An Information Dissemination Framework for Location Based Services

Preeti Saxena¹ and Raj Kamal²

School of Computer Science and Information Technology, DAVV, Indore

Abstract: Spatial information service such as, location based services (LBS) are popular these days. They are large scale information dissemination applications that provide location dependent information to a large number of service user’s interested in them. Publish-subscribe approach is one of the approaches accepted in implementing an effective LBS application for dissemination of location information. The LBS environment is built around several network technologies, database technologies, platforms, positioning system technologies. It requires processing and disseminating interested piece of information for millions of subscribers efficiently in such a heterogeneous distributed environment. A suitable framework is presented in this paper for Messaging-based LBS applications using XML-based selective dissemination of information system exploiting publish-subscribe approach. It is proposed to speed up the filtering process for its subscriber provided their subscription (interest) in the form of XPath queries. The proposed framework presents the methods for delivering efficient and fast location services to its service user through a common interface. It suggests a scalable distributed platform to manage large number of users (subscribers) and a large number of service providers (publishers).

Keywords: Location based services, publish-subscribe, XML filtering, XPath expressions

1. INTRODUCTION

The emergence of mobile devices, wireless and mobile computing technology opened new opportunities in the field of information services. They provide information to a large number of service users interested in them. Information dissemination system is a popular way to distribute information received from several service providers to multiple users. It delivers enjoined information subscribed by the users [1,2,3]. Publish-subscribe paradigm models information dissemination system in a suitable manner. The service providers are publishers of the interested piece of information for millions of users as subscribers of the information [1,4]. The main component of publish-subscribe system consists of an efficient filtering component [2,5]. Filter decides the content for each subscriber. Subscription (interest expression) is in the form of user profile. Location based services (LBS) is accepted as the example of publish-subscribe based information dissemination system [1]. The development of LBS application requires inclusion of varied technologies. Several network technologies, database technologies, platforms and positioning system technologies are integrated to build a common infrastructure for LBS application [6,7]. The framework is proposed for an LBS application when XML-based SDI system exploits publish-subscribe approach. The proposed framework speeds up the filtering process for its subscriber. The framework outlines the challenges and research directions towards building an application that provides a reusable LBS middleware component. One of the challenges is processing of large amount of service provider’s content and issuing notifications for millions of subscribers continuously and concurrently in heterogeneous distributed environment [7,8]. Another challenge is to build a common interface to user for location data that hides the working of positioning system at background [9]. Therefore, the application needs a simple, scalable and portable platform to manage large number of users and large number of service providers in heterogeneous distributed environment. The objective of this framework is to develop methods for delivering efficient and fast location services to a service user by means of data integration and abstraction through a common interface.

A scalable and distributed framework for SDI systems is thus proposed. Following sections describe components of the framework proposed for LBS applications and discusses its implementation issues. Section 2 describes the architectural requirements and the architecture of proposed framework. Section 3 describes the implementation of the proposed framework. Section 4 presents conclusions and scope for further work.

2. ARCHITECTURAL REQUIREMENTS

The presented approach modifies the concept of simple request-response to publish-subscribe paradigm in order to build a large scale SDI system. The publishers are hosting voluminous amount of data on server related to user’s interests. Subscribers are interested in receiving information notifications about their interest from potentially large service providers. They send a subscription to the corresponding application on the server. All these subscriptions are processed immediately by the middleware component as soon as the streaming data containing the desired information arrives [1]. The subscriber whose subscription matches the streaming document receives the information notification along with the interesting part of the document. Thus a matching
component responsible for comparing the interested piece of information to the incoming streaming document serves the core of an efficient publish-subscribe system. An information dissemination system for LBS is realized using XML as information transferring format between publishers and subscribers, and XPath [10] expressions as a format for specifying the subscriber’s interest (user profile). The system leads to fast and efficient filtering of interested location information [11]. Figure 1 shows the general architecture of publish-subscribe system for an LBS application. It requires user’s subscriptions that are stored on the server in the form of XPath expressions. The DTD [12] or XML schema of LBS services to be present on service provider’s server in advance. The expressions are matched with the incoming document stream. An information notification consisting of the interested piece of information is sent to the subscriber thereafter. The approach uses the schema information of the LBS to pre-filter the registered user’s interests in the form of XPath expressions. It results in optimization at pre-processing level.

The proposed framework provides a common platform for XPath filtering on the data collected from various service providers in form of XML streaming documents [2]. It is provided in form of a middleware component. The middleware selects and optimizes the XPath expressions based on pre-filters prepared using registered XML document schema of service providers. It then converts the selected set of XPath expressions into desired format required for input to the filtering system. An indexing mechanism is prepared to execute XPath query expressions on the streaming XML documents for efficient filtering. When a subscriber registers, its XPath expression is added to the subscription database. The middleware is also responsible for distribution of XML document fragments to these subscribers.

