Proceedings of the 6th AGILE
April 24th-26th, 2003 – Lyon, France

EXTENDING OPENGIS SERVICES TO GAIN ACCESS TO REAL TIME DATA THROUGH A WEB-COORDINATION SERVICE

P. Álvarez, J.A. Bañares, R.Bejar, M.Á. Latre

Computer Science and System Engineering Department, University of Zaragoza,
María de Luna 1, 50018 Zaragoza (Spain)

{alvaper, banares, rbejar,latre}@unizar.es

1. INTRODUCTION

When GIS and Location services are got together to provide Location-Based Services through Internet, Web-services interoperability and orchestration problems become obvious due to the fact that service-oriented architectures have not solved the distributed computing difficulty in “gluing together” multiple and independent Web services. Besides, if Web services exchange real time data, their execution environment is even more collaborative and dynamic. An interaction model based on a request/response scheme, such as the HTTP-protocol model, is not enough to solve these problems. It is necessary to define and support new communication and coordination models among Web services, which functionality is accessible through open technologies and standards according to the Web-service philosophy. In this context, this paper presents a model able to solve the identified lacks and to detail how services must be extended to obtain a collaborative and reactive behaviour. This work has been applied on an experiment to integrate and orchestrate OpenGIS geoprocessing services and LIF Location services. As a result of this experience, new abilities of the proposed model has been found and used in this GIS domain, such as an opportunistic strategy to chain Web services.

The concept of Web services is an evolving idea from traditional middleware platforms. The main reason of their success is due to the fact that web services solve the difficulties for the application-to-application integration in distributed environments. Entities communicate and interchange data among them using standard Internet protocols and data formats, such as HTTP and XML. In addition, for ensuring syntactic interoperability, all key players in the industry are focused on the development of wide accepted and open standards to define public service interfaces.

Geographic Information Services (GIS) and Location-based Services (LBS) are two prototypical technological contexts where many standardization initiatives have arisen to define open, ubiquitous and interoperable service interfaces. LBS extend GIS spatial processing capabilities by integrating wireless communications, location data and Internet technology [Niedzwiadek00, ESRI00]. In this context, two well-positioned organizations have emerged as the leaders of the LBS interoperability: LIF (Location-Interoperability Forum, http://www.locationforum.org/) and OGC (Open GIS Consortium, http://www.opengis.org/, and its Open Location Service Initiative (OpenLS), http://www.openls.org/). Both are promoting and defining standard interfaces for a collection of wireless, Location and GIS services to provide the required LBS functionality [OGCSA02, OpenLSTestbed00, VanderMeer02]. These public interfaces facilitate the integration of individual computing entities that, in an isolated way, would have a very limited functional value into end applications through Internet.

In this technological context, it is necessary to integrate the OpenLS and LIF services to provide LBS functionality that may be incorporated into end applications, such as automatic vehicle monitoring systems [ZiliaskopoulosTravis00, Thill00], decision support systems [ABMZ02] and emergency or natural risk management systems [ABFM02]. At a first experience (figure 1.), a high-level service to track mobile resources with an installed location-device (such as emergency vehicles or employees with a mobile phone with an integrated GPS-receiver) was developed. To visualize their real-time positions, the service combines the digital map returned by a Web Map Server (WMS) and the result of a Tracking Web Feature Server (TWFS) query and applies a visualization style specified by the service client to the displayed geoinformation. The TWFS was developed from a previous WFS implementation to be used like a repository of "mobile" features and to exploit its standard geospatial query language. A communication component acquired real-time location data from the remote mobile devices and a wrapper inserted them into the TWFS database. This communication component was the core of another Location Service developed according to the LIF proposal. The wrapper was an ad-hoc solution to chain location data acquisition and GIS services, whose responsibility was to solve interoperability problems among them.

