The Conference Assistant: Combining Context-Awareness with Wearable Computing

Anind K. Dey, Daniel Salber, Gregory D. Abowd
GVU Center, College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
+1 404 894 7512
{anind, salber, abowd}@cc.gatech.edu

Masayasu Futakawa
Hitachi Research Laboratory
7-1-1 Omika-cho
Hitachi-shi, Ibaraki-ken, 319-1221, Japan
81-294-52-5111
futakawa@hrl.hitachi.co.jp

http://www.cc.gatech.edu/fce/contexttoolkit

Abstract

We describe the Conference Assistant, a prototype mobile, context-aware application that assists conference attendees. We discuss the strong relationship between context-awareness and wearable computing and apply this relationship in the Conference Assistant. The application uses a wide variety of context and enhances user interactions with both the environment and other users. We describe how the application is used and the context-aware architecture on which it is based.

1. Introduction

In human-human interaction, a great deal of information is conveyed without explicit communication, but rather by using cues. These shared cues, or *context*, help to facilitate grounding between participants in an interaction [3]. We define context to be any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or physical or computational object.

In human–computer interaction, there is very little shared context between the human and the computer. Context in human-computer interaction includes any relevant information about the entities in the interaction between the user and computer, including the user and computer themselves. By improving computers' access to context, we increase the richness of communication in human-computer interaction and make it possible to produce more useful computational services. We define applications that use context to provide task-relevant information and/or services to a user to be *context-aware*.

Context rapidly changes in situations where the user is mobile. The changing context can be used to adapt the user interface to an application, providing relevant services and information to the user. While context is important to mobile computing in general, it is of particular interest to wearable computing. This is evident from the number of papers dealing with context-awareness in the previous Symposiums on Wearable Computers.

Rhodes [13] presented a list of defining characteristics for wearable computers. In each of these features, context plays an important role.

Portable while operational: A wearable computer is capable of being used while the user is mobile. When a user is mobile, her context is much more dynamic. She is moving through new physical spaces, encountering new objects and people. The services and information she requires will change based on these new entities.

Hands-free use: A wearable computer is intended to be operated with the minimal use of hands, relying on speech input or one-handed chording-keyboards and joysticks. Limiting the use of traditional input mechanisms (and somewhat limiting the use of explicit input) increases the need to obtain implicitly sensed contextual information.

Sensors: To enhance the explicit user input, a wearable computer should use sensors to collect information about the user's surrounding environment. Rhodes intended that the sensors be worn on the body, but the real goal is for the sensed information to be available to the wearable computer. This means that sensors can not only be on the body, but also be in the environment, as long as the wearable computer has a method for obtaining the sensed environmental information.

Proactive: A wearable computer should be acting on its user's behalf even when the user is not explicitly using it. This is the essence of context-aware computing: the computer analyzes the user's context and makes task-relevant information and services available to the user, interrupting the user when appropriate.

Always on: A wearable computer is always on. This is important for context-aware computing because the wearable computer should be continuously monitoring the user's situation or context so that it can adapt and respond appropriately. It is able to provide useful services to the user at any time.

From a survey on context-aware computing [6], we found that most context-aware applications use a minimal variety of context. In general, the use of context is limited to only identity and location, neglecting both time and activity. Complex context-aware applications are difficult to build. By complex, we mean applications that not only deal with a wide variety of context, but also that take into account the contexts of multiple people or entities, real-time context as well as historical context, that use a single piece of context for multiple purposes, and that support interactions between multiple users, mobile and wearable computers and the environment. This family of applications is hard to implement because there has been little support for thinking about and designing them.

This paper describes the design of a complex context-aware application that addresses these issues. We will present the Conference Assistant, a context-aware application for assisting conference attendees and presenters. We demonstrate how context is used to aid users and describe how the application was built. In particular, we will present some concepts that make it easier to design complex context-aware applications.

