AN OPEN ARCHITECTURE OF COMMON CORE COMPONENT FOR LOCATION BASED SERVICE

Jae-Chul Kim, Jong-Hyun Park and Jong-Hun Lee
LBS Research Team, Telematics Research Division, ETRI, 161 Kajong-Dong, Yusong-Gu, Taejon, 305-350, Republic of Korea, kimjc@etri.re.kr, Tel: +82-42-860-1523, Fax: +82-42-860-4844

Abstract

In this paper, we developed those LBS Common Core Services based on the Web-Service, because Web-Service environments provide a suitable method to gather requested information in an appropriate way. The proposed architecture cooperate with other OpenLS (Open Location Service) Core Services (Directory Service, Location Utility Service and Router Determination service) and is an interoperability one — it identifies those global elements of the global Web-Services network that are required in order to ensure interoperability between Web-Services. In this paper, a new architecture of LBS Common Core Service with OpenLS Core Services is proposed and tested in OpenLS Core Services environments.

INTRODUCTION

There is a need to further structure large applications into building modules in order to use well-defined components within different business processes. A shift towards a service-oriented approach will not only standardize interaction, but also allows for more flexibility in the process. A service-oriented architecture thus has to focus on how services are described and organized to support their dynamic, automated discovery and use. If the services become more and more complex, the basic mechanism of request-response is hardly applicative anymore. A couple of services include mid or even long term (trans-) actions that demand a functionality to establish an asynchronous communication between a user and the corresponding service, or two services respectively (Lim and Wen, 2003). The Web Notification Service fulfils these needs. Web-Services are self-contained, modular applications that can be described, published, located, and invoked over a network. Web-Services perform encapsulated business functions, ranging from simple request-reply to full business process interactions (Cable et al., 2002).

The proposed architecture is proposed a new LBS Common Core Services based on Web-Service to overcome the platform dependency, closed system characteristics, and distributed computing environment.

WEB-SERVICE SYSTEM

In present, W3C proposed the Web-Service standard as follow: XML (eXtensible Markup Language) is the markup language that underlies most of the specifications used for Web-Services. XML is a generic language that can be used to describe any kind of content in a structured way, separated from its presentation to a specific device (Hunter et al., 2001). In Figure 1, SOAP (Simple Object Access Protocol) is a network, transport, and programming language-neutral protocol that allows a client to call a remote service. The message format...
is XML (W3C, 2003). WSDL (Web-Services Description Language) is an XML-based interface and implementation description language. Figure 2 describes that the service provider uses a WSDL document in order to specify the operations a Web-Service provides, as well as the parameters and data types of these operations (W3C, 2001). A WSDL document also contains the service access information (W3C, 2001). UDDI (Universal Description, Discovery, and Integration) is both a client-side API and a SOAP-based server implementation that can be used to store and retrieve information on service providers and Web-Services (UDDI, 2004).

Also The Java XML Pack includes additional APIs specifically for working with web services. JAXP (Java API for XML Processing) is a document-oriented API; through a "pluggability" layer, it allows any XML-compliant parser to be used from within an application. JAXM (Java API for XML Messaging) facilitates developing programs that produce and consume SOAP messages. It provides methods such as creating SOAP messages and adding contents to the SOAP messages. JAXR (Java API for XML Registries) defines a uniform way of accessing different types of registries. Currently, JAXR supports both the ebXML registry and UDDI registries. It includes functionality for publishing, searching, modifying, and deleting entries in the registry. JAXR also includes sample JAXR clients for browsing well-known registries, including those from Microsoft and IBM. JAX-RPC (Java API for XML-Based RPC) provides an API for building web services and clients using RPCs and XML. Although it uses SOAP for messaging, the application doesn't actually deal with the parts of the SOAP message (as is the case with JAXM).
ARCHITECTURE OF THE DEVELOPED SYSTEM

OGC (Open GIS Consortium) top-level architecture

OGC has formed the Open Location Services Initiative to address service requirements that enable users to tap the rich base of spatial content and promote location application services in emerging wireless Internet markets. The Vision of OpenLS is to deliver open interfaces that enable interoperability and making possible delivery of actionable, multi-purpose, distributed, value-added location application services and content to a wide variety of service points, wherever they might be, on any device.

Figure 3 shows how the concept GeoMobility Server relates to the other elements of a LBS architecture. The GeoMobility server is an element offering basic functions on which location-based applications are built (the OpenLS Core Services). This server uses open interfaces to access network location capacity (provided through a GMLC, for instance) and provides a set of interfaces allowing applications hosted on this server, or on another server, to access the OpenLS Core Services. The GeoMobility Server also provides content such as maps, routes, addresses, points of interest, traffic, etc. It can also access other local content databases via the Internet.

