Personal Service Areas for Mobile Web Applications

Location-based mobile services let wireless mobile users access Web-based information about resources in their immediate vicinity. The authors describe an algorithm that draws on context mobility elements such as the user's travel direction and speed to form personal service areas. Their experimental context-aware tourist information system (CATIS) leverages XML technologies and Web services to provide tourist information to mobile users based on these personal service areas and the users' personal preferences. Because Web service performance depends on the underlying databases, the authors also developed a layered caching scheme for storing environmental data to improve response time.

Ariel Pashtan
Aware Networks

Andrea Heusser and Peter Scheuermann
Northwestern University

The advent of location-tracking capabilities and their potential in location-based mobile services are major motivations for developing context-aware wireless information delivery systems. Such services let mobile users access information relevant to their situation. The spatiotemporal variables comprising the dynamic context make context-aware applications for mobile users much more complex. In addition to location, elements affecting a location-based service include travel direction and speed, time of day, personal preferences (such as favorite cuisine, hotel chain, or type of entertainment), and terminal type.

A main design consideration for location-based services is the ability to determine a geographical area of relevance for the mobile user. We've developed a novel algorithm that tracks user location, derives user direction and speed of travel, and consequently determines a service area's shape in a computationally efficient manner. To improve user response times, we developed a related layered caching scheme for storing environmental information. Using such a personalized definition of a service area, tourist information services, for example, can wirelessly deliver to tourists' mobile terminals city information that is better attuned to the users' mobility context. The "Personalized Tourist Services" sidebar discusses recent research in this area.

Context-Aware Services Architecture

We implemented an experimental context-aware mobile Web information system — the *context-aware tourist information system*. CATIS adapts Web-based tourist information using context parameters, including user preferences, retrieved from

Personalized Tourist Services

A context-aware system must be able to deliver personalized information — that is, information adapted to a user's specific needs. Several research projects have demonstrated the benefits of mobile context awareness for tourism.[1–3] The Foundation for Intelligent Physical Agents (FIPA)[4] and m-ToGuide,[5] a project sponsored by the European Commission's Fifth Framework Program, also target tourism applications for mobile users as services in which context plays a significant role in delivering pertinent information.

In general, *context* refers to information about the terminal platform, the user, and the surrounding environment.[6–8] *User context* typically refers to the user's personal preferences for information content (for example, type of cuisine when searching for a restaurant). *Environment context* includes elements such as the user's location, time of day, location of landmarks, and prevailing weather.

Mobile users can leverage context awareness to enhance their touring experience by receiving tailored information about landmarks and events in their vicinity. Wireless terminals that serve mobile users aren't conducive to interactivity (because of their small screens and keyboards, for example). Context awareness can improve user interaction because a context-aware application knows a priori the user's mobility situation (such as travel direction and speed), personal preferences, information interests, and environmental conditions and automatically adapts information delivery to the user's circumstances.

In the scenarios considered in this article, a tourist specifies categories of information interests that are captured in a user profile, and the tourist's wireless terminal displays location-based information that is pertinent to the tourist's interests. Mobility context can be a key element in adapting the delivered information to a relevant geographic area.

By adapting information to the user's detected trajectory and speed, context-aware applications could provide for a differentiated service not typically found in location-based services. Typically, mobile tourist information systems, such as the Guide project,[2] generate tour suggestions and display information relevant to a stop on the tour when they detect that the tourist is at the stop. Other systems, such as TIP,[3] display information relevant to the current location, taking into account the user's history of visited locations so as not to repeat previously delivered information. Still others, such as Metromix (http://metromix.chicagotribune.com), let the user select an area's name or zip code to display corresponding points of interest.