2. The Conference Assistant

In this section, we will present a complex prototype application, the Conference Assistant, which addresses the deficiencies we pointed out in previous context-aware applications. The Conference Assistant is a context-aware application intended for use by conference attendees. We will describe the conference domain and show why it is appropriate for context-awareness and wearable computing and provide a scenario of use. We will then discuss the context used in this application and how it was used to provide the user benefit. We will end with a discussion on the types of context-aware features the Conference Assistant supports.

2.1. Conference Domain

The Conference Assistant was designed to assist people attending a conference. We chose the conference domain because conferences are very dynamic and involve an interesting variety of context. A conference attendee is likely to have similar interests as other attendees. There is a great deal of concurrent activity at large conferences including paper presentations, demonstrations, special interest group meetings, etc., at which a large amount of information is presented. We built the Conference Assistant to help users decide which activities to attend, to provide awareness of the activities of colleagues, to enhance interactions between users and the environment, to assist users in taking notes on presentations and to aid in the retrieval of conference information after the conference concludes.

A wearable computer is very appropriate for this application. The Conference Assistant uses a wide variety of context: time, identity, location, and activity. It promotes interaction between simultaneous users of the application and has a large degree of interaction with the user's surrounding environment. Revisiting Rhodes' list of wearable computer characteristics, we can show how the domain is applicable for wearable computing.

Portable while operational: During a conference, a user is mobile, moving between presentation and demonstration spaces, with rapidly changing context.

Hands-free use: During a conference, hands should be free to take notes, rather than interacting with a computer to collect context information.

Sensors: Sensors in the environment can provide useful information about the conference to the user, including presentation information and activities of colleagues.

Proactive and Always on: In a conference, a user wants to pay attention to what is being presented while maintaining an awareness of other activities. A wearable computer can provide this awareness without explicit user requests.

2.2. User Scenario

Now that we have demonstrated the utility of context-awareness and wearable computing in the conference domain, we will present a user scenario for the Conference Assistant. A user is attending a conference. When she arrives at the conference, she registers, providing her contact information (mailing address, phone number, and email address), a list of research interests, and a list of colleagues who are also attending the conference. In return, she receives a copy of the conference proceedings and an application, the Conference Assistant, to run on her wearable computer. When she starts the application, it automatically displays a copy of the conference schedule, showing the multiple tracks of the conference, including both paper tracks and demonstration tracks. On the schedule (Figure 1), certain papers and demonstrations are highlighted (light gray) to indicate that they may be of particular interest to the user.

Figure 1. Screenshot of the augmented schedule, with suggested papers and demos highlighted (light-colored boxes) in the three (horizontal) tracks.

The user takes the advice of the application and walks towards the room of a suggested paper presentation. When she enters the room, the Conference Assistant automatically displays the name of the presenter and the title of the presentation. It also indicates whether audio and/or video of the presentation are being recorded. This impacts the user's behavior, taking fewer or greater notes depending on the extent of the recording available. The presenter is using a combination of PowerPoint and Web pages for his presentation. A thumbnail of the current slide or Web page is displayed on the wearable computer display. The Conference Assistant allows the user to create notes of her own to "attach" to the current slide or Web page (Figure 2). As the presentation proceeds, the application displays updated slide or Web page information. The user takes notes on the presented information using the Conference Assistant. The presentation ends and the presenter opens the floor for questions. The user has a question about the presenter's tenth slide. She uses the application to control the presenter's display, bringing up the tenth slide, allowing everyone in the room to view the slide in question. She uses the displayed slide as a reference and asks her question. She adds her notes on the answer to her previous notes on this slide.

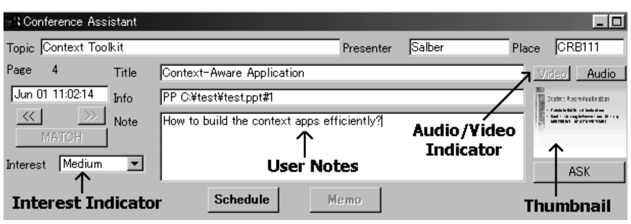

Figure 2. Screenshot of the Conference Assistant note-taking interface.

