Towards Context Aware Computing: Experiences and Lessons

Asim Smailagic, Daniel P. Siewiorek, Joshua Anhalt, Francine Gemperle
Carnegie Mellon University
Daniel Salber, Sam Weber
IBM T.J. Watson Research Center

Abstract— This paper defines an activity/attention framework for context aware computing, and categorizes several applications we have developed in spatial and temporal contexts. These context aware applications utilize the services from the activity/attention framework. The paper also introduces a generic architecture for pervasive computing, explores and refines its design space, and describes specific instantiations of the architecture and provides initial evaluation of these applications. Two different user interfaces, visual and audio, have been implemented and contrasted for the same application.

Index Terms—Context aware computing, pervasive computing, mobile computing, intelligent agent, location awareness, multi-modal user interface, privacy

1 INTRODUCTION

The effects of Moore's Law are apparent everywhere: chip density, processor speed, memory cost, disk capacity and network bandwidth are improving relentlessly. As the cost of computing drops significantly, a resource that has been ignored until now becomes the limiting factor in computer systems — user attention, namely the ability of a person to focus on their primary task.

Distractions occur especially in mobile environments, since the user is often preoccupied with walking, driving or other real-world interactions. A pervasive computing environment that minimizes distraction has to be context aware. The system needs to know the state of the user to accommodate their needs.

This paper proposes that context aware applications are built upon at least two fundamental services, spatial awareness and temporal awareness. Spatial awareness includes the relative and absolute position and orientation of a user. Temporal awareness includes the scheduled time of public and private events.

An activity/attention matrix is introduced in Section 2 and illustrated by the example deployed applications.

- Portable Help Desk (PHD)
- Matchmaker

To facilitate the creation of these context aware applications, two primary services were developed

- Location Service
- Schedule Service

Derived services, based upon the primary services, were created to address the issue of privacy of location information and to enable intelligent management of temporal and spatial information.

- Privacy Guard
- Context Aware Agents

Two different user interfaces, visual and audio, have been implemented and contrasted for the same application (PHD).

The applications and services fitting the Activity/Attention framework were primarily developed to facilitate collaborative design. An interdisciplinary class of students at CMU undertook the development of the applications and services. The students identified services that, based on personal experience, would facilitate the design process. The first application, PHD, enables the ability to instantaneously locate team members on campus. Students have a large number of meetings held at various times and various places. They are often confused about where the next group meeting is to be held. The ability to observe the teams' members' geographical location on campus enables them to determine where the meeting is and when a member who is late to the meeting might arrive. The PHD application is detailed in Section 2.1.1. The ability to connect a user's query with an expert is captured in the Matchmaker system, which employs both temporal and spatial awareness, as described in Section 2.1.2. Privacy Guard was developed to enable the user to control their location information and it is described in Section 2.1.3. Context Aware Agents employ temporal context information to proactively inform the user of relevant information, as described in Section 2.1.4. Location service has been developed as a primary service, utilizes the wireless network infrastructure to determine user location, and is described in Section 3.3. The Carnegie Mellon Campus has been equipped with 400 wireless networking access points, enabling wireless coverage for the entire campus, indoors and outdoors.

Mobile computing poses difficult challenges such as intermittent and variable-bandwidth connectivity, and client resource constraints imposed by weight and size considerations. The software architecture developed in this project is based on a pervasive software architecture proposed by the IBM T.J. Watson Research Center.

The prototype developed by the class uses Hewlett Packard Jornada 680 palmtop computers and the Itsy/Cue wearable computers communicating via Lucent Wavelan cards on Wireless Andrew. [7]

The relevant work in Humanistic Intelligence and context aware computing includes the following. Steve Mann introduced the notion of Humanistic Intelligence, proposing it as a new signal-processing framework in which the processing apparatus is supportive and dependant on the user's natural capabilities of human body and mind [5][6]. Abowd et. al. have designed a software architecture to enable the creation of context-aware applications [1][2]. Starner et. al. have developed context aware user interfaces through environment looking cameras and machine vision techniques [9]. Laerhoven and Cakmakci utilize body-mounted sensor to determine a users activity and infer their context [4]. Kortuem et. al. describe a wearable computer which alters its user interface based on the devices and services in the user's environment [3].

