Middle-East Journal of Scientific Research 3 (1): 36-48, 2008
ISSN 1990-9233

Implementation of a Geospatial Web service Using Web Services Technologies and Native XML Databases

Pouria Amirian and Ali A. Alesheikh

Faculty of Geodesy and Geomatics Engineering, K.N. Toosi University of Technology,
Vali-e-Asr St., Post Code: 1996715433, Mirdamad Cross, Tehran, Iran

Abstract: Recent developments in Web service technologies as collection of XML-based protocols and technologies have shown promise for automatic discovery, access and use of data and processing resources over internet in an interoperable manner. Like Web services in Information Technology world, in GIS community, Open Geospatial Consortium (OGC) Web service framework as collection of geospatial Web services and encodings have been recently introduced to overcome spatial non-interoperability problem associated with most geospatial processing systems. With this in mind, this paper suggests that making use of Web services technologies as enabling infrastructure for implementing geospatial Web services can significantly facilitate sharing geospatial data as well as access to processing services from multiple resources in and out of GIS community. More accurately, geospatial Web services which are developed using Web services technologies can provide access interoperability among various geospatial and non-geospatial processing systems. In addition to access interoperability, making use GML (Geography Markup Language) as an open and widely used data standard, data interoperability can be achieved. Meanwhile, proper management of geospatial data necessitates use of efficient and optimized data management systems. In this respect, based on practical performance test, the paper also describes that using native XML database systems, management and publishing geospatial data (in feature level) can be facilitated and improved significantly.

Key words: Spatial interoperability . geospatial web service . web services technologies . native XML database

INTRODUCTION

Based on OGC Reference Model [1], spatial interoperability refers to capability to communicate, execute programs, or transfer spatial data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units. As the mentioned definition suggests non-interoperability of geospatial processing systems hamper share of geospatial data and services among software applications. From technical point of view two kinds of non-interoperability can be identified in geospatial processing systems: Data and Access non-Interoperability [2].

Data non-interoperability implies different vendors' geospatial processing systems use internal data formats and produce data in formats that are different and in most cases proprietary. In order to share geospatial data, use of proprietary data formats obliges use of data converters and/or exchange formats. But using data converters and/or exchange formats involves resource and time consuming conversion process. In addition, there are so many different standards for geospatial data that converting various data formats can itself become a barrier to interoperability [3].

Access non-Interoperability means different vendors' geospatial processing systems use proprietary software access methods with proprietary software interfaces which restrict inter-process communication among various geospatial processing systems. In other words, interface definition languages, communication protocols, communication ports and even object transfer mechanisms, vary in each software development platform, so the software platform which has been used to develop the geospatial processing system imposes the use of specific and proprietary communication methods among various parts of the system. For this reason, different geospatial processing systems which have been developed by different software development platforms can't communicate and share services automatically and in an interoperable manner.

Corresponding Author: Dr. Pouria Amirian, Faculty of Geodesy and Geomatics Engineering, K.N. Toosi University of Technology, Vali-e-Asr St., Post Code: 1996715433, Mirdamad Cross, Tehran, Iran

In GIS community specific kinds of online services, which are called geospatial Web services, have been recently introduced to overcome spatial non-interoperability problem. Geospatial Web services have been developed with the goal of sharing geospatial data and services among heterogeneous geospatial processing systems. Web Feature Service (WFS), Web Map Service (WMS) and Web Coverage Service (WCS) are the most fundamental geospatial Web services which are introduced by Open Geospatial Consortium (OGC). At the same time, in IT world, the best solution for providing interoperability among heterogeneous systems in distributed and decentralized environments are Web services technologies [4].

Geospatial Web services and Web services technologies are different with each other. Web services are composed of particular set of technologies and protocols but Geospatial Web services are comprised of defined set of interface implementation specifications which can be implemented with diverse technologies.

With respect to above description, it is suggested that making use of Web services technologies as enabling infrastructure for implementing geospatial Web services would significantly facilitate sharing geospatial data as well as access to processing services from multiple resources in and out of GIS community. In other words, geospatial Web services which are developed using Web services technologies can provide access interoperability among various geospatial and non-geospatial processing systems. Furthermore, using open and platform independent data standards like Geography Markup Language (GML), data interoperability can also be achieved. Meanwhile, proper management of geospatial data (such as multiple user access and versioning mechanisms), necessitate use of efficient and optimized data management systems. In this context, considering the nature of GML (as an XML-based language), using native XML database systems is suggested for facilitating and improving geospatial data management. This paper describes development of a Geospatial Web service using Web Services Technologies and a Native XML database system to achieve spatial interoperability, while having a proper management on spatial data over the web. Based on practical tests of this research, developed system proved to be an efficient solution for developing geospatial Web services. The paper first depicts Web services technologies, Geospatial Web services and XML database systems and then explains and discusses the developed system.

WEB SERVICES TECHNOLOGIES

The Web as it exists today is intended for human consumption. Consequently, data is presented in a form that is human-readable, but this form of representation is error prone and difficult for applications to examine, extract and use both, automatically and programmatically. So there is a need for application to application communication and this is the idea of application-centric Web rather than human-centric Web (The Web as it works today).

The core idea of the application centric Web is to provide software applications the ability of cross platform communication without the intervention of human beings. In other words, automatic and direct communications among functional units which are running on heterogeneous platforms are the unique characteristics of the Application centric Web (as the next generation of Web). The promising technologies for this kind of communications are Web services technologies [5].