The next section discusses the implementation of the proposed framework and discusses the core issues raised in the implementation.

### 3. IMPLEMENTATION

A component based framework is suggested to provide concurrent and continuous services to the users of LBS. Java EE technologies is proposed to build the framework. It supports a multi-tiered architecture for implementing component based enterprise applications. The tiers separate the functionality of the overall application. The architecture of the framework is three-tier with EJB’s (Enterprise Java Beans) as an integral Part of the server-side component. The component is a self-contained functional software unit that communicates with each other in the application. The three-tier architecture separates the presentation logic, the business logic and the data schema. The advantages of this scheme are [13]:

- **Easy updation:** change in one tier does not affect the implementation of rest of the tiers.
- **Less deployment and maintenance cost.**
- **Provides higher flexibility and extensibility to the application.**

The main components of the proposed LBS application framework are:

- Application clients running on user device.
- Servlet and JSP that run on web server.
- EJB components (reusable distributed components) that run on the application server.
- Persistent data storage.

The application client provides the user interface to interact the service provider’s application. User interface is a key component of the application that makes it possible for subscribers to manage the content that appears on their device [13]. The application clients identified in the proposed LBS application framework are request client, positioning client and display client. Request client lets user to select different service. It enables users to subscribe to location information channel and also supply a list of keywords that will be matched against the required point-of-interest (POI) or information search. Positioning client provides current location of the user to the server and display client views a map and other texts on the client device.

JSP components at the middle-tier (web-tier) are used for presentation layout, custom tags and JavaBeans components for presentation logic and Servlets for dispatching. EJB provide another middle-tier component that run at application server. It is used for developing scalable, extensible, flexible, secure and consistent distributed server-side components.

The conception of Java based middle-tier leads to solution for building an LBS application framework for heterogeneous distributed environment. It is one of the core issue discussed in building such framework. The third-tier is the database side. It handles management of spatial data, POI data and non-spatial data such as subscriber’s information. Map data, route information, address geocode sources, information related to locations are stored using any spatial database technology.
Entity EJB can connect to the database through some interfaces, such as ODBC, JDBC [13]. POI and subscriber information are stored in XML database. Figure 2. shows the proposed architecture of the LBS framework.

![Figure 2](image)

**Figure 2** Three-tier LBS Architecture with Java EE Technology.

### 3.1 LBS middle-tier modules

LBS middle-tier modules are described in the following subsection.

#### 3.1.1 Subscriber Module

A subscriber initiates the location-based service application by entering a service request through client interface in form of subscription profile or user profile. The subscriber module handles these user profiles. They also manage the various operations such as updating and deleting of user profiles. Therefore, it is an application specific module.

The user profile information may include various personal or POI interest data for the user. Personal data, for example, include financial, age, context or any other information useful in personalizing LBS. Such user profile information can be used to find service information notification to a service request (for example, the nearest restaurant for financial and interest criteria stored in the user’s profile) or in otherwise pushing a service request (for example, sale offers from shops depending on the current location of the customer) [14]. The user profile information is therefore used in combination with the location information to personalize LBS.

