Using and Determining Location in a Context-Sensitive Tour Guide

In a study that provided unique insights into the challenges associated with developing location-based applications, the Lancaster Guide project used members of the general public to test a network-centric electronic tourist guide.

Nigel Davies
Lancaster University and University of Arizona

Keith Cheverst
Keith Mitchell
Lancaster University

Alon Efrat
University of Arizona

We began our work on the Lancaster Guide project in 1997 with the aim of exploring the issues arising from developing and deploying context-sensitive applications. Specifically, we aimed to develop and deploy a context-sensitive tour guide for the city of Lancaster, UK.

Motivated by the work on CyberGuide at Georgia Tech,[1] we envisioned a system that could provide tailor-made tours for visitors by adapting its behavior to changes in a user's location. For example, the system would generate a different tour depending on a visitor's interests, location, available time, financial limitations, mobility constraints, and local weather conditions. We also wanted the system to adapt tours as these conditions changed.

In contrast to the CyberGuide project, we adopted a network-centric approach. In Guide, the systems obtain information through a high-speed wireless network deployed throughout the target city. Our work explored the role of the network in supporting such location-based applications and provided data for developing future networks with similar characteristics. In addition, we decided to contrast our work with existing projects by developing a system that members of the general public could use. In this way, we hoped to gain feedback from real end users that would contribute to future applications and system designs.

After we implemented the Guide system in 1999, we ran a series of field trials involving members of the general public.[2,3] When we give talks describing our work, researchers and engineers who are developing similar systems often ask us questions that focus on two topics, which together form the basis for the subject matter of this article.

The first topic relates to our choice of positioning technology—beacons that broadcast using an IEEE 802.11 wireless network combined with user input. The most obvious choice for obtaining position information is GPS, but GPS is not necessarily the correct choice for location-based applications.

The second topic concerns techniques for generating custom tours for electronic city-guide systems. Guide generates these custom tours by taking into account multiple contextual triggers and user preferences. In practice, producing good tours and, indeed, assessing the quality of a tour are difficult tasks. While our analysis of techniques for producing custom tours is somewhat specific to the city-guide domain—which is itself an area of intense activity—we believe that the majority of our work is relevant to location-based systems in general.

THE GUIDE SYSTEM

Our system provides an electronic handheld guide that visitors to Lancaster can use to access information about the city, create tailored tours of the city, and access interactive services. The system consists of a tablet-based PC equipped with 802.11 wireless networking capabilities. We chose the Fujitsu TeamPad 7600[4] shown in Figure 1 and Lucent WaveLAN (recently renamed Orinoco) network cards, which provided a maximum bandwidth of 2 Mbps at the time of deployment. The TeamPad offers a good balance between performance and size: It features a large screen (800 × 600 pixels) with a transflexive display

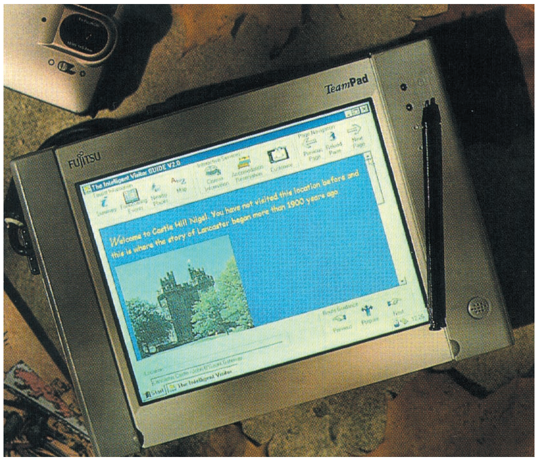

Figure 1. The Guide system consists of a tablet-based PC equipped with IEEE 802.11 wireless networking capabilities.

to enable viewing in bright sunlight, has a Pentium 166 MMX processor, and weighs approximately 850 grams.

The end-user systems use the wireless card to communicate with a series of base stations deployed around the city. These base stations each consist of a Linux-based server equipped with two network interfaces: One interface provides wireless access to the clients, and the other connects to the fixed network.

