloomington.

Aydin, G., Sayar, A., Gadgil, H., Aktas, M. S., Fox, G. C., Ko, S., et al. (2008). Building and Applying Geographical Information Systems Grids. *Concurrency and Computation: Practice and Experience (To appear)*.

Beaujardiere, J. d. l. (2004). *OGC Web Map Service Interface* (Report No. 03-109r1): Open GIS Consortium Inc. (OGC)

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., et al. (2004). Web Services Architecture [Electronic Version], from http://www.w3.org/TR/ws-arch/

Buyya, R. (1999). *High Performance Cluster Computing: Architectures and Systems* (Vol. 1). NJ, USA: Prentice Hall PTR.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). *Web Services Description Language (WSDL)* (No. NOTE-wsdl-20010315): World Wide Web Consortium (W3C)

Cox, S., Daisey, P., Lake, R., Portele, C., & Whiteside, A. (2003). *OpenGIS® Geography Markup Language (GML) Encoding Specification* (No. 02-023r4): Open Geospatial Consortium (OGC)

Dasarathy, B. V. (1991). *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*: IEEE Computer Society Press

Denning, P. J., & Schwartz, S. C. (1972). Properties of the working-set model. *Communications of the ACM, 15*(3), 130.

Evans, J. D. (2003). *Web Coverage Service (WCS), Version 1.0.0* (OpenGIS® Standard Specification No. 03-065r6)

Fielding, R. T., & Taylor, R. N. (2002). Principled design of the modern web architecture. *ACM Transactions on Internet Technology, 2*(2), 115-150.

Foster, I., & Kesselman, C. (2004). *The Grid 2: Blueprint for a new Computing Infrastructure*. San Francisco, USA: Elsevier

Fox, G. C. (2004). Grids of Grids of Simple Services. *Computing in Science and Engineering, 6*(4), 84-87.

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.-J., Nielsen, H. F., Karmarkar, A., et al. (2007). *SOAP Version 1.2 Part 1: Messaging Framework* (Standard Specification)

Holliday, G. L., Murray-Rust, P., & Rzepa, H. S. (2006). Chemical markup, XML, and the world wide web. 6. CMLReact, an XML vocabulary for chemical reactions. *Journal of chemical information and modeling, 46*, 145-157.

JAG. (1999). *Joint Advisory Group* Retrieved 03/27/2008, from http://www.isotc211.org/organizn.htm#jag

Khare, B. R., & Taylor, R. N. (2004, May 2004). *Extending the Representational State Transfer (REST) Architectural Style for Decentralized Systems.* Paper presented at the 26th International Conference on Software Engineering (ICSE'04), Edinburgh, Scotland.

Kirtland, M. (2001). *A Platform for Web Services* (Tech Report): Microsoft

Kolodziej, K. (2004). *OpenGIS Web Map Server Cookbook* (Implementation Specification No. 03-050r1): Open Geospatial Consortium Inc. (OGC)

Koontz, L. D. (2003). *Geographic Information Systems: Challenges to Effective Data Sharing* (No. GAO-03-874T). Washington, DC

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

Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., et al. (2006). Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice and Experience, 18*(10), 1039-1065.

Nanjo, K. Z., Holliday, J. R., Chen, C.-c., Rundle, J. B., & Turcotte, D. L. (2006). Application of a modified pattern informatics method to forecasting the locations of future large earthquakes in the central Japan. *Tectonophysics, 424*, 351-366.

OGC. (1994, 06/12/2008). *The Open Geospatial Consortium, Inc* Retrieved 02/14/2008, from http://www.opengeospatial.org/

OnEarth. (2007, 12/08/2007). Retrieved 03/15/2008, from http://onearth.jpl.nasa.gov

Pallickara, S., & Fox, G. (2003). *NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids*. Paper presented at the ACM/IFIP/USENIX. from http://grids.ucs.indiana.edu/ptliupages/publications/NB-Framework.pdf

Peng, Z.-R., & Tsou, M.-H. (2003). *Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks*. New Jersey, USA: John Wiley & Sons.

Pfister, G. F. (1998). *In Search of Clusters*. Upper Saddle River, NJ, USA Prentice-Hall, Inc.

Redmond, F. E. (1997). *Dcom: Microsoft Distributed Component Object Model with Cdrom* (1st edition ed.). Foster City, USA: IDG Books Worldwide, Inc.

RMI [Electronic. (2004). Version]. *Java Remote Method Invocation Specification*. Retrieved June 2008, from http://java.sun.com/j2se/1.4.2/docs/guide/rmi/

Rundle, J. B., Turcotte, D. L., Shcherbakov, R., Klein, W., & Sammis, C. (2003). Statistical physics approach to understanding the multiscale dynamics of earthquake fault systems. *Geophysics, 41*(4).

Sayar, A., Pierce, M., & Fox, G. (2005). *OGC Compatible Geographical Information Services* (Tecchnical Report No. TR610). Bloomington: Indiana University

Sayar, A., Pierce, M., & Fox, G. (2006). *Integrating AJAX Approach into GIS Visualization Web Services*. Paper presented at the IEEE, International Conference on Internet and Web Applications and Services, ICIW'06.

Sonnet, J. (2005). *Web Map Context Documents (WMC)* (Standard specs No. 05-005): Open Geospatial Consortium Inc. (OGC)

Tiampo, K. F., Rundle, J. B., Mcginnis, S. A., & Klein, W. (2002). Pattern Dynamics and Forecast Methods in Seismically Active Regions *Pure and Applied Geophysics, 159*(10), 2429-2467.

Tran, P., Greenfield, P., & Gorton, I. (2002). *Behavior and performance of message-oriented middleware systems*. Paper presented at the International Conference on Distributed Computing Systems Workshops, ICDCSW.

Turi, D., Missier, P., Goble, C., Roure, D. D., & Oinn, T. (2007). *Taverna Workflows: Syntax and Semantics* Paper presented at the 3rd IEEE International Conference on e-Science and Grid Computing (e-Science'07), Bangalore, India.

Vretanos, P. A. (2002). *Web Feature Service Implementation Specification* (Reference Document No. 02-058)

Williams, R., Ochsenbein, F., Davenhall, C., Durand, D., Fernique, P., Giaretta, D., et al. (2002). *VOTable: A Proposed XML Format for Astronomical Tables* (Standard Specification): US National Virtual Observatory

WS-I. (2002). *Web Service Interoperability* Retrieved 03/23/2008, 2008, from http://www.ws-i.org/

Yasuda, N., Mizumoto, Y., Ohishi, M., O'Mullane, W., Budav´ari, T. a., Haridas, V., et al. (2004). *Astronomical Data Query Language: Simple Query Protocol for the Virtual Observatory*. Paper presented at the Astronomical Data Analysis Software and Systems XIII. ASP Conference Series, ASP Conf. Series. from http://www.adass.org/adass/proceedings/adass03/reprints/P3-10.pdf