Web services are self-contained, self-describing, modular applications that can be published, located and invoked across the Web and perform functions that can be anything from simple requests to complicated business processes [7]. Web Services are the basic components of distributed service-oriented systems. The World Wide Web Consortium (W3C) defines Web Services as a software system designed to support machine-to-machine interaction over the Internet [7-9].

Any Web service has an interface described in a machine-processable format. Other systems and services interact with the Web service in a manner described by its description using messages. Messages are conveyed typically using Hyper Text Transfer Protocol (HTTP) with an XML serialization, in conjunction with other Web-related standards, but any other communication protocol can be used as well [7].

Web services are implemented by using a collection of standards and technologies. These standards and technologies when considered together, establish what is widely referred to as the Web services protocol stack. Figure 1 illustrates the eight distinct layers of the Web services protocol stack [10].

These eight layers are grouped into three distinct levels (Fig. 1); each level indicates a level of maturity for the layers it contains.

The enabling standards level contains two layers: the network transport protocols and meta-language. The layers within the enabling standards level contain well-defined and accepted standards and protocols that are widely used in internet and Web such as HTTP and XML.

The evolving standards level contains layers for Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI). Collectively, these layers form the core technologies for implementing Web services.

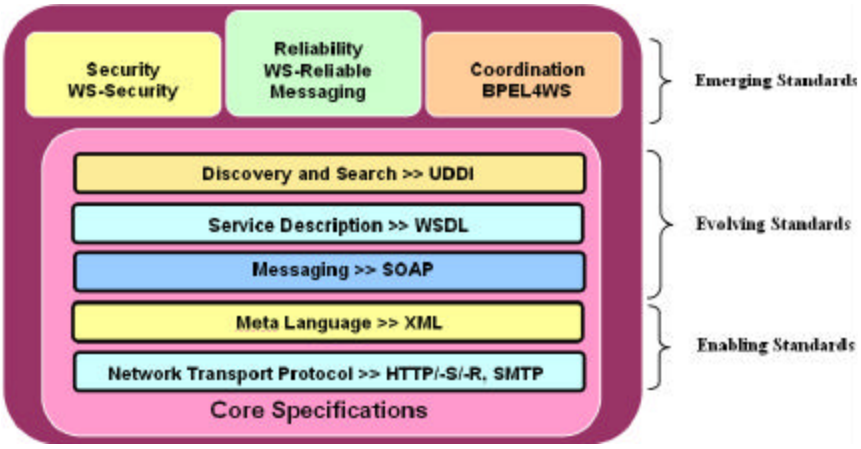

Fig. 1: Web services protocol stack

SOAP is a lightweight, XML-based protocol for exchanging information in decentralized, distributed environments. SOAP is used for messaging among Web service provider and Web service requesters. SOAP is platform independent and also it can be used with virtually any Network Transport protocols such as File Transfer Protocol (FTP), HTTP, Secure HTTP (HTTP-S) and Reliable HTTP (HTTP-R).

WSDL is XML-based specification for describing the capabilities of a service in a standard and extensible manner. Technically WSDL defines the software interface of Web service in platform independent approach.

UDDI is a set of specifications and Application Programming Interfaces (APIs) for registering, finding and discovering Web services.

The last level of standards or emerging standards level represents proposed standards that are promoted by individual vendors (such as Microsoft, IBM and Sun Microsystems). This level consists of specifications which have not yet gained broader endorsement or acceptance in the wider Web services community and have not been adopted as open standards for development by key standards bodies such as the World Wide Web Consortium (W3C) and the Organization for the Advancement of Structured Information Standards (OASIS).

Web services are based on open standards, so they provide interoperability in decentralized and distributed environments like Web. These new technologies can be developed by using any software platform, operating system, programming language and object model.

GEOSPATIAL WEB SERVICES

Nowadays, geospatial Web services have been considered as the promising technology to overcome the non-interoperability problem associated with current geospatial processing systems. They are particular kind of online services which deal with geospatial information and can provide access to geospatial information stored in a database, perform simple and complex geospatial analysis and return messages that contain geospatial information [11].

In this context, OGC has defined a comprehensive framework of geospatial Web services which is known as OGC Web services framework (OWS). OWS allows distributed spatial processing systems to interact with the HTTP technique and provides a framework of interoperability for the many web-based services, such as accessing spatial data services, spatial processing services and data locating services [12]. OWS framework consists of interface implementation specification and encodings which are openly available to be implemented by developers. The interface implementation specifications are software technology neutral details about various operations of each geospatial Web service [3]. The encodings provide the standard glue among different parts of geospatial Web services. Each service of this framework can be implemented using various software technologies and systems. Among all Web services which are defined in the OWS, Web Feature Service (WFS) plays a major role. WFS is the only OGC Web service which provides feature level access to geospatial data. This means using WFS, any geometry and non-geometry property of geospatial features can be retrieved. When a client sends a request to an OGC WFS instance, the service sends a response message that provides geospatial feature data in GML. In this case, requests for geospatial data contain Filter Encoding (FE) expressions. Using FE, spatial and non-spatial query expressions can be created to be sent to WFS. GML and FE are two main encodings of the OWS which are

heavily used in requesting and retrieving geospatial data when WFS is used. WFS and other geospatial Web services supply standard access to geospatial data thus provide access interoperability in GIS community. Next sections briefly introduce GML, WFS and FE.

Geography Markup Language (GML): GML is an XML-based markup language that is used to encode information about real world objects. In GML these real world objects are called features and they have geometry and non-geometry properties.

GML has three main roles with respect to geospatial information. First as an encoding for the transport of geospatial information from one system to another; second as a modeling language for describing geospatial information types; and third as storage format for geospatial information [13].