References

1. G. Abowd et al., "Cyberguide: A Mobile Context-Aware Tour Guide," *Wireless Networks*, vol. 3, no. 5, 1997, pp. 421-433.
2. K. Cheverst et al., "Experiences of Developing and Deploying a Context-Aware Tourist Guide: The Guide Project," *Proc. Int'l Conf. Mobile Computing and Networking* (MOBICOM), ACM Press, 2000, pp. 20-31.
3. A. Hinze and A. Voisard, "Location and Time-Based Information Delivery in Tourism," *Proc. Int'l Conf. Spatiotemporal Databases* (SSTD3), LNCS 2750, Springer, 2003, pp. 489-507.
4. *FIPA Personal Travel Assistance Specification*, Foundation for Intelligent Physical Agents, Aug. 2001; http://fipa.org/specs/fipa00080/XC00080B.html
5. *Mobile Tourist Guide*, IST-2001-36004, European Commission 5th Framework Program, 2003; http://www.motorolatele.com/MOTOnow/
6. G. Chen and D. Kotz, *A Survey of Context-Aware Mobile Computing Research*, tech. report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.
7. A. Pashtan and J. Yanosy, "An Adaptable Services Framework," *Proc. 5th Wireless World Research Forum Meeting* (WWRF), 2002 http://www.wireless-world-research.org.
8. A. Pashtan et al., "Adapting Content for Wireless Web Services," *IEEE Internet Computing*, vol. 7, no. 5, Sept./Oct. 2003, pp. 79-85.

a context and profile manager network element. Figure 1 (next page) shows the Web services-based CATIS architecture, which we implemented using Java Web services and Microsoft's .NET framework. The major architectural components include:

- A thin client terminal hosting a Web browser. The client terminals periodically send location updates to the network.
- An application server that delivers Web content prioritized according to user preferences. The application server formats the generated information content and adapts it for display on the client terminal.
- A context and profile manager (CPM) that tracks the user's dynamic context as well as static preferences (such as favored cuisine) in a user profile, which it stores in an XML database. The CPM also queries Web services and filters Web content data according to user profile context.
- A Universal Description, Discovery, and Integration services directory that provides a centralized registry of tourist information services (for example, a restaurant-finder service). The UDDI specifications describe service-related information and provide a query-and-update API for accessing registry service information.
- Web services that deliver tourism content. So far, we've implemented Web services for finding restaurants, hotels, and amusement parks. Each Web service is a wrapper around a database and provides functions for querying and retrieving service information. Web Services Description

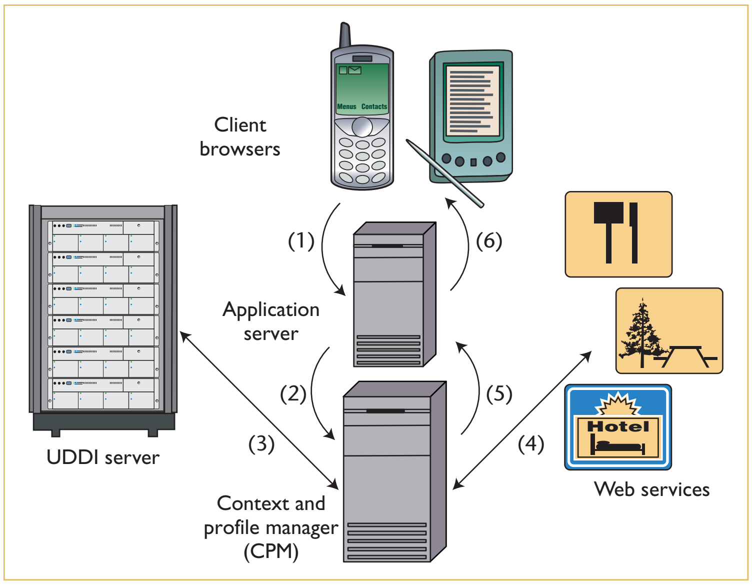


Figure 1. Context-aware tourist information system architecture. An application server formats generated content and adapts it for display on the client terminals. A UDDI service directory provides a centralized registry of tourist information services. The context and profile manager (CPM) tracks user preferences and stores them in user profiles.

Language (WSDL) documents published in the UDDI server describe these functions.

Given the Web's intrinsically distributed and heterogeneous nature, communication mechanisms must be platform-independent and as lightweight as possible. To realize such mechanisms, we used SOAP[1] on top of HTTP for all network element communication. The client terminals use HTTP for browser-based access to the application server and SOAP for relaying context information to the CPM.

