

Efficient way of Exchanging Geospatial data over the Web

Pouria Amirian ^a, Ali Mansourian ^b

Faculty of Geodesy and Geomatics Engineering, K. N. Toosi University of Technology, Vali-e-asr St., Mirdamad Cross, Tehran, Iran, P.C. 1996715433

^ap_amirian@hotmail.com, ^bmansourian@kntu.ac.ir,

Abstract

Nowadays, in the GIS world, same as other IT related disciplines, the Internet and its applications, particularly the World Wide Web (WWW), make exchange of geospatial data and publishing geospatial information much easier than ever before. On account of the nature of the Web as a human-centric media, the existing Web is an efficient platform for publishing geospatial information. But there are some barriers in front of fully utilize the potential of Web as the ubiquitous infrastructure for exchanging geospatial data.

Today the vast majority of published geospatial services over the Web do not provide the ability to integrate geospatial data from various sources remotely and in an efficient manner. In this context two sources for this issue can be identified; first these services don't offer their own geospatial data in an open and standard format and second, even if these services offer their own geospatial data in open and standard format, there must be an efficient infrastructure to allow them to communicate directly. In other words, different geospatial services which are running on various platforms should be able to communicate with each other in open and vendor neutral way. But in this context unique characteristics of geospatial data should be considered as well.

By building on broader Internet standards from the World Wide Web Consortium (W3C), the GIS community has developed an XML-based markup language, known as GML (Geography Markup Language). GML is used to express geospatial data in a manner that can be readily published on the Web. For fully access to geospatial data in a distributed environment openly and in cross platform manner, coupling geospatial data which are encoded in GML with efficient technologies and platforms seems to be an efficient solution.

This paper intends to deal with the possible interaction paradigms among service providers and service requesters in GIS world.

Keywords: Message Exchange Patterns, Internet GIS, Interoperability, Web services Technologies

Introduction

Since various Internet GIS software use proprietary way of communication, they can't communicate and interact with each other directly. In other words, there is a need for direct application to application communication and this is the major idea of application-centric Web rather than human-centric Web.

Today the primary model on which the Web is operating is based on human interactions. Humans are the primary actors for initiating the most of web requests. The Web is intended for human consumption. Consequently, data is presented in a form that is human-readable (human-centric Web). But this form of data is error prone and difficult for applications to examine, extract and use both, automatically and programmatically. In addition Internet-based applications need to be able to interact with other internet-based applications as easily as interactions between Web browsers and Web servers (application-centric Web). However, the essentially text-based Web does not support software interaction very well. By considering the new advances in Web service domain, these brand new technologies can overcome the mentioned problem with current Internet GIS applications.

Web service Technologies and GIS Community

Web services are based on open standards, so they can provide access interoperability in network environments such as Internet. In addition, to providing access interoperability, they can be created by using any platform, operating system, programming language and object model. More precisely, Web Services are loosely coupled, self-describing services that are accessed programmatically across a distributed network, and exchange data using vendor, platform, and language neutral protocols (Marks and Werrell, 2003). Therefore, Web services are the main candidate for GIS application integration and implementation of truly distributed GIS solutions. In addition to application integration and ability to providing GIS functionalities, The Web service architecture could establish a particular type of relationship between GIS service providers and requesters that supports the process flows efficiently. Following sections will briefly introduce the Web services Technologies and explains the concept of Message Exchange Patterns (MEPs).

Web service architecture

Web services are a fundamentally new framework. They consist of a set of standards for supporting network transportation, service communication, service publication and service discovery. In some cases Web services considered as a new architecture for software systems. According to Marks and Werrell (2003), today the new paradigm for software system architecture is emerged. Software system architecture evolved from monolithic mainframe systems to diverse architectures of client-server to distributed systems to service oriented systems which have been implemented using Web service architecture.

There are three major roles or components within the Web service architecture. Major components in the Web service architecture are service provider, service requester and service broker.

Service providers publish their service(s) with the registry repository of a service broker. Then, a service requester initiates a search for a service by contacting the service broker and searching the registry repository for services that meet specific search criteria. The broker returns a list of services along with details of the associated

provider for each service. Up to this point, the service requester has found the desired service(s). Subsequently, the service requester binds with a selected service provider(s) based on the provided details of registry repository and consumes them. In the mentioned scenario, there are series of standards and protocols which enable communication, description, publication and discovery of Web services. The standard mechanism for publication and discovery of Web services is UDDI (Universal Description, Discovery and Integration). UDDI provides a logical centralized registry repository of Web services. The UDDI registry repository allows service providers to publish their services by using another standard called WSDL (Web Service Description Language).