Typically in any management related tasks (like environmental management, natural resource and so on) one needs to examine and explore data from several sources, use simulation models, develop scenarios, assess impacts and provide support for decision makers [14]. In this case, use of XML-based languages for data exchange is an improvement on non XML data formats because the XML format is partially self-documenting and provides common methods for parsing files, obtaining their structure and transforming them to alternative formats [15]. GML (as an XML-based language) is well suited for encoding the geospatial information sent to and from geospatial Web services. GML is used in both the request and response messages of the WFS, which is a standard service for accessing geospatial feature data.

As a modeling language, GML provides a rich variety of objects for describing geospatial information, including geospatial features, coordinate reference systems, topology, time, units of measure and generalized values [16]. In addition, using GML spatial and non-spatial relationships among real world objects can be modeled efficiently.

As storage format GML is a plain textual file format which can be managed using any database management system. Since GML is based on XML, the same technology for managing XML data can be used to manage geospatial data stored in GML. In general XML databases are used to manage XML data. XML databases are described later in this paper.

GML has been a turning point in geomatics to the extent that many national and private organizations have already adopted this format as their main geospatial storage and exchange format [17].

Figure 2, illustrates a simple GML document fragment which consist of two features. City feature has three properties; Name, Position and IsCapitalOf. Name of the city is declared using Name element and Position of the city is expressed using gml:Point element which is defined in GML standard. The gml:Point has a srsName attribute for denoting Spatial Reference System (SRS) in which the coordinates are represented. As the name implies IsCapitalOf property states relationship between city and country features. In this case Tehran is the capital of country feature which has "T1" as its gml:id attribute. At the other hand, country feature has four non-geometry properties which state its name, continent, region and its capital city. The Capital property of country feature is used to indicate its capital

```
<City gml:id="C30">
        <Name>Tehran</Name>
        <gml:position>
                <gml:Point srsName="WGS84">
                        <gml:coordinates>3950000,530000</gml:coordinates>
                </gml:Point>
        </gml:position>
        <IsCapitalOf xlink:href="#T1" />
</City>
<Country gml:id="T1">
        <Name>IRAN</Name>
        <Continent>Asia</Continent>
        <Region>Southern Asia</Region>
        <Capital xlink:href="#C30"/>
</Country>
```

Fig. 2: A simple GML document fragment which describes some properties of Tehran as City feature and Iran as Country feature. Position of City is indicated using gml:Point element which is defined in GML standard. Association between Tehran and Iran is expressed using xlink:href attribute

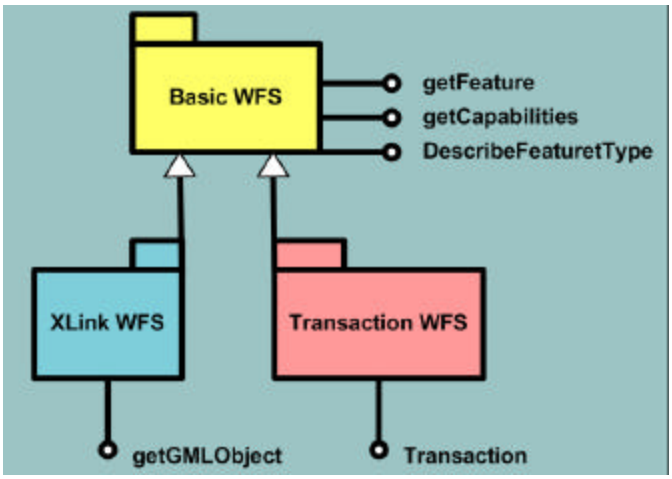

Fig. 3: Three classes of WFS and their operations

```
<wfs:Query typeName="City">
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName> Name</ogc:PropertyName>
      <ogc:Literal>Tehran</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
</wfs:Query>
```

Fig. 4: Filter encoding for retrieving city feature whose name is Tehran

city. In both features Xlink:href attribute is used to express association between country and city features. Xlink:href is defined in XLink standard. XLink is a W3C standard that specifies the syntax and behavior for hyperlink traversal in a set of XML documents [11]. These links are used in GML to express associations between geospatial features.

Web Feature Service (WFS) and Filter Encoding (FE): Web Feature Service is the main geospatial Web service for publishing and requesting vector geospatial data in GML format. When a client sends a request to an OGC WFS, the service sends a response message that provides geospatial feature data in GML. As illustrated in Fig. 3, three classes of Web Feature Services are defined in the WFS implementation specification: Basic WFS, XLink WFS and Transaction WFS [18].

A Basic WFS service implements three operations: GetCapabilities, DescribeFeatureType and GetFeature.

A client can request an XML-encoded capabilities document (containing the names of feature types that can be accessed via WFS service, the spatial reference system(s), the spatial extent of the data and information about the operations that are supported) by sending the GetCapabilities request to the WFS instance.

The purpose of the DescribeFeatureType operation is to retrieve an XML Schema document with a description of the data structure (or schema) of the feature types served by that WFS instance.

The GetFeature operation allows for the retrieval of feature instances (with all or part of their properties) as GML document.

An XLink WFS supports all the operations of a basic web feature service and in addition it would implement the GetGmlObject operation for local and/or remote XLinks. The GetGmlObject operation allows retrieval of features and elements by gml:id from a web feature service [18].

A Transaction web feature service supports all the operations of a basic web feature service and in addition it implements the transaction operation. A transaction request is composed of operations that modify features; that is create, update and delete operations on geospatial features.

