Context-awareness in wearable and ubiquitous computing

Gregory D. Abowd, Anind Dey, Robert Orr & Jason Brotherton
GVU Center
College of Computing
Georgia Institute of Technology
Atlanta, GA USA 30332-0280
{abowd, anind, rjo, brothert}@cc.gatech.edu

Abstract

A common focus shared by researchers in mobile, ubiquitous and wearable computing is the attempt to break away from the traditional desktop computing paradigm. Computational services need to become as mobile as their users. Whether that service mobility is achieved by equipping the user with computational power or by instrumenting the environment, all services need to be extended to take advantage of the constantly changing context in which they are accessed. This paper will report on work done in the Future Computing Environments Group at Georgia Tech to provide infrastructure for context-aware computing. We will describe some of the fundamental issues involved in context-aware computing, solutions we have generated to provide a flexible infrastructure and several applications that take advantage of context-awareness to allow freedom from traditional desktop computing.

Keyword: Context-aware computing, ubiquitous computing, consumer applications, personal information management, tourism, voice-only interaction, positioning systems.

1 Introduction

Wearable computing. Mobile computing. Ubiquitous computing. Perhaps these terms are synonymous in the mind of the reader. Or perhaps they evoke some religious zeal concerning fundamental distinctions in future trends for computing. Regardless, researchers in these fields all share one common belief, namely that it is time to shift our research focus away from the traditional paradigm of desktop computing. Rather than force the user to search out and find the computer's interface, our new aim is to provide an interface that can take on the responsibility of locating and serving the user. In this paper we present the case in support of a research agenda on context-aware computing. We will provide some general mechanisms and architectures to support context-awareness and justify their effectiveness through a number of case studies on applications that benefit from context-aware services.

The work reported here is a summary of research within the Future Computing Environments (FCE) Group at Georgia Tech. The FCE Group is dedicated to the invention of novel applications of computing technology to assist everyday activities. We are trying to create an array of computational services that permeate everyday life without becoming too much of a physical, cognitive or social burden. Some of the principles we have used to direct our research method are

- Maintain an applications focus, as opposed to a technology infrastructure focus.
- Assume a mode of "fail fast" research, otherwise known as rapid prototyping.
- Do not spend too much time predicting the future; aim to invent it.

After enough experience has been gained through rapidly prototyping novel applications, general themes and mechanisms can be generated. In our work in FCE, we have generated three general themes, of which context-aware computing is one (the others are the capture, integration and access problem [2] and ubiquitous software services [18, 1]). The general mechanism for context-aware computing is summarized in the following steps:

1. Collect information on the user's physical, informational or emotional state.

2. Analyze the information, either by treating it as an independent variable or by combining it with other information collected in the past or present.
3. Perform some action based on the analysis.
4. Repeat from Step 1, with some adaptation based on previous iterations.

After providing a brief overview of related research on context-aware computing, we will give an overview of four different attempts to provide context-aware computing. Each attempt is a project that has at least one (and usually more) running prototypes that we have developed. Greater details of each project can be found at our Web site (http://www.cc.gatech.edu/fce) or through various cited publications. Each project will be presented in terms of a motivating application, discussion of how context is important in that application, some general mechanisms or architectural solutions that were used in our prototyping efforts and a discussion of some issues that have arisen with respect to context-aware computing.

2 Background and related work

Future computing environments promise to free the user from the constraints of stationary desktop computing, so researchers should focus on what applications maximally benefit from mobility. Highly portable devices, such as personal digital assistants (PDAs), pagers, and cellular telephones are starting to proliferate. The wearable computing community is rapidly providing even more power to support a mobile user. Too many of the applications provided on these portable computing devices, however, are simple duplications of what we have on our desktops, or simple messaging devices that work too much like traditional ones. None of these devices take into account the one thing that changes most when a user is mobile —location. Building applications that are customized to the user's context can be of benefit in stationary modes of interaction as well. We define context-aware computing as any attempt to use knowledge of a user's physical, social, informational and even emotional state as input to adapt the behavior of one or more computational services.

The majority of context-aware computing to date has been restricted to location-aware computing for mobile applications, and our own work started out that way as well. In thinking about and developing our own location-aware application, we were greatly influenced by work such as the PARCTab at Xerox PARC [16], the InfoPad project at Berkeley [12], the Olivetti Active Badge system [16] and the Personal Shopping Assistant proposed at AT&T [5]. A more general programming framework for describing location-aware objects was the subject of Schilit's thesis and reflected a lot of the work done at PARC [15].