The service requesters can find desired services through searching in the registry repository. Based on the contact specification which is provided by service provider (most of the time it is the WSDL document), service requester can make a communication link to the Web service and consume it. For sending requests and response, Web service architecture can take advantage of different network transport protocols (such as FTP, HTTP and SMTP to name a few). The service communication layer is the most important component of Web service architecture for supporting direct communication of applications. In the service communication layer, another XML-based protocol is used: SOAP (Simple Object Access Protocol).

The flexibility and extensibility of the SOAP and WSDL equipped service providers and service requesters with various message exchange patterns. The following section will describe the various message exchange patterns.

Message exchange patterns through Web services

Messaging or communication between applications is the main idea of Web services. Web services can be simply defined as any service which is available over the Internet and uses standardized XML messaging system. The standardized messaging mechanism was the main problem of previous distributed object technologies such as Microsoft's DCOM (Distributed Component Object Model), OMG's CORBA (Common Object Request Broker Architecture) and Java RMI (Remote Method Invocation).

Another important fact about the mentioned distributed object technologies (which can be considered as an important advantage of Web Services Technologies over the other distributed object technologies) is that most of them were suffered from lack of flexibility and extensibility in providing efficient messaging patterns. More accurately, they accomplished the communication in RPC (Remote Procedure Call) manner, which is the application sending the message waits for a response. Also this problem can be solved through the use of message queuing systems, but this solution adds another level of complexity and another point of failure to the systems.

As oppose to the mentioned problem, the flexible and extensible nature of the Web service protocol stack allows various message exchange patterns to be implemented. In Web services arena, the most common message exchange patterns are the following:

- Request/response interactions.

- Request/callback interactions.
- Store-and-forward messaging.
- Publish/subscribe interactions

Request/ Response Pattern

This pattern is similar to the RPC communication in DCOM and Java RMI. In the request/response pattern, the service requester sends the request and then waits for the reply. In other words service requester will be blocked until the response is replied (Figure 1).

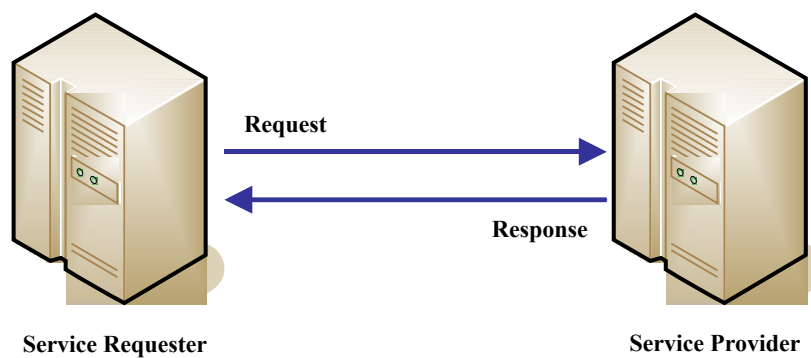


Figure 1: Request/Response Paradigm

Although WSDL and SOAP both support a request/response interaction style, it is important to understand that the SOAP and WSDL definitions are not executable and that any business logic needs to be implemented by a run-time environment such as J2EE, .NET Framework, or CORBA (Newcomer and Lommow 2005). In GIS World this pattern can be used in the situation in which a specific analysis or low volume geospatial data is needed in order to perform another (local) process. In this case, the execution of the local process is blocked until the required action takes place. This pattern is the foundation of next generation of distributed GIS applications in which functionality provided remotely and through the use of the registered and also trusted service providers over the decentralized environment (such as WWW).

Request/Callback Interaction Paradigm

The request/callback interaction paradigm is usually utilized when the service requester cannot be blocked while waiting for a synchronous response, so instead it sets up a callback agent (or process) to handle the response. Figure 2, illustrates the request/callback interaction paradigm.