FE is an XML encoding which is used as a system neutral representation of a query predicate. Query predicate which is generated as FE, can be easily validated, parsed and then transformed into whatever target language is required to retrieve or modify object instances stored in some a persistent object store [19]. Using FE, spatial and non-spatial query expressions can

be created to be sent to WFS. Figure 4, illustrates an EF expression which is used to retrieve a feature from City layer whose name is Tehran.

Geospatial web services vs. web services technologies: Geospatial web services can be considered as technology independent services. Geospatial Web service differs from the Web service technologies. The most important distinction between these two kinds of services is the fact that Web services are particular set of technologies and protocols but geospatial Web services are composed of defined set of interface implementation specifications which can be implemented with diverse technologies. Following items explain the most important differences between geospatial Web services and Web services technologies [3]:

- In the OGC Web service framework HTTP is defined as the sole distributed computing environment. In contrast, as Fig. 1 shows, Web services can be implemented virtually with any standard protocols such as HTTP, FTP and TCP to name a few. Also by considering the huge volume of geospatial data and the textual nature of HTTP protocol, it is good idea to use other binary protocols when the time is the most important factor in the exchange and share of geospatial data.
- OGC Web services do not necessarily use the usual Web services core standards, including SOAP and WSDL. In other words, in Web services technology, the main messaging protocol is SOAP and this protocol can be considered as the main cause of achieving interoperability among various applications which are running on heterogeneous platforms. In OGC Web service framework SOAP is not the main messaging protocol. In addition in most geospatial Web services, creation and publication of WSDL document has not defined yet.
- OGC Web services have particular interface for binding that might leads to interface coupling problem. In accordance with OGC Web service framework specifications, each geospatial Web services have to publish its own capabilities through a so called capabilities document. This document (which is an XML document) provides human and machine-readable information about the geospatial data and operation supported by a specific instance of a geospatial Web service. But this document is not comprehensive enough to play a same role as WSDL document. In other words, capabilities document cannot offer enough information to enable developers and thus software applications to consume a geospatial Web service programmatically and automatically, while according to Newcomer and lomow [5], ideally the service requester should be able to use a service exclusively based on the published service contract.

As mentioned before, geospatial Web services can be implemented using any existing software development technologies. It is suggested that using Web services technologies as enabling infrastructure for implementing geospatial Web services can significantly facilitate sharing geospatial data as well as access to processing services from multiple resources in and out of geospatial community. In other words, geospatial Web services which are developed using Web services technologies can provide data and access interoperability among various geospatial and non-geospatial processing systems.

XML DATABASE SYSTEMS

XML, as a rich set of technologies, is playing an important and increasing role in share and exchange of data over the Web. The more XML has been used in share and exchange of data, the more XML data management issues have to be considered. So, database researchers have actively participated in developing technologies centered on XML data management, in particular data models and query languages for XML. As a result of these researches, many XML data management systems have been implemented. In general, XML data management systems can be categorized as XML enabled databases and native XML databases [20].

Typically, an XML enabled database is a relational database which provides storage of hierarchical XML documents in relational model and provides proprietary methods for relational to XML data mapping (or conversion) for retrieving stored data as XML. The mentioned proprietary methods vary in each software package from extension to standard Structured Query Language (SQL) language to implementation of a full featured XML query language.

On the other hand, a native XML database has an XML document as its fundamental unit of logical storage, just as a relational database has a row in a relation as its fundamental unit of logical storage. A native XML database defines a logical model for its fundamental unit of storage and stores and retrieves XML documents according to that model (such as Document Object Model) [20]. The advantage of this native approach is that XML data can be stored and retrieved in their original formats and no additional

mappings or translations are needed. Furthermore, most native-XML databases have the ability to perform sophisticated full-text searches including full thesaurus support, word stubbing (to match all forms of a word: run, ran, running) and proximity searches [21].

Since there are two technologies for XML data management, some performance analysis and benchmarking have been performed by database practitioners and researchers. in this case most of the analysis and benchmarks consists of evaluation of using various methods for extracting XML data from relational databases [21-23] and evaluation of XML query languages [24, 25]. In some rare cases, two (or more) database products from both technologies were evaluated using a defined set of queries to indicate which technology provides better support for XML data management [20, 26]. Although, there is no conclusion yet as to which technology suits better for XML data management, most of the analysis and tests concluded that it is much easier to use native XML databases to manipulate XML documents than to use XML enabled databases [20]. In addition, with proper design native XML database has better performance and scalability than a XML-enabled database for handling XML documents with larger data sizes. In other words, a XML-enabled database has better performance for small document sizes (number of records less than 1,000) but it cannot handle large-sized documents as efficiently due to conversion overhead. (The native XML database engine directly accesses XML data without conversion [26]).

As mentioned earlier, GML is an XML-based technology for modeling, transporting and storing geospatial information. Since GML is based on XML, the same technology for managing XML data can be used to manage geospatial data stored in GML. Considering the fact that geospatial data are huge in volume, using native XML databases for storing geospatial data (as GML), provides an efficient solution for storing and accessing high volume geospatial data in multi-user enterprise environments. In addition, as more and more geospatial data is stored and exchanged in GML format, using XML technologies which are easy to integrate with native XML databases more spatial capabilities can be added to native XML databases. Adding spatial capabilities to native XML databases allow them to manipulate spatial data thus make them more efficient in handling geospatial data.

Considering the above descriptions, this paper suggests that coupling native XML database system, as efficient means for storing and managing geospatial data (in GML format), with geospatial Web services which are developed using Web services technologies, provides an open, interoperable and efficient geospatial information sharing environment.