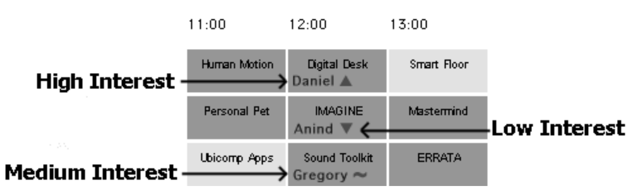

Figure 3. Screenshot of the partial schedule showing the location and interest level of colleagues. Symbols indicate interest level.

After the presentation, the user looks back at the conference schedule display and notices that the Conference Assistant has suggested a demonstration to see based on her interests. She walks to the room where the demonstrations are being held. As she walks past demonstrations in search of the one she is interested in, the application displays the name of each demonstrator and the corresponding demonstration. She arrives at the demonstration she is interested in. The application displays any PowerPoint slides or Web pages that the demonstrator uses during the demonstration. The demonstration turns out not to be relevant to the user and she indicates her level of interest to the application. She looks at the conference schedule and notices that her colleagues are in other presentations (Figure 3). A colleague has indicated a high level of interest in a particular presentation, so she decides to leave the current demonstration and to attend this presentation. The user continues to use the Conference Assistant throughout the conference for taking notes on both demonstrations and paper presentations.

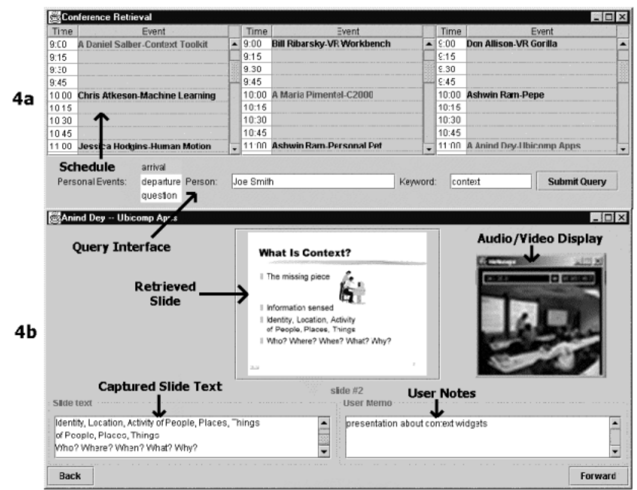

Figure 4. Screenshots of the retrieval application: query interface and timeline annotated with events (4a) and captured slideshow and recorded audio/video (4b).

She returns home after the conference and wants to retrieve some information about a particular presentation. The user executes a retrieval application provided by the conference. The application shows her a timeline of the conference schedule with the presentation and demonstration tracks (Figure 4a). The application uses a feature known as context-based retrieval [9]. It provides a query interface that allows the user to populate the timeline with various events: her arrival and departure from different rooms, when she asked a question, when other people asked questions or were present, when a presentation used a particular keyword, or when audio or video were recorded. By selecting an event on the timeline (Figure 4a), the user can view (Figure 4b) the slide or Web page presented at the time of the event, audio and/or video recorded during the presentation of the slide, and any personal notes she may have taken on the presented information. She can then continue to view the current presentation, moving back and forth between the presented slides and Web pages.

In a similar fashion, a presenter can use a third application to retrieve information about his/her presentation. The application displays a timeline of the presentation, populated with events about when different slides were presented, when audience members arrived and left the presentation (and their identities), the identities of audience members who asked questions and the slides relevant to the questions. The interface is similar to that shown in figure 4. The presenter can 'relive' the presentation, by playing back the audio and/or video, and moving between presentation slides and Web pages.

2.3. Use of Context

The Conference Assistant uses a wide variety of context to provide both services and information to users. We will first describe the context used in real time to assist a conference attendee during a conference and then will describe the historical context used after the conference by a conference attendee and a presenter.