Cell servers broadcast pages of information that users in the cell's geographic area frequently access.[2] To improve scalability and accelerate user requests, the end-user system caches these pages. Most users request pages that do not require an explicit HTTP request because they are already in the end-user system's cache. Cache misses cause the system to send a request from the client to the cell server and add the requested page to a future broadcast cycle, which disseminates new information to all users in the cell.

Cell servers also periodically broadcast a beacon—a simple datagram that encapsulates a location identifier—to inform the end-user system of their cell location. Finally, the servers provide gateway functions that let the end-user system access services such as accommodation reservation systems.

Information model

Guide relies on a geographic model that includes data such as city landmarks and an associated set of Web pages that provide information about the city. The geographic model supports functionality such as route guidance and provides information about specific physical locations. The hyperlinked Web pages provide a familiar model for information access.

The geographic model contains two distinct object types: navigation point objects and location objects. Navigation point objects represent waypoints between location points such as intersections. Guide uses navigation points to provide route guidance. As Figure 2 shows, location objects represent points of interest in the city, such as landmarks, shops, and cafés. Location objects also encompass state information associated with the entity they represent and at least one link to an associated Web page describing the location. Examples of state information include whether the location is open or closed for business and the number of times the user has visited the location.

Guide can create relationships between navigation and location objects. These relationships can have attributes that model characteristics such as the distance between two points or the associated travel costs. Hypertext pages can contain tags that let object model information influence the way Guide displays pages to

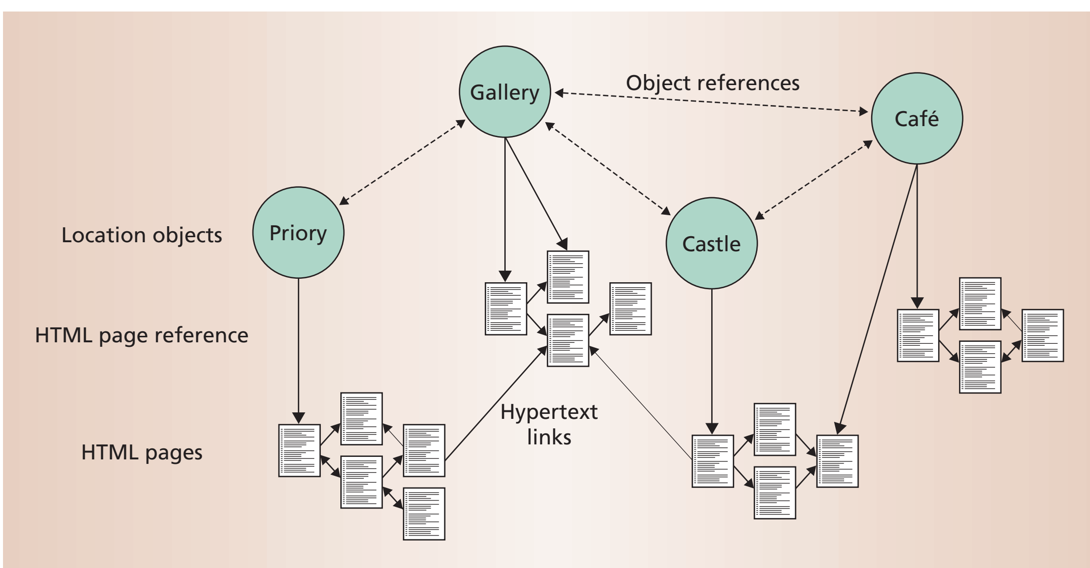

Figure 2. The Guide information model includes navigation point objects that assist in route guidance and location objects that present information about points of interest, such as landmarks and cafés, and a link to a Web page describing the location.