To register with the system, a user logs on to the application server and enters his or her preferences by selecting appropriate fields in profile forms. The application server forwards the preferences to the CPM database, which aggregates them in user profiles. If the client terminal has a GPS receiver, it sends regular location updates to the CPM. Although we focus here on GPS-based methods in which user location is available at the mobile terminal, we could also use other location-tracking methods, such as network-based triangulation, which relies on multiple cell sites for determining a terminal's position. The CPM computes user direction and speed from the user's tracked location history and stores the latest dynamic context values in the user's profile.

A tourist seeking local information about nearby restaurants, for example, connects to the application server to request the information (step 1 in Figure 1). The application server forwards the query to the CPM (step 2), which queries the UDDI server for the addresses of available restaurant Web services (step 3). The CPM then sends a request to the identified Web services specifying a service area for the user (step 4). The Web services search their databases for the appropriate resources, filtering out those that fall outside the specified area, and return an XML list to the CPM. The CPM filters the XML list according to the user's context and augments the restaurant information with user and environment context, such as the user's cuisine preference level and distance and direction to the restaurant. Figure 2 shows the restaurant XML information augmented with context data (blue elements).

The CPM sends this context-enriched information to the application server (step 5). The application server then uses an XML stylesheet to generate a prioritized restaurant list in the appropriate markup for browser display (HTML, for example) and forwards it to the client terminal (step 6).

Mobile Location Areas

The concept of a service area is defined in other fields, and cellular networks, in particular, use wireless location areas to track mobile users so they can efficiently establish calls when users travel within a network or roam to other networks. Researchers have proposed various methodologies (such as two-tier and multitier schemas) for specifying mobile location areas.[2] The main trade-off is generally between update frequency when mobile users cross cell boundaries and search time latency when a particular user must be located. Cellular networks decrease user-tracking cost by forming groups of cells, or *paging location areas*. Mobile users update their locations in a location registry database only when crossing boundaries between location areas. To find a user, a mobile switching center queries all cells in a given location area in a process referred to as *paging*.

A complementary recent research direction, *moving objects databases* (MOD),[3,4] focuses on designing databases that manage spatiotemporal data of large numbers of moving objects. MOD addresses the unique modeling requirements of location data, moving object access methods, and the processing of novel types of spatiotemporal queries. MOD databases use dynamic attributes — that is, ones that change dynamically over time without update operations — and allow queries that can simultaneously handle space and time constraints, including projections of the future. For example, in the query, "retrieve the fleet's vehicles that will be in Northbrook between 3 p.m. and 5 p.m.," the specified location area is the town of Northbrook. To answer such a query, the database must keep models of user trajectories and events (for example, accidents or traffic congestion) that might cause the database to reevaluate the query in order to maintain the presented data's accuracy. A MOD-based tracking system can include a specification of the boundaries of the areas of interest for preferred services. However, MOD databases can't explicitly incorporate the user preferences and all mobility parameters into their query-processing framework. The personal service area, as implemented in the CPM, is a context-sensitive dynamic area that the CPM automatically determines from the user's current mobility circumstances.

```
</Restaurants>
   <Restaurant>
     <Name>Oceanique</Name>
     <Street>505 Maine St</Street>
     <Cuisine>Caribbean</Cuisine>
     <PriceLow>10</PriceLow>
     <PriceHigh>15</PriceHigh>
     <Latitude>42.034052</Latitude>
     <Longitude>-87.67793</Longitude>
     <Open>11:00</Open>
     <Close>22:00</Close>
     ...

<CuisinePreferenceLevel>3</CuisinePreferenceLevel>
     <Distance>0.33</Distance>
     <Direction>-76.75</Direction>
   <Restaurant>
</Restaurants>
```

Figure 2. Example restaurant XML information augmented with user and environment context (in blue). The context data is retrieved from the user's profile and is used in a subsequent prioritization of the restaurant list.

Personal Service Area

A key issue in determining which local resources to display on the client terminal is finding the tourist's relevant surrounding region since the size and shape of this region is dependent on the user's mobility context.

CATIS tracks a user's previous locations and uses the data to compute the user's direction of travel and average speed. We devised a sliding window methodology that considers location, direction, and travel speed when determining the list of recommended points of interest (POIs). For example, it won't list POIs in the opposite course of travel or beyond a certain distance threshold.