When the user is attending the conference, the application first uses information about what is being presented at the conference and her personal interests to determine what presentations might be of particular interest to her. The application uses her location, the activity (presentation of a Web page or slide) in that location and the presentation details (presenter, presentation title, whether audio/video is being recorded) to determine what information to present to her. The text from the slides is being saved for the user, allowing her to concentrate on what is being said rather than spending time copying down the slides. The context of the presentation (presentation activity has concluded, and the number and title of the slide in question) facilitates the user's asking of a question. The context is used to control the presenter's display, changing to a particular slide for which the user had a question.

The list of colleagues provided during registration allows the application to present other relevant information to the user. This includes both the locations of colleagues and their interest levels in the presentations they are currently viewing. This information is used for two purposes during a conference. First, knowing where other colleagues are helps an attendee decide which presentations to see herself. For example, if there are two interesting presentations occurring simultaneously, knowing that a colleague is attending one of the presentations and can provide information about it later, a user can choose to attend the other presentation. Second, as described in the user scenario, when a user is attending a presentation that is not relevant or interesting to her, she can use the context of her colleagues to decide which presentation to move to. This is a form of social or collaborative information filtering [15].

After the conference, the retrieval application uses the conference context to retrieve information about the conference. The context includes public context such as the time when presentations started and stopped, whether audio/video was captured at each presentation, the names of the presenters, the presentations and the rooms in which the presentations occurred and any keywords the presentations mentioned. It also includes the user's personal context such as the times at which she entered and exited a room, the rooms themselves, when she asked a question and what presentation and slide or Web page the question was about. The application also uses the context of other people, including their presence at particular presentations and questions they asked, if any. The user can use any of this context information to retrieve the appropriate slide or Web page and any recorded audio/video associated with the context.

After the conference, a presenter can also use the conference context to obtain information relevant to his/her presentation. The presenter can obtain information about who was present for the presentation, the times at which each slide or Web page was visited, who asked questions and about which slides. Using this information, along with the text captured from each slide and any audio/video recorded, the presenter can playback the entire presentation and question session.

2.4. Context-aware Features

Pascoe [11] introduced a set of four context-aware capabilities that applications can support. We will present each capability and will show how the Conference Assistant supports each one.

Contextual sensing: A system detects context and simply presents it to the user, augmenting the user's sensory system. The Conference Assistant presents the user's current location, name of the current presentation and presenter, location of colleagues, and colleagues' level of interest in their presentations.

Contextual adaptation: A system uses context to adapt its behavior instead of providing a uniform interface in all situations. When a new presentation slide or Web page is presented, the Conference Assistant saves the user's notes from the previous slide and creates an empty textbox in which notes on the new current slide can be entered.

Contextual resource discovery: A system can locate and use resources that share part or all of its context. When a user enters a presentation/demonstration area, the Conference Assistant creates a temporary bind to the presentation server in the local environment. The shared context is location. This binding allows the application to obtain changes to the local presentation/demonstration.

Contextual augmentation: A system augments the environment with additional information, associating digital data with the current context. All the notes that a user makes on presented information are augmented with contextual information (location, presentation title and presenter, and time). This augmentation supports retrieval of the notes using context-based retrieval techniques.

The Conference Assistant exploits all four of the context-aware capabilities presented by Pascoe. These capabilities are used to provide substantial benefits to both conference attendees and presenters.

3. Application Design

In this section, we describe the design of the Conference Assistant. We illustrate the software architecture of the application, as well as the context-aware architecture it was built on top of. We discuss the concepts the architecture supports that make it easier to build and evolve context-aware applications. Finally, we also describe the hardware used to deploy the application.

3.1. Software

The Conference Assistant is a complex context-aware application. It uses a wide variety of context, supporting both interaction between a single user and the environment and between multiple users. This application would have been difficult to build without a great deal of underlying support. It was built on top of an architecture designed to support context-aware applications [5]. We will first briefly describe this architecture and then will show how the architecture was used to build the Conference Assistant.

3.1.1. Context Architecture[1]

In previous work [5,14], we presented an architecture and toolkit that we designed and implemented to support building of context-aware applications. We will briefly discuss the components of the architecture and its merits.

