CATIS: A Context-Aware Tourist Information System

Ariel Pashtan[1], Remy Blattler[2], Andi Heusser[2], Peter Scheuermann[2]

*[1]Aware Networks
Buffalo Grove
IL 60089, USA*

*[2]Dept. of ECE,
Northwestern University,
Evanston, IL 60208, USA*

ariel@awarenetworks.com, remy.blaettler@gmx.ch, aheusser@gmx.ch, peters@ece.nwu.edu

Abstract

In this paper we describe our current implementation and future plans for CATIS, a context-aware tourist information system that leverages Web services and XML technologies for its implementation. We review notions of context as they relate to tourists, and provide relevant tourism scenarios that helped drive our design. Our system architecture is Web services-based and includes a context manager element that manages both dynamic and static context. The elements of context in our work are location, time of day, speed, direction of travel, personal preferences, and device type. We describe how these elements are leveraged to adapt Web-based information that is delivered to mobile tourists. Our future plans for incorporating mobility in the user's context, and how to scale our system to support large numbers of mobile users are described as well.

1. Introduction

Context-awareness seeks to acquire and utilize information about the physical and social situation in which users and their wireless handset devices are embedded to provide improved services that are appropriate to the particular user given the interaction platform, place, time, environment, and surrounding events. Wireless handset devices, physical and social contextual-information, all correlated with the infrastructure back-end information can truly enhance wireless services in terms of relevance, quality, and appropriateness of delivered information. Example envisioned services include tourist information services that will wirelessly deliver relevant city information to tourists' mobile devices.

In the following we describe our current implementation and future plans for CATIS, a context-aware tourist information system that leverages Web services and XML technologies for its implementation. We start by reviewing notions of context as they relate to tourists, and provide relevant tourism scenarios that helped drive our design. We then describe our Web services-based system architecture and introduce the concept of a context manager. A description of how context is leveraged to adapt information services follows. The relevant context elements are location, time of day, speed, direction of travel, personal preferences, and device type. We end by describing the current status of our project and our plans for incorporating mobility in the user's context, and how to scale our system to support large numbers of mobile users.

2. Motivation

Context has been classified in different ways by various researchers in this field (e.g., [5]). In general, context refers to information about the device platform, the user, and the surrounding environment. User context typically refers to the user's personal preferences for information content, and environment context includes the user's location and current time of day.

Context has been used in distributed event-based applications for wide-area networks. These applications are characterized by a publish/subscribe model [4], [6], [11], where receivers subscribe to information of interest to them without regard to any specific source, while senders publish information without addressing it to any specific destination.

Context-awareness is very critical for mobile users. These users have at their disposal devices that are not very conducive to interactivity (e.g., small screens, space constrained keyboards). Context-awareness can help improve user interaction by knowing a-priori the user's situation, personal preferences, information interests, and environment conditions, so that the user doesn't have to specify these constraints, and information delivery is automatically adapted to his circumstances. For example, if a context-aware system knows that a user is driving a car, it can adapt alerts to audio messages without requiring the user to input details about his current situation.

Context-aware services will also enhance the wireless Internet experience in a significant way. Mobile users that

To appear in Proceedings of the 4th International Workshop of Mobile Computing, Rostock, June 2003

want to access Web-based content will expect these information services to exhibit adaptation capabilities [12]. The adaptation we have addressed in our work includes:

1. Location and time-based adaptation: the system will track users' location and will provide them with information relevant to their locale and time of day (e.g., open restaurants in their respective direction of travel).
2. Personal adaptation: users will have access to services that are tailored to their specific profile (e.g., users want to receive information about resources that match their personal tastes such as nearby restaurants that offer their preferred cuisine, or want to be notified of certain events if they occur in their vicinity, or want the system to generate travel directions customized to their preferences).
3. Device adaptation: knowing the device features, the delivered information can be tailored to the particular hardware and software platform. For example, the device has a certain screen size and may not support certain image formats.