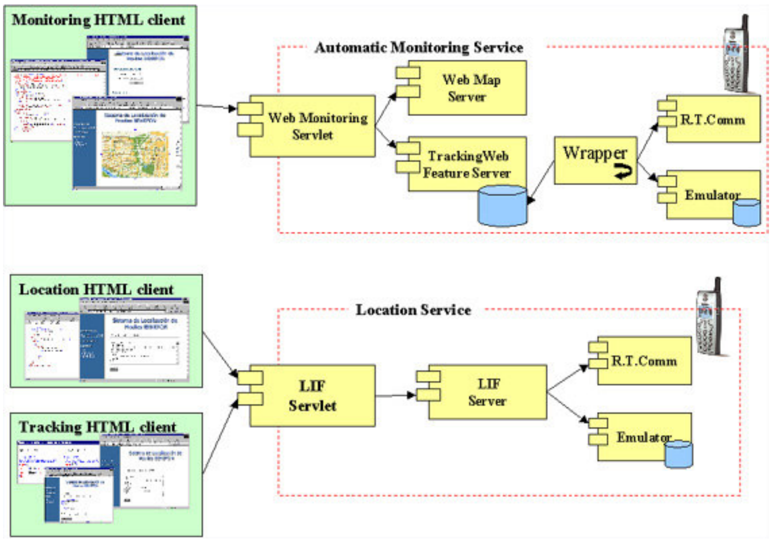

Fig. 1 Ad-hoc integration of OpenLS and LIF services

After this ad-hoc service chaining, the next step is to achieve a generic solution to enable the LIF Location service to provide real-time location data directly to the high-level tracking service. This is a prototypical example of Web-services orchestration. Some interoperability problems have been discovered in the orchestration of OpenGIS and LIF servers, and they may be extrapolated to the general orchestration of Web services: 1) Location services have not persistence, therefore if many different clients need to know real-time positions of the same mobile, each one must invoke an independent HTTP request. Each request supposes a wireless communication with the remote mobile device. 2) When a client makes an HTTP request to get a real-time location, the amount of time that is required for receiving the requested location is not known, due to wireless network delays. The HTTP protocol provides a synchronous request/response model, but an asynchronous model is more adequate in this case. 3) And finally, in other cases, to intensify the necessity of an asynchronous model, the initiative comes from the mobile device. For example, remote devices generate alarms or events when come into/out from a circular geographical region.

The problems found are essentially Web-service coordination and orchestration ones. To solve them, it is necessary to provide a persistent repository to store and forward interchanged data among services, at least for a period, and new communication models to support a collaborative work among them. Two new models should be provided: 1) a many-to-many communication model, instead of a peer-to-peer one, such as models used by traditional object-oriented middleware platforms (CORBA, DCOM, JavaBean) or those based on RPC (SOAP), and 2) an asynchronous and reactive interaction model. These coordination functionalities will be used for defining chains of services, synchronizing these services with themselves and with applications, or building more complex interactions than the HTTP-based ones. The problem of service collaboration on the Web resembles the object collaboration in a distributed object-oriented system one. However, this complex problem has not been dealt with yet with enough interest.

This work presents an approach based on an extension of the Generative Communication model to coordinate services through Internet. The selected model and its improvements are described, and its advantages are justified. A new Web-coordination service has been developed from the improved model and it used in the example presented to solve the interoperability problems identified among OpenLS and LIF services. Finally, the results of this approach and the future work are detailed.

2. A MODEL OF COORDINATION FOR DISTRIBUTED SERVICES

The basic idea is to use a coordination model that acts as a glue linking services in an Internet environment. Instead of starting from the scratch, the model known as Generative Communication has been chosen, and more precisely the so-called LINDA model [CarrieroGelernter89]. LINDA is a very abstract artifact based on two concepts: tuple and tuple space. A tuple is something as [“Gelernter”, 1989] , where the components are supposed to be untyped atomic values. The tuple space is a collection of tuples that acts as a shared memory, to which certain operations can be applied in a decentralized manner. For instance, the operation *in(x?)* tries to match the tuple *x?* with a tuple in the shared space. If there is a match, a tuple is extracted from the tuple space; otherwise, it blocks until a convenient tuple appears. The parameter for *in()* can be a query tuple with a wildcard, like [“Gelenter”, ???]. The match is then free for the wildcard and literal for the constants values. Our proposal is based on this obvious observation: if this simple matching strategy is replaced with a *complex matching*, then very general kinds of interoperability can be achieved.