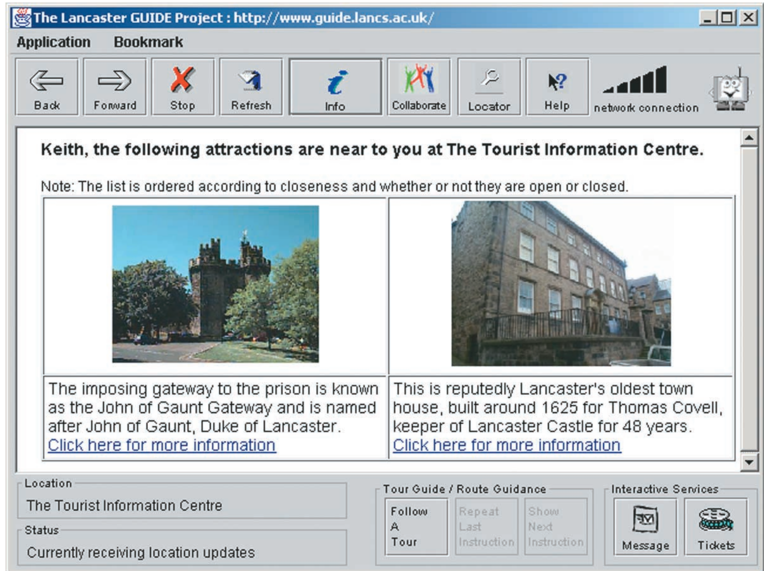

Figure 3. Example of a dynamically created page. Guide uses tags to create pages that query the object model to identify locations that are physically close to the user's current location.

the user. For example, a page can contain a tag that causes the model to display certain pieces of information only during a user's first visit to a location.

Figure 3 shows a page that Guide created dynamically using tags that interrogate the object model to determine locations that are physically close to the user's current location. In the current version of the system, the geographic component includes approximately 120 location and navigation objects and 400 relationships among these objects.

Application functionality

The Guide end-user system provides several capabilities:

- *Access to information.* Visitors can use Guide to retrieve information, including items relating to their current context such as their location and preferences.
- *Tailored city tours.* Users can enter a set of preferences and interests, and the system will generate an appropriate tour of the city. The unit then provides route guidance to help users move between locations on the tour.
- *Access to interactive services.* The system lets users access several interactive services, such as making ticket reservations or booking hotel accommodations.
- *Send and receive messages.* Users can exchange messages with other Guide users and with the Tourist Information Center staff. This can help groups of visitors keep in touch and ask questions.
- *Cooperative tools.* With context sharing,[5] users can let other members of their group know their location. Users can leave virtual stick-on notes[6] at specific locations in the city so they can share their experiences with other tourists.

Following its implementation in 1999, we used the Guide system in a field trial that lasted four weeks and involved 60 volunteer users from the general public.[3]

LOCATION-BASED FUNCTIONALITY

Location-aware city guides use location information for two main purposes: presenting information relevant to a user's location ("the area where you are now standing was once the scene of …") and providing route guidance ("in 100 meters, turn left onto Market Street"). A number of technologies could be used to obtain the required location information.

GPS, the most widely deployed location technology system, is a satellite-based navigation aid originally developed by the US military. GPS receivers obtain signals from multiple satellites and use a triangulation process to determine their physical location, which is accurate to within approximately 10 meters. The precise accuracy depends on a wide range of factors, including the number of satellites visible to the receiver, the receiver's velocity, and prevailing atmospheric conditions.

The main advantage to using GPS in city guides is its ubiquitous coverage: GPS is a truly global system. In addition, GPS receivers are relatively compact, inexpensive, and do not require ongoing subscription charges. However, GPS has several distinct disadvantages. Most critically, to obtain an accurate positional fix, a GPS receiver must be able to locate at least three satellites, which precludes using the technology inside buildings and in areas where tall buildings block the satellite signals.

The emerging cellular location-based systems offer an alternative to GPS. These systems typically provide accuracy within approximately 50 meters for assisting emergency services providers in locating callers (http://www.fcc.gov/e911). However, location-based applications also can use the technology.

Finally, indoor location technologies such as Active Badge,[7] Active Bat,[8] Cricket,[9] and Radar[10] are available, but these systems are not suitable for use as the primary location mechanism for a city guide because they are not designed for deployment outdoors. A city guide could, however, use such technologies as part of an overlay scheme to provide detailed location information within a building.

Because cellular systems were not available when we designed the Guide system, our choice for providing positional information was to use either GPS or a wireless network. Although we originally chose GPS, we found that it did not provide any significant advantage over network-based location beacons, which offer accuracy within 50 to 100 meters in most cases. This degree of accuracy is sufficient to display general information about a geographic area and to provide a starting point for offering route guidance.