The architecture consists of three types of components: widgets, servers, and interpreters. They implement the concepts necessary for easing the development of context-aware applications. Figure 5 shows the relationship between the context components and applications.

[1] For more information on the context architecture, see http://www.cc.gatech.edu/fce/contexttoolkit

Context widgets encapsulate information about a single piece of context, such as location or activity, for example. They provide a uniform interface to components or applications that use the context, hiding the details of the underlying context-sensing mechanism(s).

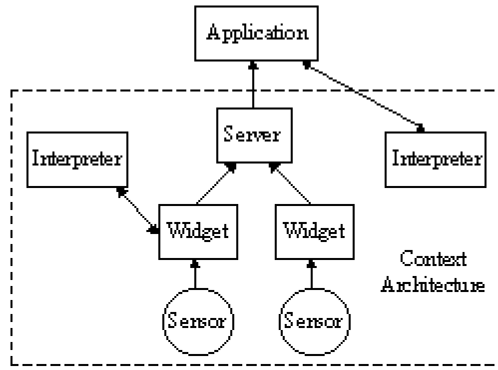

Figure 5. Relationship between applications and the context architecture. Arrows indicate data flow.

A context server is very similar to a widget, in that it supports the same set of features as a widget. The difference is that a server aggregates multiple pieces of context. In fact, it is responsible for all the context about a particular entity (person, place, or object). Aggregation facilitates the access of context by applications that are interested in multiple pieces of context about a single entity.

A context interpreter is used to abstract or interpret context. For example, a context widget may provide location context in the form of latitude and longitude, but an application may require the location in the form of a street name. A context interpreter may be used to provide this abstraction.

Context components are instantiated and executed independently of each other in separate threads and on separate computing devices. The architecture makes the distribution of the context architecture transparent to context-aware applications, mediating all communications between applications and components. It supports communications using the HyperText Transfer Protocol (HTTP) for both the sending and receiving of messages. The language used for sending data is the eXtensible Markup Language (XML). XML and HTTP were chosen because they support lightweight integration of distributed components and facilitate access to the architecture on heterogeneous platforms with multiple programming languages. The only requirements on devices using the architecture are that they support ASCII parsing and TCP/IP. These minimal requirements are particularly important for mobile and wearable computers, for which communications support tends to be small.

The context architecture promotes three main concepts for building context-aware applications: separation of context sensing from context use, aggregation, and abstraction. It relieves application developers from having to deal with how to sense and access context information, and instead, concentrate on how to use the context. It provides simplifying abstractions like aggregation and abstraction to make it easier for applications to obtain the context they require. Aggregation provides “one-stop shopping” for context about an entity, allowing application designers to think in terms of high level information, rather than low-level details. The architecture makes it easy to add the use of context to existing applications that don’t use context and to evolve applications that already use context. In addition, the architecture makes context-aware applications resistant to changes in the context-sensing layer. It encapsulates changes and the impact of changes, so applications do not need to be modified.

3.1.2. Software Design

The Conference Assistant was built using the context architecture just described. Table 1 lists all the context components used and Figure 6 presents a snapshot of the architecture when a user is attending a conference.

During registration, a User Server is created for the user. It is responsible for aggregating all the context information about the user and acts as the application’s interface to the user’s personal context information. It subscribes to information about the user from the public Registration Widget, the user’s Memo Widget and the Location Widget in each presentation space. The Memo Widget captures the user’s notes and also any relevant context (relevant slide, time, and presenter identity).

Table 1. Architecture components and responsibilities: S = Servers, W = Widgets, I = Interpreters

Component	Responsibility
Registration (W)	Acquires contact info, interests, and colleagues
Memo (W)	Acquires user’s notes and relevant presentation info
Recommender(I)	Locates interesting presentations
User (S)	Aggregates all information about user
Question (W)	Acquires audience questions and relevant presentation info
Location (W)	Acquires arrivals/departures of users
Content (W)	Monitors PowerPoint or Web page presentation, capturing content changes
Recording (W)	Detects whether audio/video is recorded
Presentation (S)	All information about a presentation

There is a Presentation Server for each physical location where presentations/demos are occurring. A Presentation Server is responsible for aggregating all the context information about the local presentation and acts as the application’s interface to the public presentation information. It subscribes to the widgets in the local environment, including the Content Widget, Location Widget, Recording Widget and Question Widget.