Considering this basic idea, let us particularize the concept of complex *matching*. To this aim, we work with a version of LINDA where the tuples admit a description by means of attribute/value pairs, like:

[(author, “Gelernter”), (year, 1989)]

Although this is still an untyped setting, this bit of structure, allows recovering information from a distributed context. Thus, if an operation *in()* is invoked on a different entity, where the term “author” is not used, but “creator” is used in its place, and if there is a convenient mapping between ontologies, then the request [(creator, ???), (year, 1989)] can be successfully satisfied. This kind of semantic matching is implemented through Internet, using XML as transfer format. For the presented approach, it is important to remark that the “top-level” structure of any XML document admits the expression

[(att1, <val 1>),…,(attN, <val N>)]

but where each <val i> is structured (in particular, it may be XML-based).

Besides, if LINDA is oriented to be used in an open and hostile environment where entities communicate using synchronous protocols, such as Internet and HTTP respectively, new uncoupled and asynchronous operations must be provided for supporting non-instant match. A collection of event-based operations suggests the possibility of improving the proposal of the Communication Generative model, adding a more reactive interaction style. Entities subscribe their interest in receiving event notifications when other writing entities insert specific tuples into the shared space, instead of being blocked until tuples appear.

Now LINDA can be used in a XML context supporting a semantic and structured matching, and providing a collection of operations that promise an interesting way to coordinate, communicate and collaborate on the net.

3. ADVANTAGES OF THE COORDINATION MODEL BASED IN GENERATIVE COMMUNICATION

To make possible the described coordination-style among services through Internet, the solution is to design and develop a Web-Coordination service (WCS) that encapsulates the proposed model. Using this new service, distributed services over Internet could communicate and synchronize among them. This WCS has been built in accordance with the Web-service philosophy of making its integration into open architectures easier: coordination operations must be accessible via ubiquitous and universal Internet protocols and data formats, such as HTTP and XML; its open interface must hide service implementation details, such as hardware or software platform on which it is implemented or the programming language in which it is written; and it must encourage a flexible and loosely-coupled interaction among Web services.

This WCS provides a persistent repository for exchanging XML data and a collection of writing and reading operations used by service clients to store and retrieve data into/from it. An important fact of the model is that writing clients do not need to know anything about reading ones and *vice versa*. It promotes what it is called an uncoupled interaction style in the space. But besides, due to the event-based operations that have been included, it is being promoted an uncoupled style in time, too.

Instead of beginning from the scratch the core of the WCS, it has been implemented using JavaSpaces Technology [FHA99], an implementation of the LINDA model based on Java programming language and Jini services. It provides a shared repository of Java objects and LINDA model operations to write and read objects into/from it. Besides, it includes the Jini Distributed Event Model to support event notifications when objects are written into the shared space that matches templates.

On the other hand, the implemented WCS allows Web services to collaborate according to an opportunistic strategy. A Web service communicates or coordinates with other Web services, the most suitable ones at the moment. For example, performing vehicle-tracking tasks through Europe will depend on the country the vehicle is in: it will be necessary to request location data from different Location services provided by local telecommunication companies in each specific country. By using an opportunistic strategy, clients that receive location data through the WCS do not need prior knowledge about the providers who are generating data. That is possible due to the fact that the Generative Communication model is similar to a *blackboard* architecture, that represents an opportunistic strategy to solve problems.

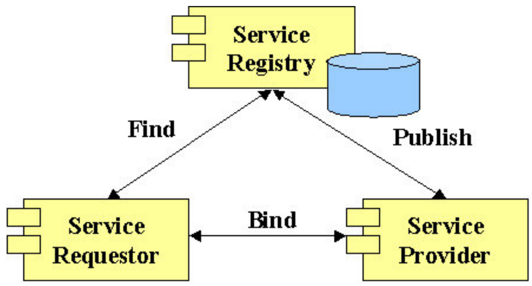

Fig. 2 Service-Oriented Architecture

It is important to emphasize the difference regarding to an approach based on a Service-Oriented Architecture (SOA). When a service requestor finds a service description published into a service registry, it is its responsibility to use the service description to bind to or invoke Web services hosted by the service provider (see figure 2.). The service requestor works with the provider previously found until it decides to search for another different provider in the service registry. However, by using the proposed WCS for communicating services, or, in a future, for searching service providers, Web services are able to retrieve available data or descriptions into the shared space in an opportunistic way. They need not worry about which service or services are providing these data.