Tourism is an area that will benefit from context-awareness as demonstrated by research projects in mobile context-awareness [1],[7]. The FIPA (Foundation for Intelligent Physical Agents) standards organization [10] and m-ToGuide [9], a project sponsored by the European Commission's 5th Framework Program, are also targeting tourism applications as a new service for mobile users where context will play a significant role in the delivery of relevant information to tourists on the go.

Tourist scenarios provided the motivation for the CATIS system design. In the following we describe scenarios that take into account user location, time of day, speed of travel, and direction of travel.

3. Tourist Service Scenarios

In the scenarios we considered, the tourist can specify categories of information interests, and the tourist's wireless device will display related location-based information as he moves between city locations. For example, the wireless device can display landmark information. As the user changes locations, the displayed information updates to the new locality. Besides location, time of day can also trigger a change in the presented information. Speed and direction of travel also impacts the displayed information. Finally, as the tourist's situation changes, for example first driving then walking, the displayed information will adapt to different presentation formats, for example audio then graphics and text.

In our first implementation phase we took into account user location and time of day. The scenario is as follows:

It is noon and Sue is in the Chicago metropolitan area. Based upon the time context, her wireless device automatically initiates a request to get restaurant information for eating lunch. The network-based tourist information service is queried for restaurants in the immediate neighborhood of Sue. It subjects the answers to a sequence of filter, transform, and sort steps in order to take into account the user preferences for restaurants and the type of device that she is using. For example, pizzerias appear at the top of the list because they are Sue's favorite, and Chinese restaurants appear second as her alternative choice. Restaurants with cuisines that Sue doesn't like, or that are too expensive, are filtered out. Finally, the resulting restaurant list is transformed into Web pages in a format appropriate for display on her client wireless device.

Our second implementation phase adds speed of travel and direction of travel context considerations. The scenario is as follows:

It is evening and Guy is in the New-York metropolitan area. Guy is traveling in a rented car. Based upon the time context, his wireless device automatically initiates a request to get restaurant information for eating dinner. The network-based tourist information service is queried for restaurants in the neighborhood of Guy and in his detected direction of travel. In addition to considering the user preferences for restaurants and the type of device that he is using, the tourist information service tracks his speed and direction of travel. Restaurants that are within a predetermined range of his location, for example up to half an hour of driving, and in his detected general direction of travel, will be candidates for display. Finally, the resulting restaurant list is transformed into Web pages in a format appropriate for display on his wireless device. Figure 1 depicts how time and location context affects the display of tourist information: upon moving from one geographic area to the next (the different circles) the displayed information content changes in relation to the context.

In the following, we describe the development of CATIS, a context-aware information system for mobile tourists that implements a Web services-based system architecture and leverages XML technologies for its adaptation capabilities.

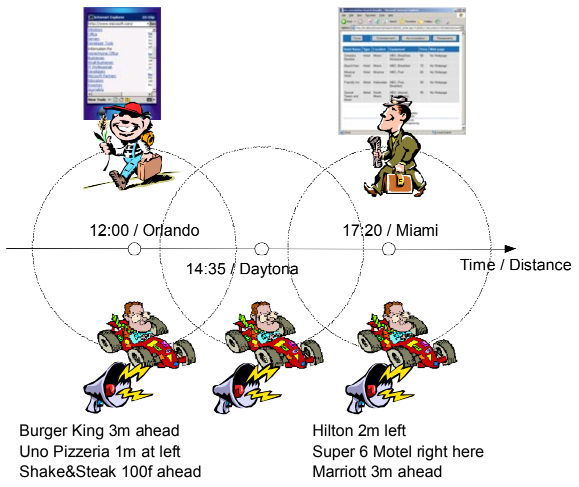


Figure 1 Time and location context affect

4. CATIS Architecture

The CATIS architecture [3] that supports a tourist information service is shown in Figure 2. This architecture is Web service-based and the major elements include:

- A thin client device that hosts a Web browser.
- An application server that delivers web content customized to the user's context.
- A UDDI (Universal Description, Discovery and Integration) services directory that provides users with a centralized registry of tourist information services (e.g., a restaurant finder service). The UDDI specifications describe the information to provide for each service, as well as provide a query and update API to access information in the registry. For the UDDI implementation we used Systinet's WASP UDDI Enterprise Server.
- A context manager that keeps track of the user's dynamic context (e.g., location, wireless device features) as well as the user's preferences.
- A collection of Web services that deliver tourist content (e.g., landmark information, restaurant locations, etc). Each Web service has a WSDL (Web Services Description Language) document in XML format that describes the Web service's interface and gives a concrete binding to a network address.