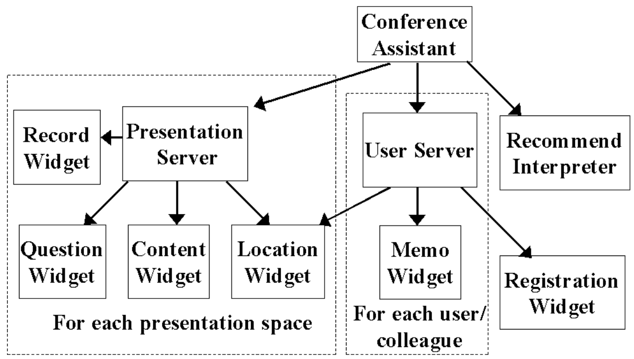

Figure 6. Conference Assistant capture architecture.

When an audience member asks a question using the Conference Assistant, the Question Widget captures the context (relevant slide, location, time, and audience member identity) and notifies the local Presentation Server of the event. The server stores the information and also uses it to access a service provided by the Content Widget, displaying the slide or Web page relevant to the question.

The Conference Assistant does not communicate with any widget directly, but instead communicates only with the user’s User Server, the User Servers belonging to each colleague and the local Presentation Server. It subscribes to the user’s User Server for changes in location and interests. It subscribes to the colleagues’ User Servers for changes in location and interest level. It also subscribes to the local Presentation Server for changes in a presentation slide or Web page when the user enters a presentation space and unsubscribes when the user leaves.

In the conference attendee’s retrieval application, all the necessary information has been stored in the user’s User Server and the public Presentation Servers. The architecture for this application is much simpler, with the retrieval application only communicating with the user’s User Server and each Presentation Server. As shown in Figure 4, the application allows the user to retrieve slides (and the entire presentation including any audio/vide) using context via a query interface. If personal context is used as the index into the conference information, the application polls the User Server for the times and location at which a particular event occurred (user entered or left a location, or asked a question). This information can then be used to poll the correct Presentation Server for the related presentation information. If public context is used as the index, the application polls all the Presentation Servers for the times at which a particular event occurred (use of a keyword, presence or question by a certain person). As in the previous case, this information is then used to poll the relevant Presentation Servers for the related presentation information.

In the presenter's retrieval application, all the necessary information has been stored in the public Presentation Server used during the relevant presentation. The architecture for this application is simple as well, with the retrieval application only communicating with the relevant Presentation Server. As shown in Figure 4, the application allows the user to replay the entire presentation and question session, or view particular points in the presentation using context-based retrieval. Context includes the arrival and departure of particular audience members, transitions between slides and/or Web pages, and when questions were asked and by whom.

3.2. Hardware

The Conference Assistant application is being executed on a variety of different platforms, including laptops running Windows 95/98 and Hewlett Packard 620LX WinCE devices. It was not actually run on a wearable computer, but there is no reason why it could not be. The only requirements are constant network access and a graphical display. For communications with the context architecture, we use Proxim's RangeLAN2 1.6 Mbps wireless LAN for WinCE devices and RadioLan's 10BaseRadio 10 Mbps wireless LAN.

The retrieval applications are running on desktop machines, under both the Windows 95/98 and Solaris operating systems.

The context components were executed on a number of different computers running different operating systems. This includes Powermac G3's running MacOS, Intel Pentiums running Windows 95 and Windows NT, and SPARC 10s running Solaris.

To sense identity and location of the conference attendees and presenters, we use PinPoint Corporation's 3D-iD™ Local Positioning System. This system uses radio frequency-based (RF) tags with unique identities and multiple antennas to locate users in a physical space. Although, it can provide location information at the resolution of 6 feet, we used coarser-grained information to determine when users entered a room.