Our sliding window algorithm uses a two-step approach in which it

- determines the tourist's pertinent surrounding area and then
- generates a list of POIs based on a preferred maximum travel distance.

This approach relies on detecting the user's direction of travel and leveraging a prepartition of the metropolitan area into a grid of square-shaped cells, as Figure 3 shows.

We predefine cell size such that the time to traverse a cell is about the time the user is willing to spend traveling to a listed POI at a given speed. The user's direction of travel determines the number of cells that the CPM needs to retrieve from a Web service following a user query. In Figure 3, the user issues two successive queries, at time 1 and time 2; the shaded cells represent the corresponding retrieved cells. The CPM derives the personal service area from the intersection of the

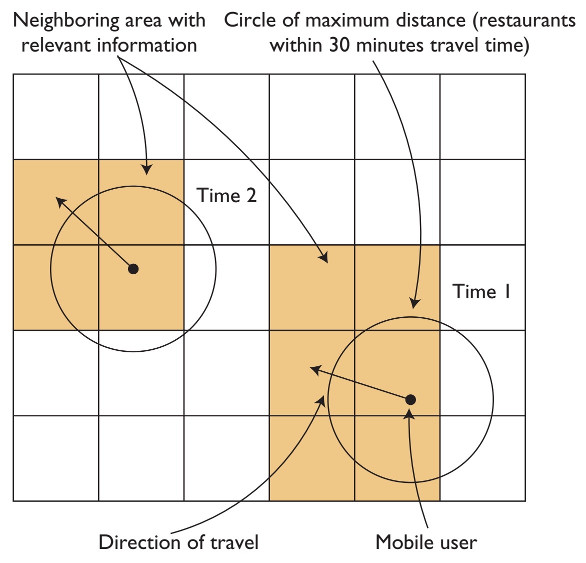

Neighboring area with relevant information
Circle of maximum distance (restaurants within 30 minutes travel time)
Time 2
Time 1
Direction of travel
Mobile user

Figure 3. Personal service area based on direction, speed, and distance to resource. The circles show the maximum distance that the user is willing to travel from his current location; in this case, half an hour of travel at the user's current speed.

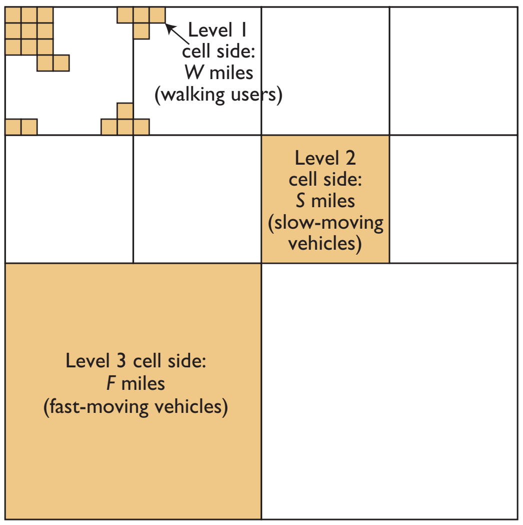
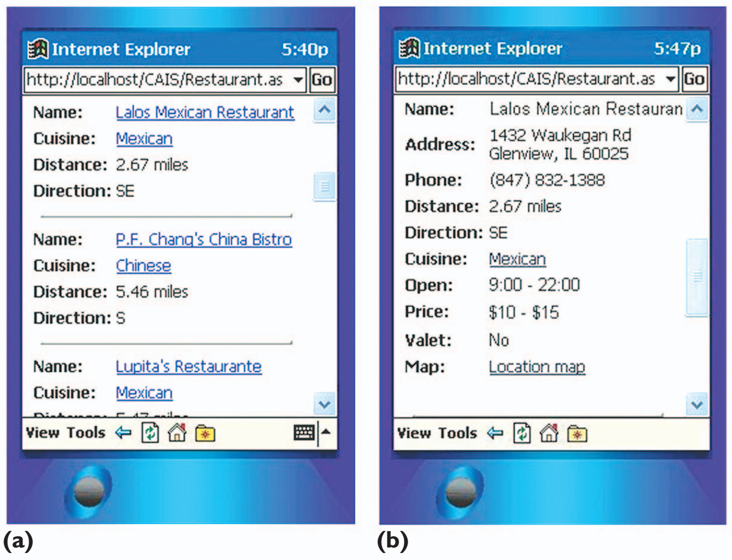

Level 1 cell side: W miles (walking users)
Level 2 cell side: S miles (slow-moving vehicles)
Level 3 cell side: F miles (fast-moving vehicles)

Figure 4. Overlapping grid layers used in determining personal service areas. Each point of interest falls into one cell at each of the three layers.