There has been some interesting work recently directly related to context-aware computing. Essa and Pentland have used computational perception techniques in an attempt to match actual facial expressions with some prescribed expressions indicating the state of the human (e.g., smiling, frowning, surprised, etc.) [8, 9]. Though this work does not claim to be a way to predict human emotions, there is a clear suggestion of how this and related perception research can improve the quality of contextual information that can be gathered. Picard's work on affective computing [14] suggests a similar objective, only through the use of bio-electric signals, coupled with theories on emotion and cognition. Informational context, defined below in Section 4, has been the subject of work done at Apple (Data Detectors [4]) and Intel (Pandit and Kalbag's Selection Recognition Agent [13]).

3 Cyberguide: Using location to provide context

There are a number of PDAs available today, yet there are not many people who are devoted to using them, for a wide variety of reasons. One way to thwart this lack of market success is to invent new ways to use a PDA. The size of current PDAs is similar in size to a guidebook that a tourist might take with them on vacation. The book lists places to visit and provides important practical information, such as locations of interesting sights, or categorization of hotels and restaurants. The one thing a book does not know, however, is where the tourist is located when they want information. Position information augmenting an electronic guidebook could address that problem. We initiated the Cyberguide project to experiment with location as a context cue [10, 11, 3].

We have developed a number of Cyberguide prototypes that support both indoor and outdoor tours. Figure 1 shows a version of the outdoor Cyberguide used for touring local establishments in Atlanta.

3.1 How context-aware services arise in Cyberguide

In general, the purpose of context-awareness in a touring application is to predict what the user is at-

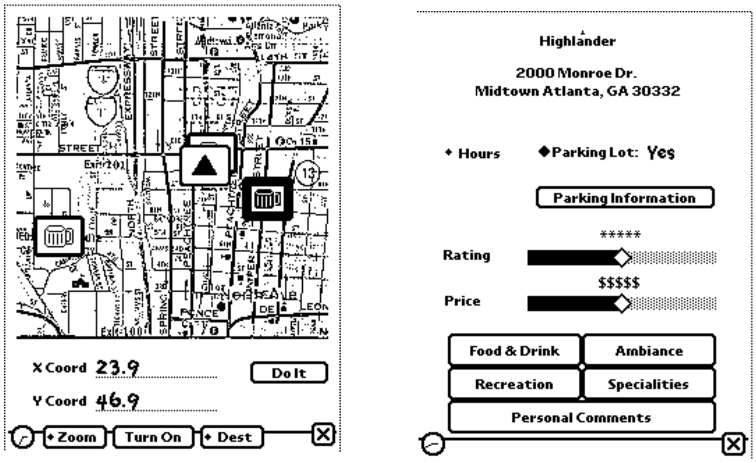

Figure 1: The CyBARguide interface. The left shows the interactive map indicating the user's location (the triangle) and the location of establishments previously visited (the beer mugs). The user modifiable database shown on the right supports the long-term development of touring information for a location.

tending to and provide information on that entity. As the user moves around, her location (the arrowhead in Figure 1) is updated on the map. The user can either explicitly query the map for information in the local surroundings, or that information can be automatically provided using some proximity algorithm. Getting close to something is an indication that information on that object is to be requested.

We currently have a very limited notion of context in Cyberguide —physical location and crude orientation. Tracking where a single individual is currently located provides adequate information for very simple context-awareness. We have experimented with capturing historical context (what sights have been visited already), but there are a number of other aspects of the tourist's context that can be useful. For instance, knowing where everyone else is located might suggest places of potential interest. Knowing the tourist's reaction to some exhibits would help in suggesting other related places of interest. Being aware of time of day or day of week may aid in more intelligent user queries of interesting attractions or activities.

3.2 Mechanisms developed

The strategy in the Cyberguide project was to build a number of prototypes that varied on certain critical features. This was our first experimentation with context-aware computing, so we chose to work with the most easily obtained physical context —the position of a mobile user. We developed prototypes using both indoor and outdoor positioning systems. Our high-level design of the family of Cyberguide prototypes separated the physical positioning system from the rest of the system, in the hope that a system built for outdoor use could be easily adapted for indoor use, and vice versa.