4. Related Work

In this section, we will discuss other work that is relevant to the Conference Assistant, in the areas of conference assistants, context-aware tour guides, note taking, and context-based retrieval.

There has been little work in the area of context-awareness in a conference setting [10]. In the Mobile Assistant Project, Nishibe *et al*. deployed 100 handheld computers with cell phones at the International Conference on Multiagent Systems (ICMAS '96). The system provided conference schedule and tourist information. Social filtering, using the queries of other conference attendees, was used to determine relevant tourist information. The system supported community activity by allowing attendees to search for others with similar interests. Context was limited to "virtual information" such as personal interests, not taking into account "real information" like location.

Somewhat similar to a conference assistant is a tour guide. Both applications provide relevant information about the user's current context. The context-aware tour guide application is, perhaps, the canonical context-aware application. It has been the focus of much effort by groups doing context-aware research. Feiner *et al.* developed a tour guide for the Columbia University campus that combined augmented reality with mobile computing [7]. Fels *et al*. and Long *et al*. built tour guide applications for visitors attending an open house at their respective laboratories [8,2]. These systems use static configurations and can not deal with changes to tours at runtime. In contrast, the context aware architecture used in the Conference Assistant is able to make runtime changes transparent to the application.

There have been a number of systems that support individual users in taking notes on presentations. The Classroom 2000 project used an augmented classroom that captured audio, video, web slides visited and whiteboard activity to make the student notetaking activity easier [1]. The NotePals system aggregated the notes from several notetakers to provide a single group record of a presentation [4]. Stifelman built an augmented paper notebook that allowed access to the audio of a presentation during review [17]. The context most extensively used in these applications is time. The Conference Assistant expands the range of context used.

One of the most important projects in context-based retrieval was the Forget-me-not system from Lamming and Flynn [9]. It kept a record of a person's activity throughout the day in a diary format, allowing retrieval of the activity information based on context. Rhodes' wearable Remembrance Agent used context information about notes a user wrote, such as co-located people, location, and time to allow automatic retrieval of those notes that most closely matched the user's current context [13]. Rekimoto *et al*. used an augmented reality system to attach notes to objects or locations [12]. When users approached those objects or locations, the note was retrieved. This is similar to the Locust Swarm project by Starner *et al* [16], which allowed the attachment and retrieval of notes from infrared-based location tags.

5. Conclusions and Future Application Work

We have presented the Conference Assistant, a prototype mobile, context-aware application for assisting conference attendees in choosing presentations to attend, taking notes, and retrieving those notes. We discussed the important relationship between context-awareness and wearable computing. We demonstrated this relationship in the Conference Assistant. We showed how the Conference Assistant made use of a wide variety of context, both personal and environmental, and how it enhanced user interactions with both the environment and other users. We discussed the important concepts that our architecture supports, that make it easier to build and modify complex context-aware applications: separation of sensing and using context, aggregation, and abstraction.

The Conference Assistant is currently a prototype application running in our laboratory. We would like to deploy the application at an actual conference. This would require us to provide many handheld devices (in case the conference attendees do not have their own wearable computers), a wireless LAN, and an indoor positioning system. This would allow us to perform a realistic evaluation of the application.

There are also additional features that we would like to add to the Conference Assistant. The first is the addition of an improved assistant for demonstrations. Currently, the application doesn't treat paper presentations any differently from demonstrations. We would like to enhance the application when a demonstration is being given, by providing additional information about the demonstration. This includes relevant web pages, research papers, and videos.

Currently, the Conference Assistant only uses information about PowerPoint slides and Web pages being presented. We would like to extend this to use other presentation packages and mediums. This will require no change to the application, but will require the development of additional context widgets to capture the presentations and relevant updates.

Other features to add deal with access to information about the user's colleagues. Presently, at registration, users indicate the colleagues about whom they would like to receive information on. This is actually the opposite of how we should be approaching this problem, from a privacy point of view. At registration, users should actually indicate who is allowed to access their information (location and level of interest). This allows users to manage their own information. A related feature is to allow users to access their colleagues' notes with the retrieval application. This would provide additional information that would augment the user's own notes on a presentation and would be a source of notes for presentations that the user did not attend.