Furthermore, in our experience, the location-based functionality that users most often require is for a guide to answer questions about objects they can see around

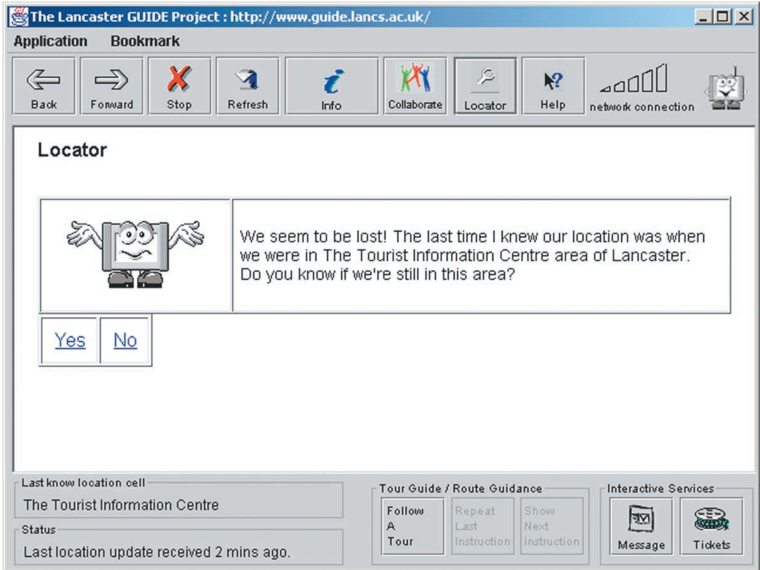

Figure 4. When the user activates the locator component, Guide asks a series of questions to help the user identify the specific location.

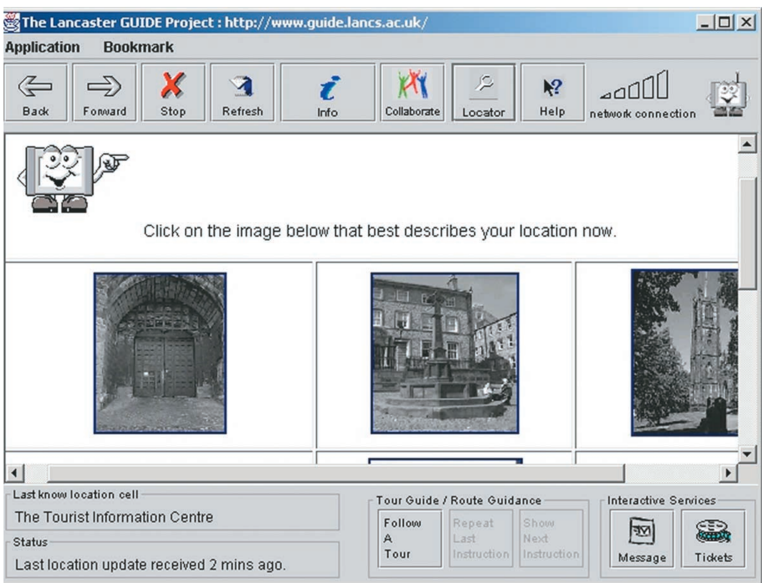

Figure 5. Guide displays thumbnail images to help users determine their location.

them. We found that the additional accuracy that GPS provides does not help to answer such questions. Even if it can determine a person's exact location, an electronic guide can't answer a question in this form unless it can tell what the user is looking at. Therefore, we decided to use network beacons to provide approximate location information and then to use further input to refine the position where necessary.

This choice meant that we avoided the cost and additional power consumption associated with using a GPS receiver in addition to the network card a city guide requires to support information download and interactive applications.

THE GUIDE APPROACH

Guide's cell servers broadcast location beacons that identify the cell where the end system is currently located. In a worst-case scenario, this system provides an accuracy of approximately 350 meters, which reflects the transmission range of the WaveLAN network cards. In general, however, the 802.11 network's propagation patterns provide a much greater degree of accuracy. The network's relatively poor propagation through buildings means that city buildings typically bound Guide's cells. For example, if we place a cell in a small city square, the propagation down the access roads to the square will be relatively poor. Hence, end-user systems receiving beacons from the square's cell server typically would be located within or very close to the square.