Given the Web's intrinsically distributed and heterogeneous nature, communication mechanisms must be platform-independent and as lightweight as possible. SOAP (Simple Object Access Protocol) [8] is used for the communication between the application server and the directory service (UDDI) as well as between the application server and the various Web services. SOAP is an XML-based protocol for messaging and remote procedure call and is used here on top of HTTP.

When a new user registers himself in the system, he logs on to the application server and enters his preferences (e.g., restaurant preferences). These preferences are then forwarded for storage to the context manager. If the client device possesses a GPS receiver then the client sends location updates to the context manager at regular intervals. User speed and direction is computed from the user's tracked location history. If the user doesn't have a GPS receiver then at the point of the information inquiry he enters his location manually in the form of a city name or zip code.

In our implementation we used both Java Web services and Microsoft's .NET Framework, a native XML Web services platform. Microsoft .NET includes ASP.NET, a framework for creating application servers that support dynamic web pages, standard Web service technologies like SOAP and WSDL, as well as a multi-language development environment (including C#). One of our Web services was implemented in C#.

When a tourist needs information, for example about restaurants in his vicinity, he connects to the application server to request the information (step 1 in Figure 2). The application server queries the context manager for user context information such as location and restaurant preferences (step 2). It then sends an inquiry to the UDDI Server to get the addresses of the available restaurant Web services (step 3). The application server then sends a request to all the Web services along with the user's location and desired distance from the user (step 4). The Web services search their databases for the appropriate addresses and filter out those that are too far away. The Web services return an XML list to the application server. The application server filters the XML documents according to the user's preferences, prepares the presentation of the information and sends it to the client (step 5).