Outdoor positioning was a simple matter, as it is now easy to purchase inexpensive off-the-shelf GPS systems with a serial interface. The high price of the infrastructure to support GPS (satellites) has been paid for by military applications and this cost is not passed on to the consumer. Indoor positioning is a more difficult proposition, as we discuss below.

3.3 Issues

3.3.1 Precision of positioning information

It is easy to get caught up in the particulars of positioning information. A lesson we have learned is that it is less important to focus on more and more precise positioning for the purpose of information delivery. This is not the case for a system which would require the registration of information overlaid on top of a physical object, as is suggested by augmented reality research. In Cyberguide, the key is to determine what the user is attending to in order to predict what information they are likely to request. It would be much more useful to augment crude positioning information with orientation information.

3.3.2 The difficulties of indoor positioning

There are a number of indoor positioning systems currently on the market, and others that are simple enough to assemble. The systems vary dramatically in price and resolution. There is no one system that can meet every user need. Most systems excel in a few dimensions and are forced to compromise in others. The dimensions can be broken into two categories: quantitative and qualitative. The quantitative dimensions are: cost, number of users supported, area of coverage, data rate, latency, resolution, and accuracy. Scaling to support a large number of users and a large indoor area are the two hardest dimensions to satisfy simultaneously.

The qualitative dimensions are: obtrusiveness to the user, obtrusiveness to the environment, and location of information (either with the user or in the environment). For the CyberGuide application, these dimensions are very important. The positioning system used should be unobtrusive to the user and the environment. Users do not want to be constrained in any way by the technology, ruling out all systems that require the user to be tethered (mechanical and magnetic systems). Whether the user or the environment should hold position information depends on the application mode. If all information is local and CyberGuide is working in single user mode, the user benefits by having the position information. If information is distributed and CyberGuide is working in a collaborative mode, users benefit by having the system know their positions. This enables the system to dynamically provide pertinent information based on position and to notify users of each others' positions.

3.3.3 Providing ubiquitous positioning and communication

In the design of our prototypes, we modularized communications capabilities and positioning to enable us to experiment with different realizations of each. Ultimately, we are aiming for a positioning service that works both indoors and outdoors. Communication is important for several reasons. It is required in order to keep track of the location of all mobile units and to broadcast information to them. Our view of the mobile unit was not one of a mass storage device, so it will soon become necessary for information (a map of a location or a description of an interesting sight) to be delivered on demand to the unit, rather than stored locally. This is even more important for dynamic information, such as traffic reports.

There is an interesting relationship between the positioning and communication systems. In one version of Cyberguide we provided indoor cell-based IR positioning using communication beacons with a limited range. Communication and positioning requirements, therefore, could be satisfied by one system. However, wiht the Sharp IR units, we can couple positioning and communication, but the range of the IR link is so limited (3 feet) that communication is cumbersome. However, it can be impossible or undesirable to couple positioning and communication together. For example, if position is coming from GPS, then a separate means of communication must be used. It makes sense in some cases to use a short-range IR positioning system that doesn't cover all space but where it is available provides more exact positioning (and orientation) information. Communication, however, needs to be uniform throughout some space, so a short-range communication system would require many beacons to cover some larger area.

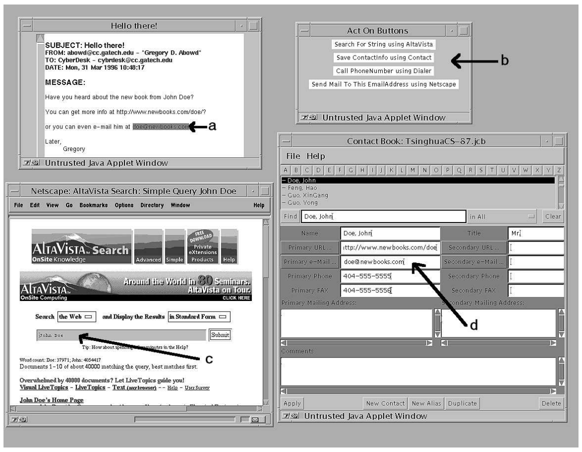

Figure 2: An annotated scenario of usage for the CyberDesk system.