IMPLEMENTATION AND EVALUATION OF A BASIC WFS

In order to evaluate the feasibility of using XML database systems as back end geospatial data store and Web services technologies as platform independent connecting technologies for implementing Geospatial Web services, a Basic WFS instance was designed and developed. This section describes the architecture and capabilities of the implemented system. As mentioned before, three operations have been defined for a Basic WFS; GetCapabilities, DescribeFeatureType and GetFeature.

These operations provide the software interface of the WFS system. In other words, internal details of the functional units and software components as well as communications are transparent to consumer applications; the consumer application just can communicate with the WFS system through the operations and defined set of parameters which are specified in WFS implementation specification. Software components, communication among them and physical location of each component are specified in logical and physical architecture of the WFS system.

Physical architecture is quite different from a logical architecture. The physical architecture is about the number of machines or network hops involved in running the application. Rather, a logical architecture is all about separating different types of functionality in software components [27].

Traditionally logical architecture of software applications consists of three tiers; presentation and user interface tier, business logic tier and data management tier. With the advent of new technologies and software design patterns the traditional logical architecture is rarely efficient for the modern software applications. Today, the business logic tier is often physically splits among a client, Web server and application server. In addition, with new software design patterns (such as façade, flyweight, adapter and composite) the business logic breaks up into multiple parts and components.

In this research the WFS designed in four logical tiers; presentation and user interface tier, business logic tier, data access tier and data management tier.

As the name implies, the presentation and user interface tier provides the end user a friendly tools which hides details of local and remote computational tasks from user. This tier is responsible for gathering the user inputs, validating the user inputs, composing FE statements based on the user inputs to make requests, validating requests against proper schemas, sending validated requests to WFS server and displaying the returned geospatial data.

The business logic tier includes all business rules for the WFS system. For the implemented WFS theses rules consist of translate requests to DBMS specific query language statements and dispatch them to the next tier.

Data access tier interacts with the data management tier to retrieve geospatial information. The data access tier doesn't actually manage or store the data; it merely provides an interface between the business logic and the database. Logically, defining data access as a separate tier enforces a separation between the business logic and how application interacts with a database. This separation provides the flexibility to choose later whether to run the data access code on the same machine as the business logic, or on a separate machine. It also makes it much easier to change data sources or data access technologies without affecting the application. It also makes it much easier to change data sources technologies without affecting the application. In addition by isolating the data access code into a specific layer, the impact of changes in data access technologies is limited to a smaller part of the application.

The last tier handles the physical retrieval, update and deletion of data. This is different from the data access tier, which requests the retrieval, update and deletion of data.

The mentioned four tier logical architecture have been developed using Microsoft.NET 2.0 framework and SQL Server 2005 DBMS..NET windows forms (as set of Class hierarchies for developing desktop applications using.NET framework) were used to implement the client side application (user interface and presentation tier). Windows forms provide much more flexibility and capability to use the client machine resources when compared with browser based applications. Web services infrastructure was utilized in all interactions between client side application and WFS server. In other words, WSDL was used to create software interfaces to enable remote communication between client side application and business objects of WFS server. SOAP was used to transport every interaction (request and response) between interfaces in client side and business logic over the Web.

In client side application, response to DescribeFeatureType operation specifies which feature types and properties can be requested using GetFeature operation. Then FE statements can be created using various logical and comparison operators which are provided as a part of user interface. The created FE then is sent to the WFS server and the requested geospatial data is sent back to client side application.

In addition to developed client side application, any Web browser or application can consume the implemented WFS using WFS specifications.

Two famous software patterns were used for implementing business logic and data access components respectively; façade and flyweight patterns.

Façade pattern provides a unified interface to a set of interfaces in a subsystem. A Façade pattern encapsulates a design feature where there is a simple interface that acts as a central point of reference for many interfaces [28]. In general, façade pattern forces interfaces to communicate with use of chunk of data as method parameters. This way of communication ensures the minimal network roundtrips and thus increases the performance of the application drastically.

Flyweight pattern is a pattern that allows client programmers to think that they are using a factory method to create their own object, when 'their' object is actually being shared by multiple clients. Normally, this is done to save memory and improve performance, by avoiding the creation of many equivalent objects [29]. In other words, the overhead required to continuously release and create any server-side resource that is frequently used and expensive to create for each client, limits the performance and reduces the maximum number of clients that can be served simultaneously; Flyweight pattern Manage the reuse of objects when it is either expensive to create an object or there is a limit on the number objects of a particular type that can be created and thus resolves the mentioned problem [30].

In the developed system, objects and components in business logic and data access tiers work together to prepare an appropriate response message. More accurately, user supplied parameters are parsed by business objects to determine which methods have to be executed. In the case of GetFeature operation, user supplied FE statements are translated to appropriate XQuery statements. Then XQuery statements are delivered to objects and components in data access tier to be sent to DBMS.

In the last tier of the architecture, geospatial data were stored as GML 3.1 in the back end native XML database. For retrieving geospatial data, XQuery statements which were sent by data access components are executed and result are sent back to the data access component. Data access components dispatch retrieved geospatial data to business logic components. Afterwards geospatial data are prepared to be valid against WFS specifications. Finally, prepared GML data are sent back to the client using Web services infrastructure (using SOAP).