We use a smoothing function to avoid problems at cell boundaries where an end-user system could receive location beacons from multiple sources. This function dictates that an end-user system must see at least *N* consecutive beacons from a new cell server before reporting the cell as its new location. End-user systems on access roads to a square are likely to receive beacons directed down these roads as well as beacons leaking data from the square itself. In future systems where more beacons will overlap, an obvious refinement would be to treat these overlaps as distinct locations.

System refinements

While it provides a useful starting point for location-based applications, the approach described thus far leaves two problems unresolved: what to do when the end-user system cannot receive a beacon signal and how to answer questions regarding specific objects within the city.

To address the first problem, users let Guide know when they think the system is lost. This typically occurs when the system gives the user directions that don't appear to correspond to local landmarks or when the network connection meter on the application's display indicates that the system has lost network connectivity. As Figure 4 shows, when the user activates the Guide's locator component, the application indicates the last location where contact was made and asks whether the user is still at this position. Guide asks a series of questions, using maps and photos as appropriate, to help the user identify the specific area.

If the user has moved since the last confirmed location fix, the system asks for information about the current location—either a street name, a city area, or the closest landmark. The system then shows a series of thumbnails and allows the user to select the one that corresponds to the current location, as Figure 5 shows. These thumbnails help ensure that the location is correct and provide the user with positive feedback at the end of the location process. As a fallback strategy, the user can browse through lists of icons to look for places that appear familiar.

Object identification

Guide uses a similar approach to provide information about specific objects within the city. When the user sees an object at the present location and asks a question about it, the system checks to see if it has an up-to-date location fix. If not, Guide invokes the locator component. Once Guide knows the user's current location, it asks whether the user is looking at something nearby or far away. This simple question helps reduce the search space for potential target objects.

The system then displays a series of potential target objects and asks the user to select an object on screen. In most cases, it is easy to predict the objects a user will ask about, and the system works extremely well with only about 12 thumbnails for any given location. Of course, in locations with many objects, using additional questions—for example, "Are you looking at a building?"—could reduce the set of possible objects to a manageable number.

During the field trial, we explored users' reactions to these methods of locating and identifying objects. Overall, the results were extremely positive. Users could easily interact with the system to identify and retrieve information on a range of landmarks. We also noticed an unexpected side effect of our approach to object identification: The list of thumbnails displayed guided a user's exploration of a location to some extent. When users request information about something they can see, Guide presents a set of pictures of other nearby objects they might find interesting. This raises a question about whether it would be beneficial to use image recognition to automatically identify an object, even if this were possible.

DEVELOPING CUSTOM TOURS

To fulfill its role as an electronic city guide, we considered it important for the Guide system to create custom tours. Creating a city tour is a difficult problem because a good tour is not simply the shortest path between a set of points. A good tour depends on achieving the correct balance between several factors. For example, a professional human guide would take into account at least the following points:

- The visitor's interests.
- The city's attractions. Visitors often want to visit a city's highlights, even if they do not fall directly within their normal interests.
- Mobility constraints. Physical limitations can restrict a visitor's travel patterns through the city.
- Available time. Most visitors have limited time to spend in a city, and tours must maximize the use of this time without making the visitor feel rushed.
- Time sensitivity of attractions. In many cases there are better and worse times to visit specific attractions. For example, in a temperate climate, it is probably better to visit a park in the afternoon during the warmer hours of a day, while it might be best to visit a museum in the morning before the crowds arrive.
- Weather. In many cases, the prevailing weather makes a big difference in the amount of time visitors spend at an attraction. The weather also affects the desirability of certain travel paths. For example, if it is raining, users will clearly ignore an electronic tour guide that suggests a stroll along a river bank followed by a picnic.
- Cost. Many visitors have financial constraints that a tour guide needs to understand.

In most cases, the system works well with only about 12 thumbnails for any given location.

Sometimes an unexpected event—a sudden change in the weather, for example, or a visitor spends longer than anticipated at a particular attraction—requires recalculating a tour. To customize tours, the Guide system allocates numeric values to attractions and to routes between attractions. Both the current context—weather, time, and so forth—and user preferences can modify these values. Comparing the total scores for a set of attractions and routes provides a starting point for constructing quality tours based on contextual triggers and varying constraints.