(a)

Internet Explorer 5:40p
http://localhost/CAIS/Restaurant.as Go

Name:	Lalos Mexican Restaurant
Cuisine:	Mexican
Distance:	2.67 miles
Direction:	SE
Name:	P.F. Chang's China Bistro
Cuisine:	Chinese
Distance:	5.46 miles
Direction:	S
Name:	Lupita's Restaurante
Cuisine:	Mexican

(b)

Internet Explorer 5:47p
http://localhost/CAIS/Restaurant.as Go

Name:	Lalos Mexican Restauran
Address:	1432 Waukegan Rd Glenview, IL 60025
Phone:	(847) 832-1388
Distance:	2.67 miles
Direction:	SE
Cuisine:	Mexican
Open:	9:00 - 22:00
Price:	$10 - $15
Valet:	No
Map:	Location map

Figure 5. Client terminal display. (a) The terminal lists restaurants by cuisine preferences and then by distance within the user's personal service area. (b) The user selects a restaurant on the list to get detailed information about it.

retrieved cells with a circle representing the maximum distance the user will travel to a listed POI. This algorithm proves more efficient than an exhaustive computation of distance over all POIs, and it adapts the list of relevant POIs to the user's direction of travel.

To attune service area selections to user speed, we refined the metropolitan POI grid to create three levels of overlapping grids, with one grid for each representative travel speed, as Figure 4 shows. Grid layering is a practically motivated structure that provides a good balance between the quadtree (a tree-based data structure) and the grid files[5] for efficient cache utilization.

Each POI, and each user's location, falls into exactly three cells, one at each level. The grid cells have side lengths of *W*, *S*, and *F* miles at levels 1, 2, and 3, respectively, where $W < S < F$. We use smaller grids for slower-moving users, basing grid-size selection on typical speed limits in the metropolitan area. Our current implementation has a grid partition like that in Figure 4, in which grids of size *W* are for walking users, grids of size *S* are for slow-moving (for example, 30 miles per hour) vehicles, and grids of size *F* are for fast-moving (for example, 60 miles per hour) vehicles. We could extend this scheme to include more grid levels if we desired higher accuracy in the service area computation.

The CPM determines the appropriate grid level according to the user's speed and the list of cells to be used in the personal service area computation is determined by the user's travel direction. After retrieving the cells' POI information from the Web service, a filtering step selects only those POIs within a predetermined distance — for example, within half an hour of travel at the user's current speed.

The application server orders the list of restaurants displayed on the user's terminal by cuisine preferences first, and then by distance within the user's personal service area, as the PDA-like screen in Figure 5a illustrates. As Figure 5b shows, selecting a restaurant on the list displays more detailed information.

System Performance

The CATIS restaurant Web service prototype includes about 2,200 restaurants from the greater Chicago area. We performed tests to evaluate system response to single-user queries. The tested system consisted of three Microsoft Windows-based Pentium 4 PCs: one for the application server, another for the CPM and UDDI directory, and the third for the Web service. The PCs communicated over the Internet.

Web Services Performance

The restaurant service prototype uses the Xindice native XML database (http://xml.apache.org/xindice/). Our tests revealed that each call from the

CPM to the Web service takes between 5 and 15 seconds. Using a Microsoft Access database improved response time to between 1 and 5 seconds.

We repeated the tests when the Web service didn't query a database, which let us measure the SOAP messaging overhead across the network. In this case, the Web service response time was typically under 1 second. Although network delays can impact response times, we concluded that the type of database used to store Web service data is the main factor impacting Web service performance.

Caching Solutions

As previously described, we specify a user's personal service area in terms of cells covering a geographical area in the mobile user's vicinity. In a city or other heavily visited area, many personal service areas might overlap. Because each Web service request can take up to 15 seconds, caching restaurant information can be beneficial because many users might have similar requests.