For implementation of the data management tier Microsoft SQL Server 2005 DBMS were used. Microsoft SQL Server provides the best performance and compatibility when.NET framework is used to develop the data driven applications. This DBMS is by far considered as one of the most powerful commercial relational databases. In contrast with earlier versions

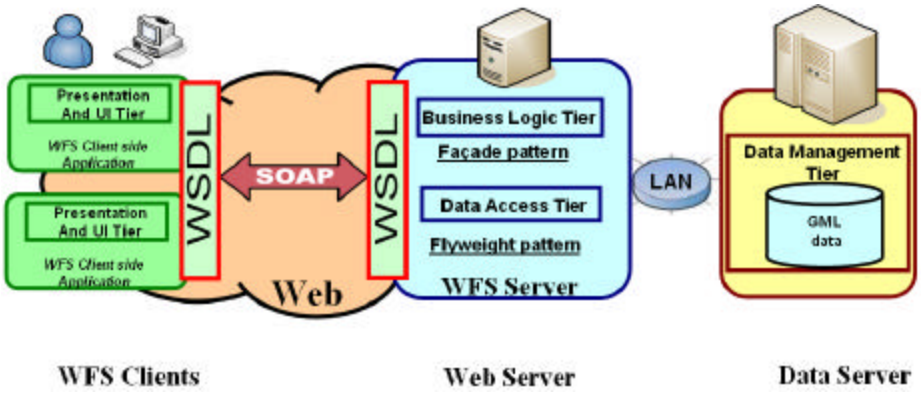

Fig. 5: Physical architecture of the implemented WFS

(such as SQL Server 2000), SQL Server 2005 defines a model based on SQL/XML (standard extensions to the SQL language specification) for an XML document and stores and retrieves documents according to that model [31]. As a result, SQL Server 2005 can be considered as a native XML database.

In implemented physical architecture of WFS, two server machines were employed. Data access and business logic components resided on the Web server machine (which utilized Microsoft Internet Information Services as Web server application) and native XML database installed on data server machine. Web server and data server machines connected through a high speed Local Area Network (LAN). As a result, Communication between data access components and DBMS were performed using well defined and DBMS specific binary protocols to optimize the performance. Figure 5, illustrates the implementation of the WFS physical architecture.

The four tier logical architecture of the implemented WFS ensures that the logical model has enough tiers to provide flexibility and extensibility. So, the WFS can be configured into an appropriate physical architecture that will depend on our performance, scalability, fault-tolerance and security requirements. The more physical tiers included, the worse the performance will be but the potential to increase scalability, security and/or fault tolerance will be improved.

In addition, making use of modern software patterns and appropriate software platforms as well as flexible physical architecture, the scalability of the implemented WFS is improved significantly when compared with the standard Web and Client-Server applications.

On account of using Web services technologies, interoperability among heterogeneous platforms is achieved. Since Web services are the foundation of new type of application-to-application communication, they provide an unprecedented opportunity to connect heterogeneous platforms and applications. With the help of Web services technologies, it is an easy task for a developer to utilize WFS functionality into almost any type of geospatial or non-geospatial processing system.

Besides, spatial data and access interoperability is achieved through the use of standard interfaces and data format of OGC Web service framework. Since native XML databases outperform other types of databases for storing and retrieving XML data, storing geospatial data as GML in native XML database, retrieving geospatial data from WFS no longer needs time and resource consuming process of data conversion. It is worth noting that data conversion causes problems for real-time geospatial data access. Furthermore, use of XQuery and other native XML manipulation and processing technologies inside native XML databases, geoprocessing functions can be developed more efficiently and resourcefully. If other kinds of databases or methods (rather than native XML database), are used for storing and retrieval of geospatial data, GML data have to be mapped to database specific internal data models (for example relational model). In this case geospatial data retrieval as GML (for WFS service), can be accomplished using one of the following methods:

- Utilizing DBMS engine to map stored data to GML format (XML enabled approach). In this approach there are one or more tables for each feature class. Using proprietary methods of DBMS, relational data should be converted to GML data. In SQL Server 2005 extension to standard SQL can provide such functionality. More accurately, in SQL Server 2005, FOR XML clause which is an add-on to the end of a SELECT statement returns normal relational data as XML data [32]. Various modes of FOR XML clause enable the DBMS to dictate how the relational data should be transformed into GML.

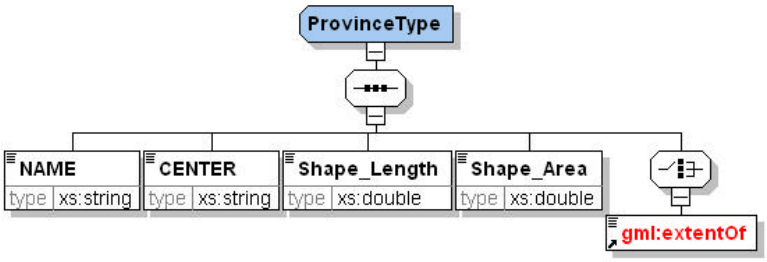

Fig. 6: ProvinceType schema

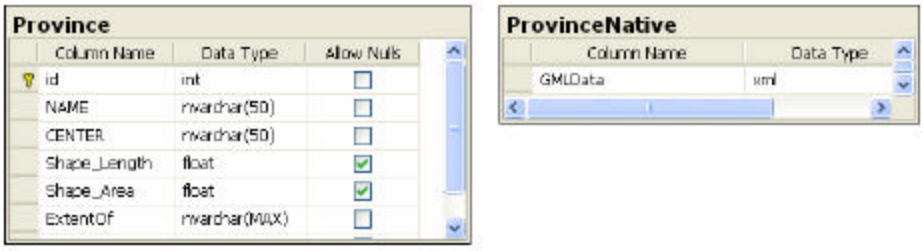

Fig. 7: Province and ProvinceNative tables to store geospatial data in relational and native (GML) forms respectively