Finally, the last key element in this coordination strategy through Internet is how to extend Web services to integrate them in this collaborative and dynamic environment. A Web service may be interested in retrieving from the WCS coordination XML-tuples and executing actions to modify its internal state according to the tuples read. To support this reactive behaviour, an internal process associated to the Web service is responsible for reading tuples and executing actions. But a Web service can also act on its own initiative. It can be programmed for generating events like XML-tuples when it reaches a specific internal state. Internal daemons detect these states and generate the respective events into the WCS. In the next section, LBS examples are presented to illustrate this reactive behaviour.

4. SOLVING OPENLS AND LIF INTEGRATION PROBLEMS

Once our coordination approach has been presented, we return to the initial example to explain how it is able to solve the real-time data communication problems identified among Location service and Tracking Web Feature Services. Figure 3 shows how involved services collaborate through the WCS. Location services publish through Internet real-time location data according the LIF specification into the WCS. On the other hand, one or more TWFS are subscribed into the WCS to be notified when new location data are inserted into the shared space. When a notification is received, the TWFS internal process recovers the new location data and inserts it into its database updating the mobile location.

So, the shared space encapsulated into the WCS stores location data provided by the Location services. Once these data have been inserted, they can be retrieved by many service requestors, solving the 1) problem. Besides, the coordination model proposed offers publishing and subscribing operations based on events over HTTP to support an asynchronous communication style, solving the 2) and 3) problems. Web-services can show their interest and subscribe to the WCS to be notified when new real-time data, alarms or events are published into the WCS.

The sending of events over HTTP protocol is implemented by means of HTTP Streaming. It entails maintaining open socket connections and sending asynchronous data through them. This technique allows the sending of events to different types of clients: to light clients, such as an HTML page that visualizes the location of a vehicle over a digital map, sending them JavaScript events through open connections; or clients implemented in any programming language able to establish an HTTP connection (Java, Lisp, C++, …).

In addition, clients of the high-level service to track mobile resources can create demons into the TWFS to filter location events, for example, when a mobile comes into/out a geographical area (street, building block, town, state…) or when it is around 10 km of a specific area. The TWFS extends its interface to offer a new operation, called *SetDaemon*, which has a similar syntax to the *GetFeature* request of the standard Web Feature Server interface (the spatial filter is specified using the *Filter Encoding Specification*). However, its effect is to create a internal daemon in the TWFS to publish location events according the specified filter, and to show to the requester client where and how it must subscribe to be notified when events happen. When a location data is updated in the TWFS by the internal process, the spatial filter is applied and, if it is valid, the TWFS inserts the location event into the WCS. Finally, this WCS notifies to all clients previously subscribed.