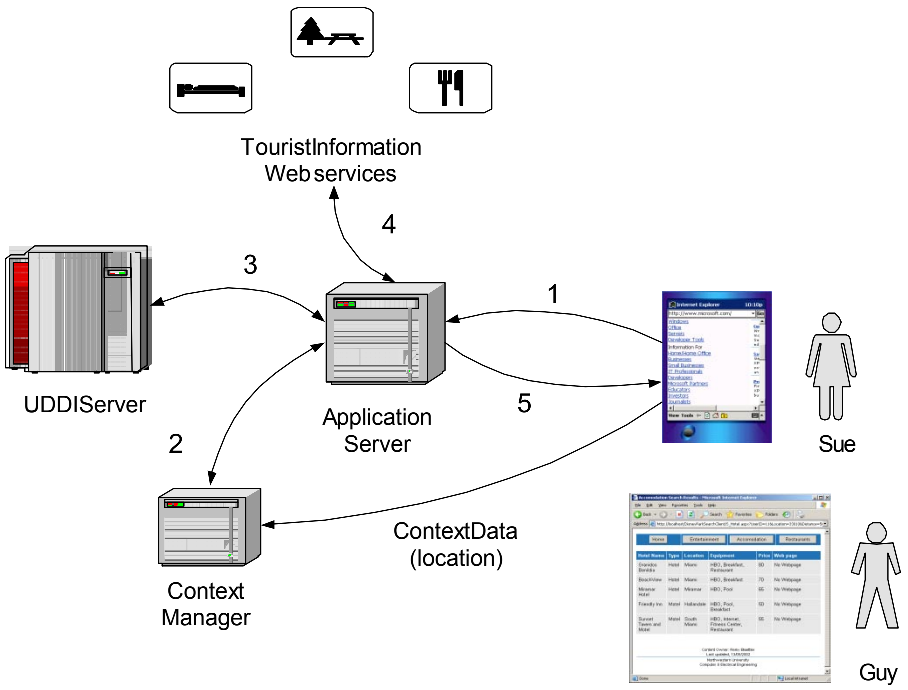


Figure 2 CATIS Architecture

5. Location and Time-Based Adaptation

As tourists travel in a city, they come across different landmarks and have different needs depending on time of day. Tourists want to learn about the landmarks they are facing, or may want to find a nearby restaurant that serves their favorite food if it is dinner time. While location is a prime determinant of what information should be sent to the user's wireless device, time can affect the suggested information content that is delivered. For example, if it is lunch time and a tourist expressed interest in receiving a list of restaurants in close proximity, the delivered list should not contain those restaurants that serve food starting at a later time of day. The application server issues a request to the restaurant Web service and the Web service returns a list of restaurants filtered based on location and time of day.

In addition, tourists may be traveling by car or public transportation, and the information delivered to their device needs to be updated to reflect their changing environment. Additional elements of context that affect the displayed information include speed and direction of travel. We devised a sliding grid-window methodology in which location, speed, and direction of travel are taken into account to deliver a list of restaurants (Figure 3). Speed of travel determines whether further away restaurants should appear on the list, and direction will narrow the list to those in the direction of travel. For example, restaurants that are in the opposite course of travel will not be listed.

Our sliding grid window algorithm uses a two-step approach whereby first the relevant tourist's neighboring area is determined, and then a list is generated based on a preferred maximum distance of travel to a restaurant. This approach relies on detecting the user's speed and leveraging a pre-partition of the metropolitan area into a grid of square areas where the user is traveling as shown in Figure 3. The user's GPS instrumented client device sends periodic location updates to the CATIS context manager (Figure 2). From each location update the CATIS application can determine where the user is currently located. By tracking a user's location changes, the application can determine the user's speed and direction of movement. From these latter two elements of context, the application selects a set of squares that form the area that surrounds the user. The faster the user travels, the greater is the area.

Once the surrounding area is determined, an information request is issued to the restaurant Web service to retrieve the area's restaurants. By requesting only the restaurants within a certain sliding grid window we are allowing for the possibility that Web service could make use of spatial data structures [13] in order to find the relevant data set. When this latter set is returned to the CATIS application, a filtering step takes place to select only those restaurants that are within a pre-determined distance from the tourist. For example, only those restaurants that are within half an hour of travel, at the current user's speed, will be selected for display. This algorithm proves to be more efficient compared to an exhaustive computation of distance over all restaurants, and is also more attuned to take into account the user's direction of travel.

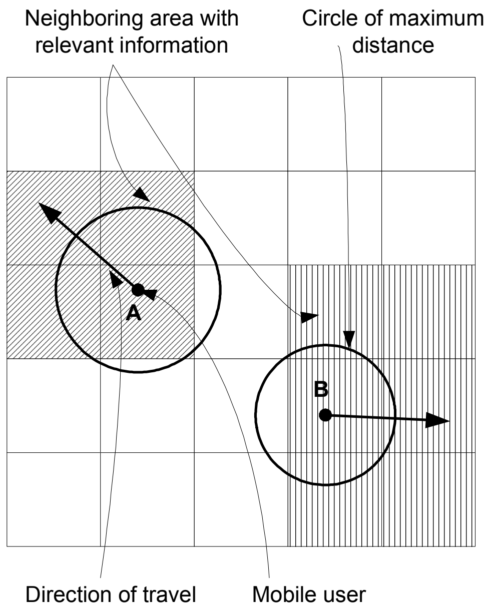

Neighboring area with relevant information
Circle of maximum distance
A
B
Direction of travel
Mobile user

Figure 3 Selection of relevant area based on speed and direction of travel

6. Personal Adaptation

An important aspect of context-awareness is the system's ability to deliver information that is personalized, i.e., adapted to a user's specific needs. The personalization aspect includes system awareness of the user's information interests and service preferences. Personalization has been deemed by the Wireless World Research Forum as an important determinant in context-aware computing where the human is placed at the center and communication systems are built based on the analysis of individual communication spaces and provide for individual adaptation [2]. The adaptation performed in CATIS is done based on user context information, including his preferences, retrieved from the context manager. A tailored list of restaurants that meets the user preferences is generated as shown in Figure 4.