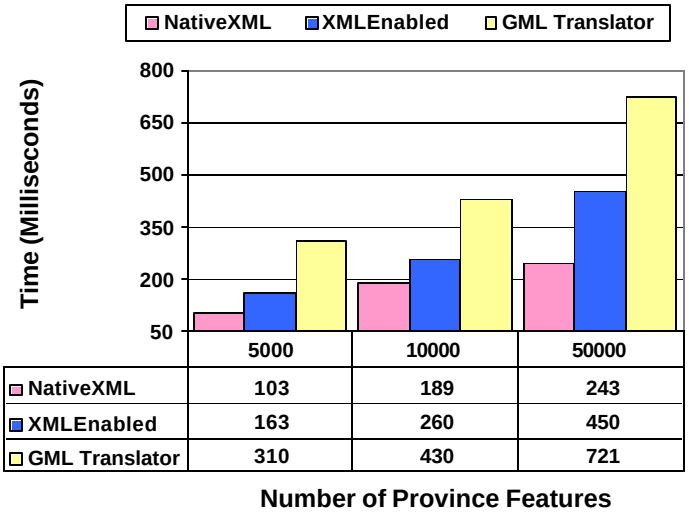

Fig. 8: Performance test for retrieving single province feature (Q1)

- Making use of dedicated component outside DBMS to retrieve data from database and turn it to GML data (GML translator approach). In this case, in business logic tier, there should be a dedicated component which is in charge of reformatting the retrieved data. Reformatting the data generates a GML document which conforms to the schema of feature class.

In both approaches, additional overhead required to map data to hierarchical structure of GML data, which consume some expensive computational resources (mostly memory of the data server) thus limits the overall performance of the system. If the high volume geospatial data are required, the mentioned issue can introduce a significant performance bottleneck. For ensuring that using native XML database is the most efficient approach, a simple performance test was performed. Next section briefly describes the test.

Performance test: In order to carry out the performance test, three GML document created using the ProvinceType schema (Fig. 6). The ProvinceType

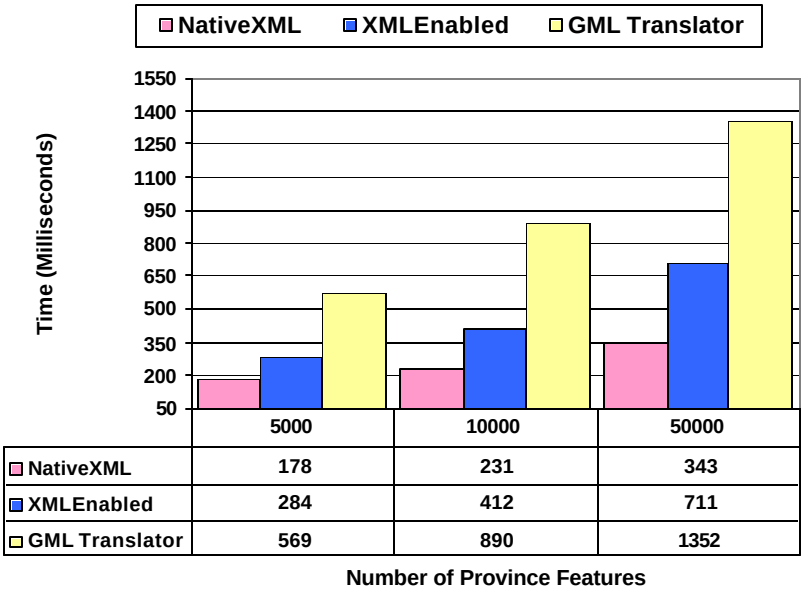

Fig. 9: Performance test for retrieving whole province features (Q2)

schema defines the structure of non-geometry and geometry properties of province features. Each province feature should have Name, Center, Shape_Length and Shape_Area properties. Geometry of each province feature (as polygonal features) is denoted using gml:extentOf element. The ProvinceType schema was used to create the Province table (Fig. 7). The Province table is used to store geospatial data in relational model in SQL Server 2005. For storing geospatial data as GML in their native form, NativeProvince table is created (Fig. 7). The ProvinceNative table just contains one field (of type xml) to store GML data.

For realistic testing of performance 5000, 10000 and 50000 randomly generated province features are inserted into Province and ProvinceNative tables. Since Basic WFS can only retrieve geospatial data from geospatial data store, two queries were designed to evaluate the performance of all approaches. First query (Q1) is designed to retrieving a single province feature and second one (Q2) evaluates the retrieval of whole GML documents. Running time of each query was used as performance metrics. Each query executed 1000 times and average running time was calculated (Fig. 8 and. 9).

As results of the test illustrated, native-XML database outperforms the XML-enabled and GML translator approaches in both queries. Since there is a direct mapping between the original GML document and its physical representation within the native XML database, native storage strategy is more efficient solution for retrieving geospatial data as GML. Also the results are more serious as the number of province features increases.

CONCLUSION

In this paper implementation of a geospatial Web service (Basic WFS) using native XML database system and Web Services technologies was described.

Structure and data model of XML (and hence GML) documents does not correspond to any schema model of the widely used database technologies. This issue has led to the implementation of so-called native XML databases. Since GML is based on XML, the native XML databases can be used to manage geospatial data. Based on practical tests of this research it is concluded that using native-XML databases provides an efficient solution for storing and accessing high volume geospatial data in multi-user enterprise environments.

Developing WFS with the use of cutting edge Web services technologies as well as make use of native XML database to store geospatial data as GML, provides spatial data and access interoperability among various geospatial processing systems. Since Web services technologies are foundation of cross-platform application-to-application communication, functionality of the implemented geospatial Web service can be simply added to any geospatial or non-geospatial processing systems which are running on heterogeneous platforms. Using Web services technologies for implementing geospatial Web services and utilizing native XML databases to manage geospatial data as

GML are new topics in GIS world. Hence more tests and evaluations are needed to prove their efficiency.