4 CyberDesk: using informational context to automate service integration

4.1 Describing the application

The previous example of context-aware computing emphasized use of a user's physical context. In this second example, the CyberDesk project [18, 6], we use informational context to aid in the integration of desktop and network services. Informational context refers to any artifact (e.g., words on a screen or a picture at a museum) that the user is attending to. Examples of network and desktop services are an e-mail browser, a Web-based map service, or a contact manager. Interacting with one of these services might suggest some action to be invoked on another service. For example, as shown in Figure 2, the user can be reading an e-mail message which has information on a new book written by a favorite author. The e-mail contains a Web site address and an e-mail address for the author. The user highlights the e-mail address (a), thus explicitly announcing the information context, and the system gives him some suggestions (b) on what he can do: search for more information on the author, put the author's contact information in the contact manager, call the author, or send an e-mail to the author. He chooses the first two options (c and d), and saves the e-mail.

The behavior we are trying to provide in CyberDesk is *automatic* integration of separate services. Current software suites suffer from problems due to poor integration of their individual tools. They require the designer to think of all possible integrating behaviors and leave little flexibility to the user. CyberDesk is a component software framework that automatically integrates desktop and network services, requiring no integrating decisions to be made by the tool designers and giving total control to the user.

4.2 Use of context in CyberDesk

When integrating the behavior of separate applications, the programmer or the user needs to transfer data from one application to another. The typical scenario, as shown above, takes part of the output from one application (the name in the e-mail browser in Figure 2) and uses it as input to another application. In CyberDesk, after the user selects some string in one application, all possible type interpretations of that string are generated and announced (e.g., name, date, friend, etc.). Any resident service which can accept the announced data then becomes available to the user via a simple menu button. The context-awareness provides the ability to change the set of resident services and frees the user from having to remember any integration procedures.

4.3 Mechanisms and architecture

The CyberDesk system has a simple but innovative architecture. It is based on an event-driven model, where components act as event sources and/or event sinks. Events, in this current version, are generated from explicit user interaction with the system. The system consists of five core components: the Locator, the IntelliButton, the ActOn Button Bar, the desktop and network services, and the type converters. The Locator maintains the registry of event sources and sinks. This allows the IntelliButton to automatically find matches between event sources and event sinks based on a given input event, a task normally required of the system or service designer. The IntelliButton displays the matches in the form of suggestions to the user, via the ActOn Button Bar. It is through the ActOn Button Bar that the user accesses the integrating functionality of CyberDesk. The services are the event sources and sinks themselves, and are the tools the user ultimately wants to use. The type converters provide more powerful integrating behavior by converting given events into other events, allowing for a greater number of matches. These five components are discussed in greater detail elsewhere [6].

The CyberDesk framework was designed to be easily extensible, and we have so far been able to incorporate over 50 different desktop and network services, including ones that involve mobile PDAs connected over a wireless network. Simple extensions to CyberDesk include adding additional types, type converters, desktop services and network services. The real advantages with CyberDesk can be seen with more complex extensions that include adapting the behavior of CyberDesk to individual use and creating more interesting integrating behavior. Two examples we have implemented are chaining, in which the output of a service also acts as a type converter to announce higher level information to the system, and combining, using multiple pieces of information to invoke better services. For example, in chaining, a name selection might suggest an e-mail address lookup service. By executing this service automatically, the system can also suggest e-mail address-related services, such as searching newsgroups or sending an e-mail. In combining, using time with a name can allow the automatic search of a friend's calendar.

4.4 Issues

4.4.1 Reliance on explicit indication of context

The CyberDesk system relies on an explicit indication by the user of the informational context. For example, the user must select a region of text to activate any suggestions of integrating behavior. While this may not seem so onerous in the example cited above, when we start to use physical context informa-