Upon receiving the list of restaurants from the Web services, the application server can modify directly this list, and then apply selection criteria to extract from the list only restaurants that meet user pre-specified preferences. Initially, the list of restaurants is modified to include preference levels as indicated in user profiles, and distance from the user's current location. The preference levels specify the user cuisine preferences. The result of this step is a modified list with *Restaurant* elements that include *PreferenceLevel* and *Distance* sub-elements (bold entries) as shown in Listing 1.

```
<Restaurant>
   <Name>Blue Ribbon Restaurant</Name>
   <Street>6301 N Ridge Ave</Street>
   <City>Chicago</City>
   <State>IL</State>
   <Zip>60660</Zip>
   <Phone>(773) 973-4825</Phone>
   <Cuisine>Southern/Soul</Cuisine>
   <PriceHigh>35</PriceHigh>
   <PriceLow>10</PriceLow>
   <Latitude>41.995796</Latitude>
   <Longitude>-87.675693</Longitude>
   <Open>10:00</Open>
   <Close>23:30</Close>
   <Valet>0</Valet>
   <Distance>2.6015</Distance>
   <PreferenceLevel>3</PreferenceLevel>
</Restaurant>
```

Listing 1 Restaurant element modified with preference

The extraction of specific restaurant elements that meet user criteria is done by modifying the XSLT stylesheet (partially shown in Listing 2) that is used to convert the list of restaurants to displayable markup (XSLT is further described in the next section). The *for-each* statement in Listing 2 goes through the list of restaurants and selects only those that fulfill the criteria specified in the *select* attribute. The next two *sort* statements tell the XSL transformer how to sort the list. The remaining lines just include HTML tags and tell the transformer which sub-elements to select for display from the Restaurant element.

```
<xsl:for-each select="PRICE_SELECTION">
  <xsl:sort select="PreferenceLevel" order="descending"
        data-type="number" />
  <xsl:sort select="Distance" order="ascending"
        data-type="number" />
    <TR>
      <TD>
        <xsl:value-of select="Name" />
      </TD>
      <TD>
        <xsl:value-of select="Cuisine" />
      </TD>
      <TD>
        <xsl:value-of select="Street" /><BR/>
        <xsl:value-of select="City" />,
        <xsl:value-of select="concat( State, ' ', Zip )" />
      </TD>
```

```
<TD>
  <xsl:value-of select="Phone" />
</TD>
```

Listing 2 XSLT stylesheet for generating markup

The string "PRICE_SELECTION" in the XSLT stylesheet is replaced by the application server before actually applying the XSLT file to filter out too cheap and too expensive restaurants, as well as remove the restaurants that the user likes least. The string replacement is done using XML's *Xpath* query language that views XML documents as trees and uses path expressions to select document components that need to be changed. The particular Xpath expression that is used to filter the restaurants is shown in Listing 3. As a result of the "PRICE_SELECTION" string replacement, the selected restaurants will have a *PriceHigh* element less than or equal to 40, a *PriceLow* element higher than or equal to 10, and a *PreferenceLevel* greater than 1.

```
Restaurants/Restaurant[ (PriceHigh <= 40) and
                        (PriceLow >= 10) and
                         PreferenceLevel > 1]
```