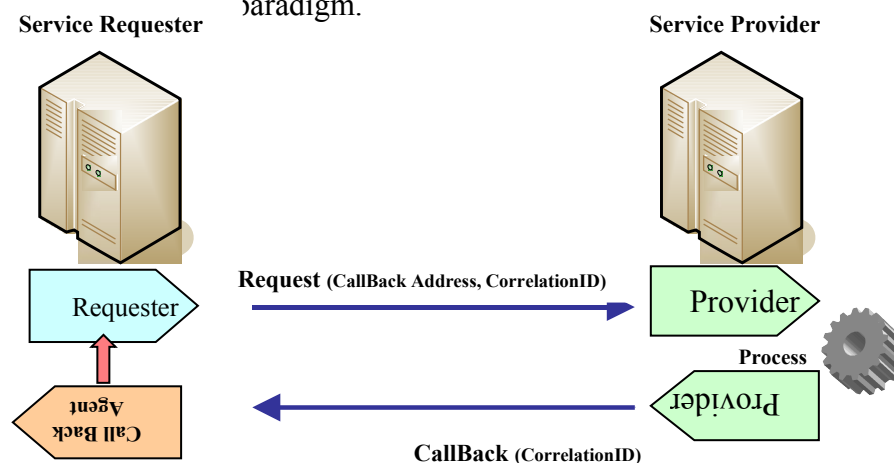


Figure 2: Request/Callback Paradigm

In Request/Callback pattern, The typical sequence of actions is (Figure 2):

1. The service requester sends a request message to the service provider using a one-way request message. The request message includes a correlation ID and a callback address. After the service requester sends the request message, it continues executing and does not block while waiting for the response (for this reason the service requester sends the one-way message).
2. The service provider receives the request message, composes a response, and sends a callback message to the callback service by sending a one-way response message to the callback address that was included in the service requester's original request, including the correlation ID.
3. The callback service receives the response message and processes it as appropriate (which may include notifying the service requester of the response or dispatching the result of a remotely processed task).

This pattern can be employed when there is a need to perform some (more than one) tasks in sequential order. An example of this situation is the chain of geoprocessing analysis in which the output of the first task will be the input of the second one. Since that is a time consuming task, the requester should be unblocked to be able to perform other tasks. In addition to chain of services, when high volume geospatial data is needed to be downloaded from remote resource, this pattern provides an alternative solution over the use of queuing systems. In Web service protocol stack, WSDL and SOAP do not provide formal support for request/callback interactions, and it is up to the application layer to manage the various elements of the request/callback interaction, including defining callback addresses and generating correlation IDs (Newcomer and Lommow 2005).

Asynchronous Store-and-Forward Messaging

According to make use of store and forward paradigm, a message queuing system should be used and coupled with the service provider and service requester Web service system. In this interaction style, communication between service provider and service requester is accomplished via the use of persistent queues. Since both the service provider and service requester roles can be interchanged at any time, this style are well suited for the partial connected and mobile clients. In this case the service on mobile client places a request message in a request queue, and the messaging

technology reliably delivers the message to another mobile client or the main service provider where the receiving service dequeues it and processes it (when the connection is available) (Figure 3).

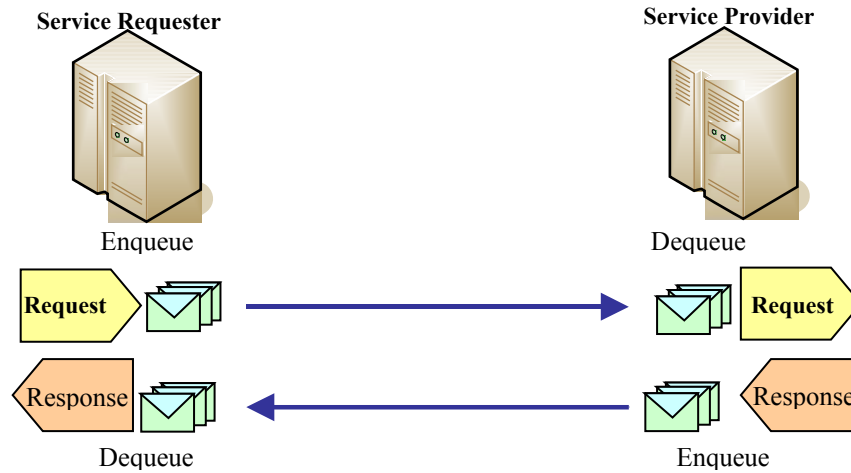


Figure 3: Store/Forward Paradigm

One of the advantages of this message exchange pattern is that a request queue can be persistent, allowing the application to continue working whether or not a connection to the remote machine is available. As mentioned before, this paradigm is the main candidate for the mobile GIS applications.