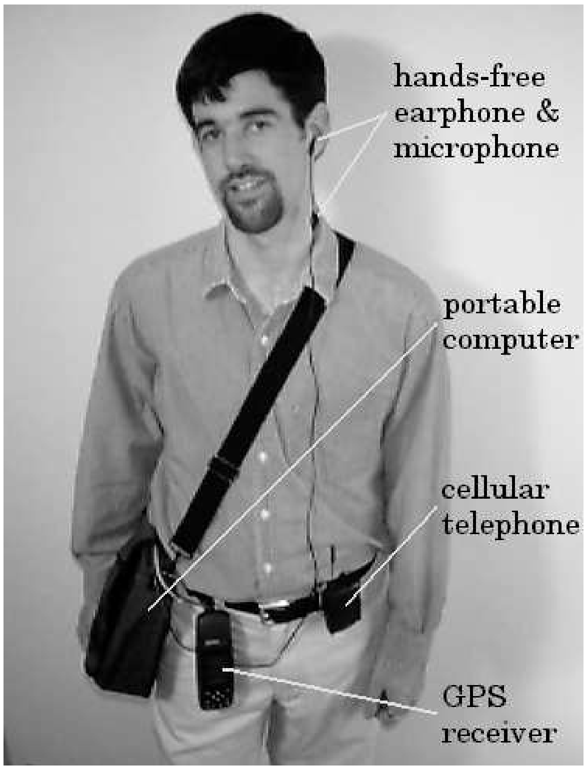

Figure 3: A tourist equipped to use the Savoir interface to Cyberguide.

tion, such as position, it will be less desirable for the user to explicitly announce information to the system. Moving around (considered implicit with respect to CyberDesk) should be the only action required by the user.

4.4.2 Suggestion overload

Perhaps the biggest limitation of the system is the user interface implemented by the ActOn Button Bar. It consists of a window that displays a list of suggested user actions. It is clear that the number of possible suggestions could quickly become overwhelming to the user. We are currently looking at different ways to adapt the interface to initially show actions that the user is likely to take, but provide a way for the user to see other possible actions as well. It is likely that some other contextual data (physical, emotional) could assist in determining the most desirable actions to offer. We are also looking at different presentation methods for the suggestions, including pop-up hierarchical menus and document lenses.

4.4.3 Resolving conflicts

One of the problems we have found with chaining is the potential for multiple services to generate a data type: a Name object, for example. Since the services are running independently, the Name objects that they generate could be different. If a suggested action to the user is to put this name in the Contact Manager, which Name object should be used? We need to investigate methods for determining relevancy and confidence of suggested actions and results, in order to rank suggestions for the user.

5 Using physical and informational context to support more effective voice-only interaction

5.1 Describing the application

In the Savoir (Somewhat assisted voice-only interaction research) project, we are exploring a number of issues related to context awareness and voice-only computer interactions [7]. As computing and communication devices become smaller, lighter, and more powerful, voice will become a primary medium of interaction. Our work examines voice-only interactions and aims to develop a prototype application that demonstrates useful voice-only functions and well designed voice-only interfaces. We have created a prototype application that allows a user to retrieve information from the Internet using a standard wired or cellular telephone. The user has access to e-mail, US and world news, financial reports, weather forecasts, restaurant listings, and sports scores. We do not currently have access to a voice recognition system robust enough to handle natural spoken interaction, so we have instead developed a Wizard of Oz infrastructure to support our investigation until better speech technology becomes available. We are currently developing a version of Cyberguide using the Savoir infrastructure, adding an interactive tour guide facility to Savoir that utilizes outdoor GPS data. In addition to interpreting a user's voice commands/query, the Savoir version of Cyberguide will receive positioning information to help predict what requests for information a user might make. Figure 3 shows a tourist equipped to use the Savoir Cyberguide interface.

5.2 Use of context in Savoir

Our aim is to enable a user to travel from place to place and to request and receive relevant information based on location and other contextual information (such as time-of-day or user history). Current voice recognition technologies are not adequate to support full vocabulary natural language interactions. In Savoir, we use informational context (noting that requests from a user are likely to be based on information that they have recently heard) to provide the recognition engine with a relevant grammar, thus improving recognition of continuous speech. In the Savoir Cyberguide prototype, we will be able to use physical context to help select a recognition grammar as well. For example, in our campus tour-guide application, if the user is standing in front of the Financial Aid building, we would load a grammar that contains words such as "aid" and "tuition", in addition to the standard grammar. We could also pre-fetch information on the Financial Aid office for the user to access next. In this way, we use physical location to establish context within the application and improve the accuracy of the speech recognizer.

5.3 Mechanisms and architecture

The position information in Savoir is provided by a portable GPS receiver that is connected to a portable computer worn by the user. The portable computer parses the output of the GPS and keeps track of the position state of the user. When the position changes by a large enough distance, the portable computer encodes the current position in a sequence of DTMF tones and sends these tones over the cellular telephone carried by the user. This cellular telephone forms the communication link between mobile user and immobile base station.