Listing 3 XPath expression for filtering restaurants

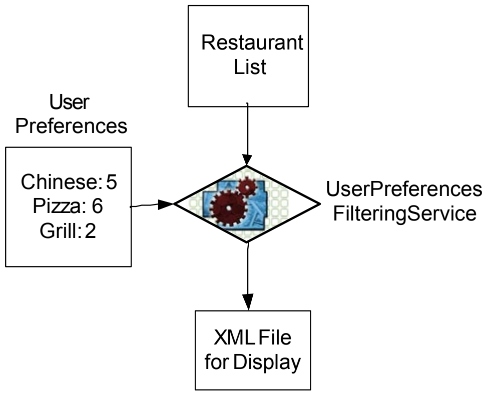
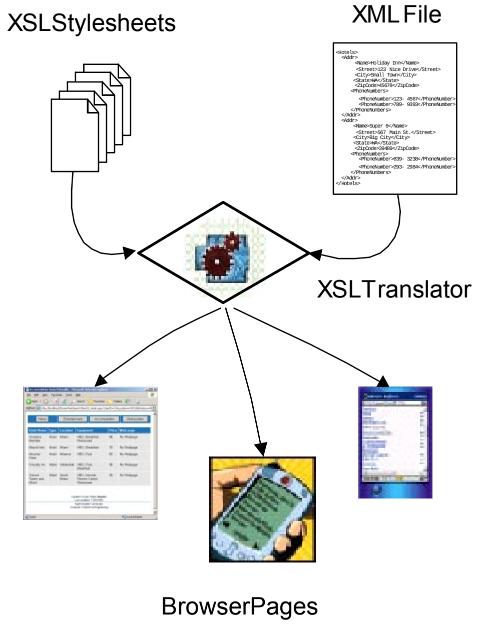


Figure 4 Cuisine preferences filtering

7. Device Adaptation

The mobile tourist's device has certain characteristics of screen size, image support, and browser type. The filtered list of restaurants is adapted correspondingly for display purposes. The screen size will determine what size pictures can be displayed on a user's device, and the number of pages that they will navigate through. The list of accepted content types indicates to the server what markup (e.g., WML, HTML) the device can interpret.

Depending on the content types supported by the client browser, an appropriate formatting sheet in XSLT (Extensible Stylesheet Language Transformations) is used to convert the content representation into appropriate markup for display on the user's mobile device. For example, if the device supports WML, an XSLT sheet that generates WML markup will be chosen. Figure 5 shows the XSL translator component that selects the right XSLT stylesheet for a user's client device and applies it to the XML list of restaurants for generating displayable markup. In the current implementation, the application server uses three different XSLT style sheets, one for a PC, one for a PDA and one for a cell phone. The system determines the kind of device by searching for certain information in the useragent string that is included in every HTTP request. This string contains information about the operating system, the browser version and the screen size. Currently, all XSLT files only generate HTML web pages. For example for PCs the HTML will be a nice table with all the information from the XML file, but for cell phones only part of that information will be used and the information is displayed in a list.



```
<Hotels>
  <Addr>
      <Name>Holiday Inn</Name>
       <Street>123 Nice Drive</Street>
      <City>Small Town</City>
      <State>WA</State>
      <ZipCode>45678</ZipCode>
    <PhoneNumbers>
       <PhoneNumber>123- 4567</PhoneNumber>
       <PhoneNumber>789- 9393</PhoneNumber>
    </PhoneNumbers>
  </Addr>
  <Addr>
      <Name>Super 6</Name>
       <Street>567 Main St.</Street>
      <City>Big City</City>
      <State>WA</State>
      <ZipCode>39489</ZipCode>
    <PhoneNumbers>
       <PhoneNumber>839- 3238</PhoneNumber>
       <PhoneNumber>293- 2984</PhoneNumber>
    </PhoneNumbers>
  </Addr>
</Hotels>
```

Figure 5 Device adaptation

8. Current Status and Future Extensions

We have presented the architecture of the CATIS system that incorporates a number of context variables relating to mobility, such as time and location, and type of device. Our prototype, supporting three different type of Web services and implemented using Microsoft .Net is operational and is entering the second version.