Open LBS common core component architecture

As shown in Figure 4, LBS common core components are connected to LBS platform. And the interface of core service is implemented with Web-Service. The proposed system consists of Web-Service server and Web-Service client developed from EJB (Enterprise Java Beans). A developing tool is WSAD (Websphere Studio Application Developer) 5.0 and WAS (Websphere Application Server) 5.0 is used in web server. The interface of service platform considering interoperability is implemented on the basis of OpenLS (OGC, 2002a, 2002b, 2002c) Spec. v.0.2. Each external CP (Content Provider) gateway communicate with LBS platform to access LBS common core components, also it communicate with its client by using the JSP and Servlet.
Open LBS common core components

Directory service provides subscribers with access to an online directory to find the nearest or a specific place, product or service. Through a suitably equipped OpenLS application, the subscriber starts to formulate the search parameters in the service request, identifying the place, product or service that they seek by entering the name, type, category, keyword, phone number, or some other ‘user-friendly’ identifier. A position must also be employed in the request when the subscriber is seeking the nearest place, product or service, or if they desire a place, product or service at a specific location or within a specific area. The position may be the current Mobile Terminal position, as determined through the Gateway Service, or a remote position determined in some other manner. The directory type may also be specified (e.g. yellow pages, restaurant guide, etc). Given the formulated request, the Directory Service searches the appropriate online directory to fulfill the request, finding the nearest or specific place, product or service, depending on the search criteria. The service returns one or more responses to the query (with locations and complete descriptions of the place, product, or service, depending upon directory content), where the responses are in ranked order based upon the search criteria. The schema of request and response on Directory Service is described in Figure 5.

Location-Utility service performs as a geocoder by determining a geographic position, given a place name, street address or postal code. It also returns a complete, normalized description of the place (which is useful, say, when only partial information is known). The service also performs as a reverse geocoder by determining a complete, normalized place name/street address/postal code, given a geographic position. Both the geocoder and
reverse geocoder may return zero, one, or more responses to a service request, depending on subscriber request information, the algorithm being employed, and the match criteria. The schema of request and response on Location Utility Service is described in Figure 6.

![Figure 6: Schema of Location Utility Service. (a) request; (b) response.](image)

Presentation service renders geographic information for display on a Mobile Terminal. Any OpenLS Application may call upon this service to obtain a map of a desired area, with or without map overlays that depict one or more OpenLS ADTs, such as Route Geometry, Point of Interest, Area of Interest, Location, Position and/or Address. The service may also be employed to render route directions from Route Maneuver List ADT and/or Route Instructions List ADT. The schema of request and response on Presentation Service is described in Figure 7.

![Figure 7: Schema of Presentation Service. (a) request; (b) response.](image)

Route service determines a route for a subscriber. The subscriber must use a navigation application to set up the use of the service. They must indicate the start point (usually the position acquired through the Gateway Service, but this could be a planned trip from a specified location, say, from their home), and the endpoint (any location, like a place for which they only have the phone number or an address, or a place acquired through a search to a Directory Service). The subscriber may optionally specify waypoints, in some manner, the route preference (fastest, shortest, least traffic, most scenic, etc.), and the preferred mode of transport. The subscriber may optionally store a route for as long as needed, thus requiring the means to also fetch a stored route. The schema of request and response on Route Service is described in Figure 8.

![Figure 8: Schema of Route Service. (a) request; (b) response.](image)

In consequence, the published interface of processing module is made in a WDSL document, and the service client makes the proxy interface on the basis of the published WDSL document. Currently, Web-Service developed between the two opposing sides (IBM and Microsoft) has a problem that its interoperability can not be applied.
We need a spatial operation to execute the Directory Service that searches for nearest POIs, within distance, within boundary. In this system, Oracle 9i (SDO) is used and it assists with Directory Service spatial operation to search the interest point.

**System architecture based on EJB component**

The Enterprise JavaBeans architecture is component architecture for the development and deployment of component-based distributed business applications. Applications written using the Enterprise JavaBeans architecture is scalable, transactional, and multi-user secure (Sarang et al., 2001). These applications may be written once, and then deployed on any server platform that supports the Enterprise JavaBeans specification. In brief, EJB is designed to address issues involved with managing distributed business objects in three-tier architecture. The system architecture developed from Java Bean cannot be reused on distributed environment and other component module. For this reason, the proposed architecture has an EJB structure.

**LBS COMMON CORE COMPONENT CLIENT**

The developed client is tested on wired and wireless environments. And the platforms in client are PDA (Microsoft WindowsCE, iPAQ3660, Arm SA1110, 644MB) and WIPi mobile phone (WIPi, SamSung X9300, Arm9, 4MB). In Figure 9 and 10, the simulation result (WinCE and WIPi) of LBS Common Core Component was presented.
An open architecture of common core component for location based service

CONCLUSION

In this paper, LBS common core Web-Service based on EJB is proposed to overcome a platform dependency and to enhance the distributed computing performance. The proposed common core services are independent of a server platform. And due to the Web-Service system architecture, client is not limited to program languages. However Java based Web-Service is a little slower than .NET Web-Service in test-bed. More simulation is need in parallel computing environment and multi-user request.

REFERENCES