Once the positioning data has been sent over the cellular link, the base unit decodes the DTMF tones and makes the position data available to the application. In our initial prototype, the Map Agent, the component that provides tour-guide data to the user, can then locate the nearest feature of interest (be it a building, a park, transportation, etc.) and provide a description to the user via the application's text-to-speech module. This conversion from position information to a feature of interest is exactly the kind of conversion supported by the CyberDesk framework.

5.4 Issues

5.4.1 Maintaining context over time

We have already described how context in Savoir is used to improve the performance of speech recognition based on grammar loading. Since it is unlikely that the grammar loaded will be able to match all possible questions a user will ask, it becomes important to remember the utterances that have not been recognized in the past and even record the place where they occurred. Maintaining this context history may improve the grammar to be loaded the next time a location is visited by the same person.

It is also desirable to maintain parts of the informational context over short periods of time. For instance, a request to get "Dave's" e-mail address could result in the name "Dave" being placed in a short-term context area in order to disambiguate pronouns in later utterances (e.g., "Send these comments to him").

5.4.2 Determining more of the user state

Voice-only interfaces have more of an obligation to achieve a flexibility with respect to user expertise than is the case for graphical interfaces. Analyzing a voice pattern and matching it against features indicative of important user states (e.g., interest or confusion) can help to automatically adjust the form of voice interaction that the system supports, accommodating those that are comfortable with the system and those that are not.

6 Classroom 2000: using context in the classroom

6.1 Describing the application

Our final example application involves the use of ubiquitous computing in education. We are actively engaged in a project, Classroom 2000, which is attempting to augment both teacher and learner in a lecture-room environment [2]. Much of our lives are spent engaged in active discussions with other people. How many times have we been stuck trying to recall the one important message from a meeting only to have that memory elude us? A similar situation arises in some forms of classroom education as well. The purpose of Classroom 2000 is to use automated tools to capture different streams of classroom activities, such as prepared lecture materials, audio, video, and handwritten notes on an electronic surface. The captured material is then integrated together and made accessible via the Web to provide a facsimile of the actual classroom experience.

6.2 Use of context in Classroom 2000

The most obvious context is the time in which different events occur. We provide a lot of synchronization of class materials based on common times (e.g., linking handwritten notes on an electronic whiteboard to an audio track for the lecture). When attending a lecture, it is possible that a student will become lost in the train of thought of the lecturer. By detecting and logging this, the system can provide a pointer into the lecture during the access phase that will bring up the various streams of class activity at the point when confusion began.

As another example, if the majority of the class is confused at some point, then that piece of group context may be useful for the instructor. Students might take advantage of such a facility if anonymity were guaranteed. Frequently, questions arise during a lecture and the lecturer is unwilling to address the question. If a student were able to submit a question at the time in which it occurs to her, the lecturer could return to the question, and the context in which it arose, at a later time. Finally, effective use of context can allow relationships between material across lectures that are related by topic, not time. We are experimenting with speech technology to feed into an audio indexing and concept indexing scheme to support search across notes.

6.3 Mechanisms and architecture

The primary concern of Classroom 2000 has been to use time as a synchronizing quantity, so that lecture notes written by an instructor or student can be automatically connected to the audio or video record of the class at that time. Our challenge so far has been to reduce all contextual actions into equivalent time information. The knowledge of when significant activities occur is useful in providing hooks into other streams of information.

6.4 Issues

6.4.1 The need for implicit detection of context

Context-awareness should be done behind the scenes. In early prototypes of Classroom 2000 we used a graphical interface to allow students to provide anonymous feedback on their reactions to class pace and content. We found that most students did not use these explicit context indicator features because they provided too much of a distraction from the activity of the lecture. We need to find more implicit mechanisms to capture group context.

7 Conclusions

Future computing environments promise to break the paradigm of desktop computing. To do this, computational services need to take advantage of the changing context of the user. The context-awareness we have promoted considers physical, social, informational and even emotional information. Beginning with an applications perspective on future computing environments, we have been able to identify some of the common mechanisms and architectures that best support context-aware computing. We have described applications in the domain of tourism, education and personal information management, and in each case, there is a strong argument in favor of context-aware computing. We have also raised some issues to direct further research in context-aware computing.