The following DTD is used for implementing the service provider’s database and LBS user’s queries.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT position (Point)>  
<!ELEMENT posList (#PCDATA)> 
<!ELEMENT name (#PCDATA)> 
<!ELEMENT height (#PCDATA)> 
<!ELEMENT Start (posList)>  
<!ELEMENT Point (posList)>  
<!ELEMENT POIapplication (City+)> 
<!ATTLIST Monument id (m001 | m002 | m003) #REQUIRED >  
<!ELEMENT Monument (name, position)>  
<!ATTLIST City id (c001 | c002 | c003) #REQUIRED >  
<!ELEMENT City (name, Building+, Monument*)>  
<!ATTLIST Building id (b001 | b002 | b003 | b004 | b005 | b006 | b007 | b008) #REQUIRED >  
<!ELEMENT Building (name, height, position)>  
<!ATTLIST City id (c001 | c002 | c003) #REQUIRED >  
<!ELEMENT City (name, Building+, Monument*)>  
<!ATTLIST Building id (b001 | b002 | b003 | b004 | b005 | b006 | b007 | b008) #REQUIRED >  
```

The performance is evaluated for 10,000 of XPath query expressions. Some queries are given in Table 1.

The subscriber module handles the user profiles in XPath format and transfers them to pre-filter module for processing them according to the constraints defined by application.

#### Table 1: Sample LBS Query Expressions

<table>
<thead>
<tr>
<th>XPath Expressions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/city/*/Position</td>
<td>selects all the position elements that are grandchildren of city</td>
</tr>
<tr>
<td>/**/name</td>
<td>selects all name elements which have two ancestors</td>
</tr>
<tr>
<td>/@</td>
<td>selects all elements in the document</td>
</tr>
<tr>
<td>/city/posList</td>
<td>selects all the posList elements of all the descendant of the city element</td>
</tr>
</tbody>
</table>

#### 3.1.2 Pre-filter Module

The pre-filter module minimizes the query nodes to save the processing time or to eliminate any query that doesn’t suit the policy designed by service provider. Reduction in number of queries input speeds up the overall filtering process. The concept uses the static optimization and the dynamic optimization when DTD [12] or XML schema is already available.

#### 3.1.3 Filter Module

The selected XPath expressions are then converted into input sequences [11] that are used to represent queries for further processing. Value-based sequence is a sequence of nodes carrying data content in the form of a set of triplet of open-value-close group.

The numbers of XPath expressions that are required to be matched against streaming XML documents are very large. This requires an efficient indexing strategy for
efficient and scalable filtering. Indexing on sequences of XML nodes reduce the number of searches thereby increasing the efficiency of filtering. A sequence-based algorithm, PFilter [11], is used for XML filtering. It is designed to provide efficient and fast search in the streaming document in single parse. It supports structure matching as well as content filtering in the document. It is scalable and has shown its efficiency in terms of filtering time also. The module solves another core issue related to LBS application framework namely, scalability.

### 3.1.4 Positioning Module

All the Location based applications are directly related to the positioning systems. Determining a mobile user's current location is one of the most important functions of mobile computing environment. The positioning module is responsible for determining the position coordinate of the user. Various applications use different types of positioning systems as and when required. It can be obtained either by using the mobile communication network or by using the Global Positioning System (GPS). The position determination is one of the key requirements of the LBS. If the position is not determined automatically it can be also specified manually by the user.

The framework suggests a flexible Positioning Module, implemented as session EJB, so that the service developer need not worry about position capturing and converting coordinates into different coordinate systems. A developer can here concentrate on the actual service function and has not to deal with sensors and low-level protocols to determine the location.

Position coordinates obtained through this procedure are another input to XML filter. The PFilter module works on location rather than coordinates. The filter matches the POI based on the position and interest of the user. It is first required to be geocoded and provided to PFilter.

### 3.1.5 Geocoding and Reverse Geocoding Module

Geocoding service performs the mapping of POI location with geographical coordinates of the place. It is the process of assigning locations to addresses so that they can be placed as points on a map and analysed with other spatial data [15]. POI data are the geo-objects the user is looking for. The geocoding assigns geographic coordinate to the POI to display it on the map. The spatial GIS data, POI database and geocoding form the basic components to build a custom LBS application [16].

Here, POI data is stored in form of XML files. That simplifies the supply of POI data to different servers in form of streaming documents. The position coordinates are obtained from positioning module. The PFilter extracts the POI data of user’s interest and position. It then forwards it to the geocoding module. The major role of this module is to integrate the geo data with GIS data and provides the format suitable for map display. The formats such as GPS Logs (NMEA), Google Earth (KML) and OpenStreetMaps (OSM) are the examples popularly used in GIS applications [9].

The inverse process is followed in reverse geocoding [15]. Geocoding converts human readable address form into latitude and longitude, while the reverse geocoding converts the latitude and longitude into human readable form.

#### 3.1.6 KML Converter

The visualization of geographic information is one of the important functions of LBS system [6]. Once the desired POI is extracted and its geocode is available with respect to the physical coordinates as needed by the mapping system. It is now required to convert it in the form suitable for display. The XML schema used in the proposed framework is not a visualization format. It is platform neutral and is well suited to facilitate the exchange of spatial data over the Internet. Mapping systems commonly encode spatial data with XML or XML based language [17,18], for example, GML [19], for transmission and with a visualization format, for example, Scalable Vector Graphics (SVG) [17,19] or Keyhole Markup Language (KML) [20,21] for display.

KML encodes the shapes, locations and visual characteristics of POI. KML is widely used to provide output from a GIS into Google Earth [21]. The present work proposes the use of KML to display the output. Overall KML can be used to:

- denote locations on the surface of the Earth using latitude and longitude, and to describe geometries.
- specify symbols and descriptions at these locations.
- position images on the surface of the Earth using latitude and longitude

Thus, a high-performance XML to KML translation is required for large volumes of spatial data for mapping, and their visualization. An XML transformation technology, XSLT [22], is one alternative designed for XML to XML document transformation. The XSLT 3.0 specification is designed to allow XML streaming. It is important when processing document that is large to fit in memory. Internally XSLT uses XPath to identify subsets of the source document tree and to perform calculations [23].

XML to KML document transformation requires the extraction and translation of source XML encoded features to equivalent destination KML features. A basic understanding of the KML primitives is necessary to perform this translation. The POI’s geographical coordinates are transformed into KML to correctly display the POI pictograms. The user receives the image in the form of a KML encoded picture, which contains the map and symbols.

The DTD defined in Section 3.1.1 is used to generate documents that model city feature collection. XML to KML data translations are executed on proposed DTD to
model the sample city information.
An XSLT transforms the contents to the format required for display. The following XSLT stylesheet fragment presented below illustrates the logic necessary to translate source XML to KML.

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <kml xmlns="http://earth.google.com/kml/2.0">
      <Folder>
        <name>My Places</name>
        <xsl:for-each select="POIapplication/City/Building">
          <Placemark>
            <name><xsl:value-of select="name"/></name>
            <Point>
              <coordinates>
                <xsl:value-of select="position/Point/posList"/>
              </coordinates>
            </Point>
            </Placemark>
        </xsl:for-each>
      </Folder>
    </kml>
  </xsl:template>
</xsl:stylesheet>
```