A final feature to add to the Context Assistant is an interface that supports serendipitous information retrieval relevant to the current presentation, much like the Remembrance Agent [13]. Potential information to retrieve includes conference and field-relevant information.

We would like to add a third retrieval application for conference organizers. This application would allow them to view anonymized information about the number of people at various presentations and demonstrations and the average amount of time attendees spent at each.

Acknowledgements

We would like to acknowledge the support of the Future Computing Environments research group for contributing to the ideas of the Conference Assistant. We would like to thank Thad Starner for his comments on this work. This work was supported in part by an NSF CAREER Grant # 9703384, NSF ESS Grant EIA-9806822, a Motorola UPR and a Hitachi grant.

References

[1] G.D. Abowd *et al.*, "Investigating the capture, integration and access problem of ubiquitous computing in an educational setting", in Proceedings of CHI'98, April 1998, pp. 440-447.

[2] 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*, 3(5), 1997, pp. 421-433.

[3] H.H. Clark and S.E. Brennan, "Grounding in communication", in L.B. Resnick, J. Levine, & S.D. Teasley (Eds.), *Perspectives on socially shared cognition*. Washington, DC. 1991.

[4] R. Davis *et al.*, "NotePals: Lightweight note sharing by the group, for the group", in Proceedings of CHI'99, May 1999, pp. 338-345.

[5] A.K. Dey, D. Salber, M. Futakawa and G. Abowd, "An architecture to support context-aware applications", submitted to UIST '99.

[6] A.K. Dey and G.D. Abowd, "Towards an understanding of context and context-awareness", submitted to HUC '99.

[7] S. Feiner, B. MacIntyre, T. Hollerer and A. Webster, "A Touring Machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment", in Proceedings of the 1st International Symposium on Wearable Computers, October 1997, pp. 74-81.

[8] S. Fels *et al.*, "Progress of C-MAP: A context-aware mobile assistant", in Proceeding of AAAI 1998 Spring Symposium on Intelligent Environments, Technical Report SS-98-02, March 1998, pp. 60-67.

[9] M. Lamming and M. Flynn, "Forget-me-not: Intimate computing in support of human memory", in Proceedings of FRIEND21: International Symposium on Next Generation Human Interfaces, 1994, pp. 125-128.

[10] Y. Nishibe *et al.*, "Mobile digital assistants for community support", *AAAI Magazine 19(2)*, Summer 1998, pp. 31-49.

[11] J. Pascoe, “Adding generic contextual capabilities to wearable computers”, in Proceedings of 2nd International Symposium on Wearable Computers, October 1998, pp. 92-99.

[12] J. Rekimoto, Y. Ayatsuka and K. Hayashi, “Augment-able reality: Situated communications through physical and digital spaces”, in Proceedings of the 2nd International Symposium on Wearable Computers, October 1998, pp. 68-75.

[13] B. Rhodes, “The Wearable Remembrance Agent: A system for augmented memory”, in Proceedings of the 1st International Symposium on Wearable Computers, October 1997, pp. 123-128.

[14] D. Salber, A.K. Dey and G.D. Abowd, “The Context Toolkit: Aiding the development of context-enabled applications”, in Proceedings of CHI’99, pp. 434-441.

[15] U. Shardanand and P. Maes, "Social information filtering: algorithms for automating 'word of mouth'", in Proceedings of CHI’95, May 1995, pp. 210-217.

[16] T. Starner, D. Kirsch and S. Assefa, “The Locust Swarm: An environmentally-powered, networkless location and messaging system”, in Proceedings of the 1st International Symposium on Wearable Computers, October 1997, pp. 169-170.

[17] L.J. Stifelman, “Augmenting real-world objects: A paper-based audio notebook”, in Proceedings of CHI’96, April 1996, pp. 199-200.