Destinations concept

Having experimented with several different means of presenting attractions to users, we found the most practical way is to group locations as destinations. For example, a visitor to Lancaster can select the destination "Castle Hill" without having to identify all the different attractions in that area. Clustering destinations reflects people's natural desire to minimize travel time within a tour. Guide uses criteria such as historical interest or tourist popularity to sort destinations for users to view.

Users can also enter any additional constraints—such as available time and funds—that would influence the tour. The system generates a full set of tours that encompass the attractions the user requested, produces a total score for each tour, and then recommends the tour with the highest score. The system incorporates several techniques to improve a given tour's overall score. For example, consider the following scenario: It is 9:30 a.m. and a tourist has just picked up a Guide unit from the tourist information center. The tourist asks the system to generate a day-long tour that visits two places, a nearby museum (which opens at 10:30 a.m.) and a park on the other side of town. The system recognizes that the best sequence is to visit the museum first and then spend the afternoon in the park; it therefore suggests padding activities—such as

Using the destinations concept can dramatically reduce the number of permutations the system needs to calculate.

stopping at a nearby café or exhibition—to occupy the visitor until the museum opens.

Brute-force approach

The tour generation system relies on a geographic model in which each location object stores a range of scores reflecting the appropriateness of visiting that location for a particular set of context inputs. When a user selects a subset of a city's attractions, the system generates the optimal ordering and path that meet tour acceptability criteria in terms of travel time and cost. The requirements become significantly more complicated when the system adds padding activities to improve a tour's score. The effort the system requires to compute the padding activities that fill time between attractions depends to a certain extent on the ratio of user-suggested attractions to padding. Padding with a lot of additional activities can easily produce an enormous number of possible tours that require evaluation.

Our current system uses an approach that lies between these two extremes. We use a brute-force approach in which the system generates scores for all possible permutations of selected attractions and then selects the tour with the highest score. Each attraction has only one associated padding attraction that the system can use to improve the tour's score. The system can use padding to improve a tour's overall score, but it cannot include arbitrary additional city attractions to improve the quality.

The time required to calculate the optimal tour increases exponentially with the number of attractions to visit. Therefore, with the current end-user system and implementation, Guide cannot generate tours for more than nine attractions in a reasonable period of time. Indeed, because our problem is in essence a variation on the traveling salesperson problem, it is likely to be NP-hard.[11] While nine attractions are enough for many tours, a topic for further research is to consider how mobile devices might generate longer tours or tours involving arbitrary padding. We believe the key to solving this problem will be a combination of techniques that reduce the number of attractions that Guide must consider, for example, by pruning a hierarchy of possible tours, to discard permutations at an early stage of analysis.

Determining how to rank attractions in terms of their suitability for use as padding requires further analysis. Geographic placement, score, and interest to the user are obvious criteria for helping to reduce the vast number of possible permutations that a high degree of padding introduces. Because we currently model each location as a distinct Java object, we need to combine these algorithmic solutions with improvements in our implementation.

We propose using the destinations concept—originally introduced to facilitate user interaction—to reduce the number of separate elements that comprise a tour. Using the destinations concept in this way is possible if we assume that once a tourist has visited an attraction at a given destination (d), an optimal tour is unlikely to require the tourist to visit an attraction in another destination (d*), return to destination d, and then to d*. This technique can dramatically reduce the number of permutations the system needs to calculate. We cannot assume that an optimal tour visits all locations in a destination before heading to a new destination, especially because circular tours often involve passing through a destination twice.

An alternative design option is having the tour guide generation components of the Guide system run on a network server. However, recalculating tours on a regular basis would require greater connectivity than the current Guide system provides.

User feedback

In general, user reaction to Guide's tour-creation functionality was negative. Discussions with the field trial participants revealed that the effort required to create a custom tour was the primary reason for their negative reaction. In particular, visitors picking up a unit from the tourist information office wanted to start sightseeing rather than spending time entering preferences into the Guide unit.

As a result of this feedback, we added pregenerated standard tours such as "Lancaster's Best" and "Lancaster and the Sea: Maritime History." Users find that selecting one of these standard tours is very straightforward, and since we introduced this feature, they seldom opt to create custom tours. We need to focus future research on designing the system to encourage users to create a custom tour before we spend significant effort on improving our tour-generation algorithms.