The resultant KML code is provided to dissemination module for providing map service to display client.

### 3.1.7 Dissemination Module

LBS applications rely on information dissemination. Subscribers browse the POI or traffic updates on a WAP-enabled mobile phone or a desktop browser [6]. The web interface interprets the incoming requests, processes the filter, geocode the location, converts it into the display format and submits the request to the dissemination component. A KML file can be enabled for the map service when it is prepared. It automatically gets enabled for web access once the KML service is published. It is then accessible via KML client applications (display client), such as ArcGIS Explorer, ArcGlobe, or Google Earth [24].

The dissemination component is also responsible for subscriptions and other non-graphical, map-independent notifications. Such information dissemination is handled using messaging system. It interacts with other components that perform the actual delivery, such as publish-subscribe component, messaging component (email or SMS) [6]. An information dissemination module acts as a mediator between publishers and subscribers and disseminates the content from a single source to a potentially large number of destinations. It enables user privacy, thus publishers are unaware of subscriber’s identities as well. Furthermore, publishers and subscribers do not require connection-oriented service. They need not be active simultaneously to exchange notifications. Here the communication is asynchronous which is important in mobile environments where users are frequently disconnected or unavailable [1].

The communication is also classified as stateful, if the application maintains state across multiple service requests and as stateless, if no such state is maintained [1,13]. Each request is processed independently of past requests in stateless communication service. For example, in tracking applications, it is important to keep track of previous requested location to take decision thus it is stateful while a POI request is stateless.

The dissemination module is implemented as a distributed component. It interacts with a Java Servlet that runs within a web server and supports both HTTP and WAP [13]. The stateful as well as stateless java beans components are proposed to handle the requests based on application requirement. Publish-subscribe model discussed is inherently stateless.

The dissemination module provides publish-subscribe based distribution of notifications to subscribers [6]. It offers a standardized interface. The actual delivery functionality is provided by specialized components, Java Messaging Service (JMS) [25] server and messaging component.

These modules are defined to build a framework for delivering efficient and fast location services to a subscriber of LBS application through a common interface.

### 4. Conclusion

The challenge handled by an LBS application is processing of voluminous amount of service provider’s content and issuing notifications for millions of subscribers continuously and concurrently in heterogeneous distributed environment. A distributed framework is suggested as a concrete solution for it. The components of the proposed framework were described in the paper. It was offered for LBS applications using XML-based SDI system using publish-subscribe approach. The proposed framework presented a scalable and flexible distributed platform to manage large number of users (subscribers) and large number of service providers. It is equipped with the necessary modules that support the fundamental features of LBS servers, for example, continuous and concurrent queries and heterogeneous environment. Implementation of an application based on this proposed framework has been tested. Using the proposed framework the LBS applications can be developed in less time and less efforts.
The framework can be further extended to handle context-based and preference-based information dissemination in future.

References


AUTHOR

Preeti Saxena received the MCA degree with distinction in 1996 from MACT (Regional Engineering College), Bhopal and M.Tech.(CS), degree in 2008 from Devi Ahilya University, Indore. She is currently pursuing Ph.D. (Computer Science) under the guidance of Dr. Raj Kamal at Devi Ahilya University, Indore. Her areas of interests are computer networks, mobile and wireless systems, web technology and network management. She has authored/co-authored several research publications in national and International conference proceedings.

Dr. Raj Kamal, Ph.D. (1972) IIT Delhi, Senior-most Professor, School of Computer Science and Information Technology, Devi Ahilya University. He has 45 years of experience in research and teaching since the age of 18. He has successfully guided 12 research students and is guiding six Ph.D. students in Mobile Computing and Embedded Systems.