The CPM caches the Web services information in the three-layered grid structure shown in Figure 4. When different mobile users in the same geographical area make successive requests to the CPM, rather than issuing successive Web service calls for different requests in the same area, the CPM retrieves the requested POI information directly from its cache. Without caching, displaying a restaurant list can take on average 1 minute. Caching reduces the response time to between 2 and 10 seconds, depending on the number of restaurants in the area.

Because restaurant data doesn't change frequently, cache consistency wasn't an issue. Nevertheless, the CPM does refresh its cache periodically to allow for restaurant information changes.

Future Considerations

Currently, the CATIS CPM uses geographic distance as a sorting criterion for displaying the restaurant list. However, two objects at a certain distance *d* apart might not be reachable via road segments of distance *d* because a physical barrier such as a river might be between them. This information isn't available in the restaurant Web services databases. We intend to explore whether we could link this information from a commercially available GIS software package to our system.

In the future, we plan to study different cache-replacement policies that will account for the type and size of Web service data. In addition, we intend to investigate various cache-refreshment policies for dynamic information, such as a hotel's room availability or a restaurant's wait time, provided this information is readily available.

Because metropolitan areas can have many thousands of mobile users, we must consider additional scalability features beyond caching. In particular, it appears that a distributed architecture would fit well into this scenario. We could replicate and distribute all CATIS components, including the application server, the CPM, and the Web services. Our framework could potentially use one scheme that has proven successful with scalable distributed file systems.[6] Namely, when an application server or CPM becomes overloaded and the system utilization remains above a specific threshold, the network designer could split its contents by adding new servers or CPMs.

Finally, for information-adaptation purposes, the system could consider other context variables, such as the mobile user's current transport mode. For example, searching for Web services that can deliver information in audio form makes sense if a user is driving a car. Automated approaches that detect user situation changes can decide when to activate corresponding Web information services.

Acknowledgments

The authors thank Agnes Voisard and Goce Trajcevski for their comments.

References

1. F. Curbera et al., "Unraveling the Web Services Web," *IEEE Internet Computing*, Mar./Apr. 2002, pp. 86-93.
2. E. Pitoura and G. Samaras, "Locating Objects in Mobile Computing," *IEEE Trans. Knowledge and Data Eng.*, July/Aug. 2001, pp. 571-592.
3. G. Trajcevski and P. Scheuermann, "Triggers and Continuous Queries in Moving Objects Databases," *Proc. 14th Int'l Workshop Database and Expert Systems Applications* (DEXA 03), IEEE Press, 2003, pp. 905-910.
4. O. Wolfson, "Moving Objects Information Management: The Database Challenge," *Proc. 5th Workshop Next Generation Information Technologies and Systems* (NGITS), LNCS 2382, Springer, 2002, pp. 75-89.
5. H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, 1990.
6. R. Vingralek et al., "Distributed File Organization with Scalable Cost/Performance," *Proc. ACM-SIGMOD Int'l Conf. Management of Data*, ACM Press, 1994, pp. 253-264.

Ariel Pashtan is president of Aware Networks, where he develops Web-based services and enabling technologies for the wireless Internet. His research interests include personalized and location-based services for the mobile Web. He has a PhD in computer science from Kansas State University and is a graduate of Kellogg's Management Institute. He is the author of *Mobile Web Services* (Cambridge University Press, 2005) and is a member of the IEEE. Contact him at ariel@awarenetworks.com.

Andrea Heusser is a software developer for UBS Investment Bank in Chicago. His research interests include mobile and wireless computing and Web technologies. He has an MS in computer engineering from Northwestern University. Contact him at aheusser@gmx.ch.

Peter Scheuermann is a professor of electrical and computer Engineering at Northwestern University. His current research interests include Web-related technologies, data warehousing and data mining, parallel and distributed database systems, I/O systems, and spatial databases. He has a PhD from the State University of New York at Stony Brook. He serves on the editorial board of *IEEE Transactions of Data and Knowledge Engineering* and is an IEEE Fellow. Contact him at peters@ece.nwu.edu.