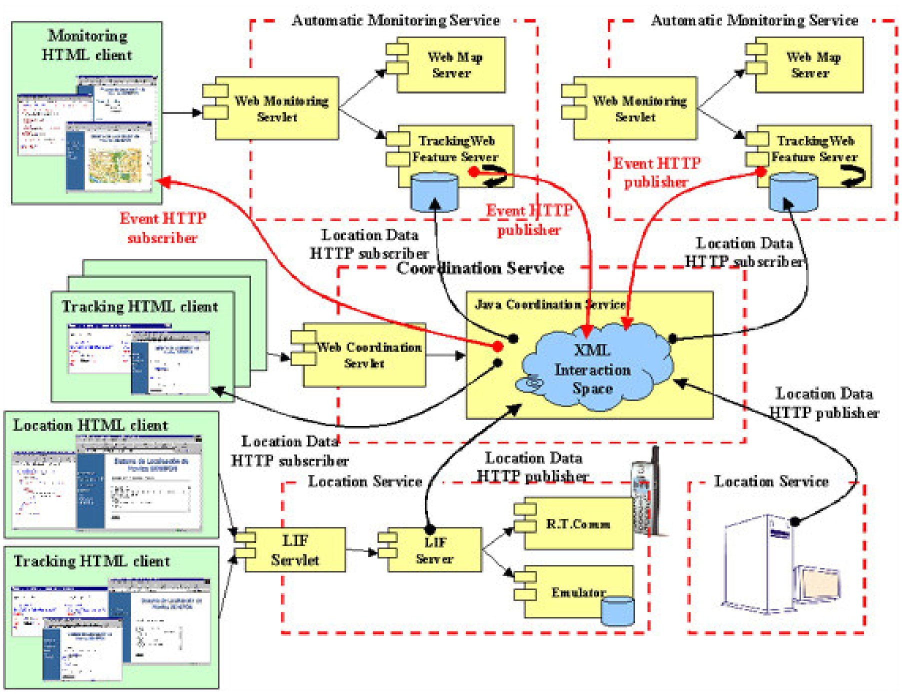

Fig. 3 Services integration new strategy based on a Web-Coordination service

5. CONCLUSIONS AND FUTURE WORK

Web-service architectures are defined using an interaction model among services based on request/response. In this work, an alternative for modeling more dynamic and collaborative interaction environments is presented. The approach proposed has been experimented on the Location-Based Service context. The global aim is to solve the identified coordination and orchestration problems for distributing real-time location data and supplying them to many OpenGIS geoprocessing services, such as Web Feature Servers that store “mobile” features.

It is necessary to provide a global architectural vision to allow the collaboration of different Web services and to exploit their real functional value. Web services must aggregate other Web services to offer a high-level set of features. Applications in the future will be built from Web services that are dynamically selected at runtime based on their cost, quality and availability.

Related research issues are opened: 1) to discover the real potential of the XML language to express synchronization conditions, work flows among services and dynamic chaining of services according to an opportunistic strategy. New prototypical examples will be examined in the context of GIS and LBS services; 2) to define new event-based communication mechanisms more reliable and safer than the HTTP Streaming technique used; and 3) to add new functional components to the Web-Coordination service to support semantic interoperability. For example, it is possible that a provider of location data and a service interested in receiving position data exist. Location and position are two equivalent concepts. Therefore, communication could be established between these services if semantic interoperability is supported.

6. ACKNOWLEDGES

The basic technology of this work has been partially supported by the Spanish Ministry of Science and Technology through the project TIC2000-1568-C03-01, TIC2000-0048-P4-02 and FIT-0700002000B270827 from the National Plan for Scientific Research, Development and Technology Innovation, and by the project P089/2001 from the Aragón Government.

7. REFERENCES

[1] [ABFM02] Amore, M., Bonaccorso, A., Ferrari, F. and Mattia, M.. “EOLO: software for automatic on-line treatment analysis of GPS data for environmental monitoring”. Computer&Geoscience, vol. 28 (2002), pag. 271-280.

[2] [ABMZ02] Álvarez, P., Bañares, J.A., Muro-Medrano, P.R. and Zarazaga, F.J.. “Integration of location based services for Field support in CRM systems”. GeoInformatics, vol. 5 (2002), Nº July/August, pag. 36-39.

[3] [CarrieroGelernter89] Carriero, N. and Gelernter, D.. "Linda in context". Communications of the ACM, vol. 32 (1989), Nº 4, pag. 444-458.

[4] [ESRI00] ESRI. "What Are Location Services? The GIS Perspective". Available in http://www.geojava.com. December, 2000.

[5] [FHA99] Freeman, E., Hupfer, S. and Arnold. K.. "JavaSpaces. Principles, Patterns, and Practice". Addison Wesley. 1999.

[6] [Niedzwiadek00] Niedzwiadek, H.. "All Businesses are in pursuit of Java Location Services". Available in http://www.geojava.com/. January, 2000.

[7] [OGCSA02] OpenGIS Project Document 02-112. "The OpenGIS Abstract Specification. Topic12: OpenGIS Service Architecture (version 4.3)". OpenGIS Consortium Inc. 2002.

[8] [OpenLSTestbed00] OpenLS. "A request for technology. In Support of an Open Location Services (OpenLS) Testbed". OpenGIS Consortium Inc. 2000.

[9] [Thill00] Thill, J.C.. "Geographic information systems for transportation in perspective". Transportation Research Part C: Emerging Technologies, vol. 8 (2000), Nº Issues 1-6, February-December, pag. 3-12.

[10] [ZiliaskopoulosTravis00] Ziliaskopoulos, A.K. and Travis Waller S.. "An Internet-based geographic information system that integrates data,models and users for transportation application". Transportation Research Part C: Emerging Technologies, vol. 8 (2000), Nº Issues 1-6, February-December, pag. 427-444.

[11] [VanderMeer02] VanderMeer, J.. "Ubiquitous Wireless Location Interoperability". Available in http://www.directionsmag.com/. July, 2002.