REFERENCES

1. The Open GIS Reference Model, 2003. Available at: http://portal.opengeospatial.org/files.
2. Peng, Z., 2004. GML, WFS, SVG and the future of Internet GIS. GIS Development Magazine July 2004. Available at: http://www.gisdevelopment.net/ magazine/years/2004/july/38.shtml
3. Amirian, P. and A.A. Aleshiekh, 2008. A Hybrid Architecture for Implementing Efficient Geospatial Web Services: Integrating.NET Remoting and Web Services Technologies. World Applied Sciences Journal 3 (1): 140-153.
4. Volter, M., M. Kricher and U. Zdun, 2005. Remoting Patterns: Fundamental of Enterprise, Internet and Realtime Distributed Object Middleware, New Jersey, USA, John Wiley and Sons, Inc.
5. Newcomer, E. and G. Lomow, 2005. Understanding SOA with Web Services, Maryland, USA, Addison Wesley, Inc.
6. Nance, K.L., B. and Hay, 2005. Automatic transformations between geoscience standards using XML, Computers and Geosciences, 31 (9): 1165-1174.
7. Stal, M., 2002. Web Services: Beyond Component-based Computing. Journal of Communications of the ACM, 45 (10): 71–76.
8. Booth, D., H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris and D. Orchard, 2004. Web Services Architecture. W3C Working Group. Available at: http://www.w3.org/TR/ws-arch
9. W3C, 2006. The World Wide Web Consortium. Web Services Activity Statement. Available at: http://www.w3.org/2002/ws/Activity.
10. Gailey, J.H., 2004. Understanding Web Services Specifications and the WSE. Washington, USA, Microsoft Press.
11. Lake, R., D. Burggraf, M. Trinic and L. Rae, 2004. Geography Markup Language, Chichester, England, John Wiley and Sons.
12. Zhang, J., J. Gong, H. Lin, G. Wang, J. Huang, J. Zhu, B. Xu and J. Teng, 2007. Design and development of Distributed Virtual Geographic Environment system based on web services. Information Sciences, 177 (19): 3968-3980.
13. Lake, R., 2005. The application of Geography Markup Language (GML) to the geological sciences. Computers and Geosciences, 31 (9): 1081-1094.
14. Kokkonen, T., A. Jolma and H. Koivusalo, 2003. Interfacing environmental simulation models and databases using XML. Environmental Modelling and Software, 18 (5): 463-471.
15. Sen, M. and T. Duffy, 2005. GeoSciML: Development of a generic GeoScience Markup Language. Computers and Geosciences, 31 (9): 1095-1103.
16. Nativi, S., J. Caron, E. Davis and B. Domenico, 2005. Design and implementation of netCDF markup language (NcML) and its GML-based extension (NcML-GML) Computers and Geosciences, 31 (9): 1104-1118.
17. Antoniou, B. and L. Tsoulos, 2006. The potential of XML encoding in geomatics onverting raster images to XML and SVG Computers and Geosciences, 32 (2): 184-194.
18. Open GIS Consortium Web Feature Service implementation specification 1.1.0., 2005. Available at: https://portal.opengeospatial.org/files.
19. Open GIS Consortium Filter Encoding Implementation specification, 2005. Available at: https://portal.opengeospatial.org/files.
20. Lu, E.J., B.C. Wu and P.Y. Chuang, 2006. An empirical study of XML data management in business information systems. Journal of Systems and Software, 79 (7): 984-1000.
21. Atay, M., A. Chebotkoa, D. Liua, S. Lua and F. Fotouhi, 2007. Efficient schema-based XML-to-Relational data mapping. Information Systems, 32 (3): 458-476.
22. Liu, J. and M. Vincent, 2004. Querying relational databases through XSLT. Data & Knowledge Engineering, 48 (1): 103-128.
23. Fong, J., H.K. Wong and Z. Cheng, 2003. Converting relational database into XML documents with DOM. Information and Software Technology, 45 (6): 335-355.
24. Hausteina, M. and T. Harder, 2007. An efficient infrastructure for native transactional XML processing. Data and Knowledge Engineering, 61 (3): 500-523.
25. Jea, K.F. and S.Y. Chen, 2006. A high concurrency XPath-based locking protocol for XML databases. Information and Software Technology, 48 (8): 708-716.
26. Chaudhri, A., A. Rashid and R. Ziecari, 2003. XML Data Management: Native XML and XML-Enabled Database Systems. Wokingham, England, Addison Wesley, Inc.
27. Lhotka, R., 2006. Expert VB 2005 Business Objects. 2nd Edn. California, USA, APress Publishing.

28. Horner, M., 2006. Pro.NET 2.0 Code and Design Standards in C#. California, USA, APress Publishing.
29. O'Docherty, M., 2005. Object-oriented analysis and design: Understanding system development with UML 2.0., New Jersey, USA, John Wiley and Sons, Inc.
30. Shalloway, A. and J. Trott, 2004. Design Patterns Explained A New Perspective on Object-Oriented Design Second Edition. Maryland, USA, Addison Wesley, Inc.
31. Amirian, P., 2006. Design and Development of a Distributed Geospatial Web services using XML and.NET technologies. Geospatial Information Systems (GIS). MS Thesis, K.N. Toosi University of technology, Tehran, Iran.
32. Klein, S., 2006. Professional SQL Server 2005 XML. Indianapolis, Indiana, Wiley Publishing, Inc.