In order to support effectively mobility, other variables such as the surrounding environment need to be considered also. For example, if a user is driving a car, it makes sense to see if there are any Web services that can transfer the required information to his wireless PDA in audio form. As a further enhancement of this scenario, let us consider the case when the user is driving his car and his wife is traveling with him. In this case, we may consider also displaying the information about a given service in video form. This discussion implies that we also need to keep track in our UDDI registery of the media capabilities of the services, i.e., text, audio or video. Currently, the Web service framework does not provide this capability and we plan to investigate corresponding extensions to the UDDI directory service.

The current implementation uses two dynamic context variables, namely time and location (in addition to static user preferences). Based on the time of day, the application server, to which the mobile device connects, decides which type of service to invoke, i.e., a hotel finder, a park finder, or a restaurant finder. Currently, the location is determined by the user specifying a zip code. This provides a location with large granularity. Distance from the user's current location is specified in terms of number of adjacent zip codes rather than actual miles. We intend to incorporate fine-grained location tracking with GPS enabled devices, such as cell phones with built-in GPS receivers.

We also plan to investigate issues regarding the scalability of our system. Consider a metropolitan area having thousands of mobile users. In our current implementation, the Web services send information relevant to a given geographic area to the application server and this latter network element filters the information for a particular user. In order for these operations to be performed efficiently, their costs need to be amortized across other users.

Caching the results from the Web services is another important issue that we plan to address. Rather than issuing a request to the Web services every time a user wants to get information from a certain area, it would be much faster to temporarily cache the results of a previous query and then serve users based on the data in the cache.

In addition, instead of sending separate requests to the corresponding Web services, the application server should recognize that one request suffice and after the response is received different filtering operations need to be performed for different users. Similarly, if the application server receives requests from users in neighboring areas, such as Evanston and Skokie it would make sense to aggregate them into a single request since the response set contains overlapping information. In both of these examples, the application server needs to perform some kind of *aggregation* of the requests. The response will be *split* into different results taking into account the user preferences.

Furthermore, it also makes sense to make use of a distributed scheme, whereby we have different application servers for separate geographical areas and each application server will perform aggregation and splitting operations only for the users in its area.

9. References

[1] G. Abowd et al., "Cyberguide: a Mobile Context-Aware Tour Guide", *Wireless Networks 3*, pp. 421-433, 1997.

[2] M.van Bekkum et al., "A User-Centric Design of a Personal Service Environment", *3rd WWRF Proc.*, Sept. 2001.

[3] R. Blättler , "Context-Aware Information System for Mobile Users" MS. Project, Northwestern University, June 2002.

[4] A. Carzaniga and A. Wolf, "Content-Based Networking: A New Communication Infrastructure", *Proceedings NSF Workshop on Infrastructure for Mobile and Wireless Systems*, 2002.

[5] G. Chen and D. Kotz, "A Survey of Context-Aware Mobile Computing Research", *Technical Report TR2000-381*, Dept. of Computer Science, Dartmouth College, 2000.

[6] G. Chen and D. Kotz, "Context Aggregation and Dissemination in Ubiquitous Computing Systems", *Technical Report TR2002-420*, Dept. of Computer Science, Dartmouth College, 2002.

[7] K. Cheverst et al., "Experiences of Developing and Deploying a Context-Aware Tourist Guide: The GUIDE Project", *ACM MOBICOM 2000*.

[8] F. Curbera et al., "Unraveling the Web Services Web," *IEEE Internet Computing*, March/April 2002.

[9] EC 5th Framework Program, "M-ToGuide", www.mtoguide.org, 2003.

[10] FIPA, "FIPA Personal Travel Assistance Specification", www.fipa.org, 2001/08/10.

[11] Y. Huang and H. Garcia-Molina, "Publish/Subscribe in a Mobile Environment," *Proc. 2nd ACM Intern. Workshop on Data Engineering for Wireless and Mobile Access*, May 2002.

[12] A. Pashtan and J. Yanosy, "An Adaptable Services Framework", *5th WWRF Proc.*, Mar. 2002.

[13] H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, 1999

[14] A. Silberschatz, H. Korth, and S. Sudarshan, *Database System Concepts (4th ed.),* McGraw-Hill, 4. edition, 2001