Our experimental research method is very heavily biased in favor of an applications focus. We argue, along with Weiser [17], that the primary purpose for mobile, ubiquitous and wearable computing is the applications that are enabled. Though research on infrastructure is also essential, the current balance of research is weighted too much in favor of infrastructure at the expense of applications. We hope that our success at generating a productive research group is enough to encourage other researchers to focus more on applications-level research.

Acknowledgements

The authors would like to acknowledge the active support of all members of the FCE Group for their comments that have lead to the formulation of these general notions of context-aware computing.

References

[1] G. D. Abowd. Ubiquitous computing: Research themes and open issues from an applications perspective. Technical Report GIT-GVU 96-24, GVU Center, Georgia Institute of Technology, October 1996.

[2] G. D. Abowd, C. G. Atkeson, A. Feinstein, C. Hmelo, R. Kooper, S. Long, N. Sawhney, and M. Tan. Teaching and learning as multimedia

authoring: The classroom 2000 project. In *Proceedings of the ACM Conference on Multimedia — Multimedia'96*, 1996.

[3] G. D. Abowd, C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context-aware tour guide. *ACM Wireless Networks*, 1997. To appear.

[4] . Apple Research Laboratories. Apple data detectors homepage. Available at http://www.research.apple.com/research-/tech/AppleDataDetectors/, 1997.

[5] A. Asthana, M. Cravatts, and P. Krzyzanouski. An indoor wireless system for personalized shopping assistance. In L.-F. Cabrera and M. Sattyanarayanan, editors, *Workshop on Mobile Computing Systems and Applications*, pages 69–74. IEEE Computer Society Press, December 1994.

[6] A. Dey, G. D. Abowd, M. Pinkerton, and A. Wood. Cyberdesk: A framework for providing self-integrating ubiquitous software services. Technical Report GIT-GVU-97-10, GVU Center, Georgia Institute of Technology, June 1997.

[7] A. K. Dey, L. D. Catledge, G. D. Abowd, and C. Potts. Developing voice-only applications in the absence of speech recognition technology. Technical Report GIT-GVU-97-06, GVU Center, Georgia Institute of Technology, March 1997. Submitted to DIS'97.

[8] I. Essa and A. Pentland. A vision system for observing and extracting facial action parameters. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 76–83. IEEE Computer Society, 1994.

[9] I. Essa and A. Pentland. Facial expression recognition using a dynamic model and motion energy. In *Proceedings of the International Conference on Computer Vision*, pages 360–367. IEEE Computer Society, Cambridge, MA, 1995.

[10] S. Long, D. Aust, G. D. Abowd, and C. G. Atkeson. Rapid prototyping of mobile context-aware applications: The cyberguide case study. In *Proceedings of the 1996 conference on Human Factors in Computing Systems — CHI'96*, 1996. Short paper.

[11] S. Long, R. Kooper, G. D. Abowd, and C. G. Atkeson. Rapid prototyping of mobile context-aware applications: The cyberguide case study. In *Proceedings of the 2nd Annual International Conference on Mobile Computing and Networking*, November 1996.

[12] A. C. Long, Jr., S. Narayanaswamy, A. Burstein, R. Han, K. Lutz, B. Richards, S. Sheng, R. W. Brodersen, and J. Rabaey. A prototype user interface for a mobile multimedia terminal. In *Proceedings of the 1995 conference on Human Factors in Computing Systems — CHI'95*, 1995. Interactive experience demonstration.

[13] M. Pandit and S. Kalbag. The selection recognition agent: Instant access to relevant information and operations. In *Proceedings of Intelligent User Interfaces '97*. ACM Press, 1997.

[14] R. Picard. Affective computing. Technical Report 321, MIT Media Lab, Perceptual Computing, November 1995. Available as MIT Media Lab Perceptual Computing Techreport # 362 from http://vismod.www.media.mit.edu/vismod/.

[15] W. N. Schilit. *System architecture for context-aware mobile computing*. PhD thesis, Columbia University, 1995.

[16] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, January 1992.

[17] M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7):75–84, July 1993.

[18] A. Wood, A. Dey, and G. D. Abowd. Cyberdesk: Automated integration of desktop and network services. In *Proceedings of the 1997 conference on Human Factors in Computing Systems — CHI'97*, pages 552–553, 1997. Technical note.