Publish/Subscribe Interaction Paradigm

With publish/subscribe interactions, event subscribers indicate which event types they are interested in, and then they receive event notifications when event publishers generate events in which they are interested (Figure 4).

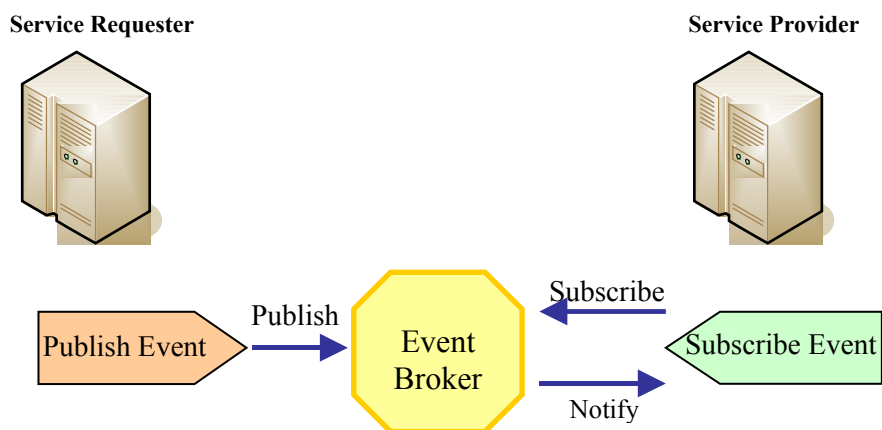


Figure 4: Publish/Subscribe Paradigm

Publish/subscribe is the most loosely coupled interaction paradigm because there is a dynamic, many-to-many relationship between event publishers and event subscribers:

- There can be any number of publishers for any type of event.
- There can be any number of subscribers for any type of event.
- The number of publishers and subscribers can change at any time.

This is different than request/response interactions where a service requester sends a single request to a single, well-defined service provider.

This style of interactions can be used in enterprise-level Object Oriented Geospatial Data Models. In Object Oriented Geospatial Data model, geospatial entities (features) are not limited to have just spatial and non-spatial properties. In contrast, they can fire events in certain situations such as when a specific spatial or non-spatial property value has changed or in response to change in related or neighbourhood features. Implementation of such data model is a complicated task and this model is not implemented in large scales to the present time.

Conclusion

Web service technologies are the indispensable part of future interoperable and distributed applications and currently, these technologies are supported by all major players in IT business, including Microsoft and IBM. So this is the responsibility of GIS community to join to this mainstream and also supply specific GIS related new technologies to Web service protocol stack.

By use of Web service and XML technologies, GIS community can overcome the non-interoperability of current Internet GIS software. Meanwhile, there are lots of issues due to immature nature of Web services to be resolved. In addition, considering the nature of spatial data and GIS, some more specific issues are faced by GIS Web services such as high volume of spatial data, methods of compression and supporting the complicated GIS workflows that should be dealt with.

Development of a GIS Web service based on the mentioned message exchange paradigms in this paper and evaluating the use of XML-based technologies for supporting GIS functionalities is ongoing.

References

- Amirian, P (2006). *Design and Development of a Geospatial Web Service Using .NET and XML Technologies*, MSc Thesis, K.N.Toosi University of Technology, Tehran, Iran.
- Amirian, P, Mansourian, A (2006). *Integration of GML and Web Services Technologies for Implementing Distributed GIServices*, GeoWeb 2006, British Columbia, Canada.
- Gailey, J, H (2004). *Understanding Web Services Specifications and the WSE*. Washington, USA, Microsoft Press
- Hariri, S, Parashar, M (2004). *Tools and Environments for Parallel and Distributed Computing*, New Jersey, USA, John Wiley & Sons, Inc.
- Marks, E and Werrell, M (2003). *Executive's Guide to Web Services*, New Jersey, USA, John Wiley & Sons, Inc.
- Newcomer, E (2002). *Understanding Web Services*, Wokingham, England, Addison Wesley, Inc.
- Newcomer, E and Lomow, G (2005). *Understanding SOA with Web Services*, Maryland, USA, Addison Wesley, Inc.
- Volter, M , Kricher, M, Zdun, U (2005). *Remoting Patterns: Fundamental of Enterprise, Internet and Realtime Distributed Object Middleware*, New Jersey, USA, John Wiley & Sons, Inc.