The Guide system provides unique insight into the challenges facing developers of location-based applications. Focusing our research on a tour-guide system for end users has helped us gain practical experience in how to construct one class of location-based applications.

To work effectively, Guide relies not only on technology but also on assistance from users. This partnership offers distinct benefits, both in terms of accuracy and in fostering a relationship between the end user and the Guide system. Our goal is to create a more engaging and compelling experience than might otherwise be possible if we relied on technology alone.

Our techniques are clearly applicable to a wide range of location-based applications, especially those that use cellular location systems. We are currently working on porting the Guide system to the Compaq IPAQ PDA, which will allow us to explore system development and usability issues for a new class of device. ✷

References

1. S. Long et al., "Rapid Prototyping of Mobile Context-Aware Applications: The Cyberguide Case Study," *Proc. 2nd Ann. Int'l Conf. Mobile Computing and Networking* (Mobicom 96), ACM Press, New York, 1996, pp. 97-107.
2. N. Davies et al., "Caches in the Air: Disseminating Information in the Guide System," *Proc. 2nd IEEE Workshop Mobile Computing Systems and Applications* (WMCSA 99), IEEE Press, Piscataway, N.J., 1999, pp. 11-19.
3. K. Cheverst et al., "Experiences of Developing and Deploying a Context-Aware Tourist Guide: The Lancaster Guide Project," *Proc. 6th Ann. Int'l Conf. Mobile Computing and Networking* (Mobicom 00), ACM Press, New York, 2000, pp. 20-31.
4. Fujitsu, "ICL's Family of Handheld TeamPad," http://www.icl.com/retail/handheld/7100.htm (current July 2001).
5. K. Cheverst et al., "The Role of Shared Context in Supporting Cooperation between City Visitors," to appear in *Computers & Graphics*, vol. 25, no. 4, 2001.
6. P.J. Brown, "The Stick-E Document: A Framework for Creating Context-Aware Applications," *Proc. Int'l Conf. Electronic Documents, Document Manipulation, and Document Dissemination* (EP 96), Palo Alto, Calif., 1996, pp. 259-272. Proceedings published in *Origination, Dissemination, and Design* (EP-ODD), vol. 8, no. 1, John Wiley & Sons, New York, 1996, pp. 1-13.
7. R. Want et al., "The Active Badge Location System," Olivetti Research Laboratory, *ACM Trans. Information Sys.*, vol. 10, no. 1, 1992, pp. 91-102.
8. A. Harter et al., "The Anatomy of a Context-Aware Application," *Proc. 5th Ann. Int'l Conf. Mobile Computing and Networking* (Mobicom 99), ACM Press, New York, 1999, pp. 59-68.
9. N. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," *Proc. 6th Ann. Int'l Conf. Mobile Computing and Networking* (Mobicom 00), ACM Press, New York, 2000, pp. 32-43.
10. P. Bahland and V.N. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proc. IEEE Infocom 2000*, IEEE Press, Piscataway, N.J., 2000, pp. 775-784.
11. E.L. Lawler, ed., *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley & Sons, New York, 1985.

Nigel Davies *is an associate professor in the Department of Computer Science at the University of Arizona and holds a Personal Chair in the Computing Department at Lancaster University. His research interests include systems and middleware support for mobile and context-aware computing. He received a PhD in computer science from Lancaster University. Contact him at nigel@cs.arizona.edu.*

Keith Cheverst *is a lecturer in the Computing Department at Lancaster University. His research interests include computer-supported cooperative work, context awareness, human-computer interaction, distributed systems, and ubiquitous computing. He received a PhD in computer science from Lancaster University. Contact him at kc@comp.lancs.ac.uk.*

Keith Mitchell *is a research associate in the Computing Department at Lancaster University. His research interests include context awareness, distributed systems, mobile and ubiquitous computing, and human-computer interaction. He received a BSc (Hons) degree in computer science from Lancaster University. Contact him at km@comp.lancs.ac.uk.*

Alon Efrat *is an assistant professor in the Department of Computer Science at the University of Arizona. His research interests include computational geometry and geometric optimizations. He received a PhD in computer science from Tel-Aviv University. Contact him at alon@cs.arizona.edu.*