2 Approach

The Distraction Matrix, shown in Figure 1, categorizes activities by the amount of attention they require. The activities are Information, Communication and Creation. Individual activities are categorized by the amount of distraction they introduce in units of increasing time: Snap, Pause, Tangent and Extended. The Snap duration is an activity that is usually completed in a few seconds, such as checking your watch for the time. The user should not have to interrupt their primary activity to perform this activity. The Pause action requires the user to stop their current activity, switch to the new but related activity, and then return to their previous task within a few minutes. Pulling over to the side of the road and checking directions is an example of a pause. A Tangent action is a medium length task that is unrelated to the action that the user is engaged in. Receiving an unrelated phone call is an example of a tangent activity. An Extended action is when the user deliberately switches their task, beginning a wholly new long-term activity. For the car driver, stopping at a motel and resting for the night is an extended activity.

As distractions on the left of the matrix take less time from the user's primary activity, our intent is to move activities of the matrix towards the left side (Snap). Our goal is to evaluate how this process extends to a larger sample of applications.

2.1 Applications

A set of interactive applications to support mobile team design activities has been implemented. To complement each other while reducing user distraction, these applications address a related set of activities within the Distraction Matrix.

2.1.1 Portable Help Desk

The Portable Help Desk, or PHD, provides quick information retrieval. This tool allows a mobile user to build maps of their immediate area, including static and dynamic resources and the location of their colleagues, contact information and resources availability. While tracking a colleague, their contact information is displayed. Printer queues, restaurant hours and stock of carbonated beverages and food in connected vending machines can be displayed. The PHD application is a spatially aware system. Figure 2 shows the activities supported by the PHD system and the attention each demands of the user.

We have built both visual and audio interfaces for the PHD application. Each interface supports users in different contexts. A user who is walking around is less distracted by the hands-free speech interface, while a stationary user may use the richer visual interface.

Figure 2 illustrates a visual user interface for the PHD application. People and resources are selected in the left pane, the results of the queries are presented in the middle pane while locations of people and resources are displayed in the right pane. The speech interface is free of the distraction of a visual interface enabling the user to walk about and retrieve information that should not be displayed in a public location. Speech-PHD utilities the same database of information, so all responses are formatted in similar manners. Figure 5 is a transcript of the same queries that were made in Figure 4. This system is aware of the user's current location, allowing it to answer questions such as "Where is the nearest ATM?"

The visionary scenario of the visual and audio-based PHD systems is to deliver relevant information to the user in both proactive and user driven manners. Proactive information is delivered to a user when they are engaging infrastructure resources such as printers. When a user begins a print job, PHD will alert the user to a large print queue and suggest a nearby printer with a shorter print queue. PHD can suggest a printer near the destination of a user in route. Using respond to queries such as "Is Fred coming to the meeting?"

A design group waiting for a team member can use PHD to locate their missing colleague and estimate their time of arrival. The group has access to the user's phone numbers. This helps avoid repeating the beginning of a meeting for every late member. When the team is getting hungry, they can lookup the hours of nearby restaurants find an ATM or just check if the coke machine is full.

2.1.2 Matchmaker

The Matchmaker is an application that helps a user with a question rapidly identify an expert user with the knowledge to help solve an unexpected problem. The suitability of an

Time

			Snap	Pause	Tangent	Extended
Information						
	active	• Receiving • Notifying • Monitoring • Serendipity	– Message arrival – Information accessible – Auction – Stocks, Sports, Matching similar needs – Free food			– Audio, Walkman – Transferring files from network – Reading news
		• Seeking	– Line length – Bus arrival – Locate person	– Exam calendar – Software/hardware help – Calendaring – Navigation	– Looking for Class Notes – Who else is doing this now? – Access personal data	
	passive	• Browsing • Finding		– Information on web or built environment	– Poster, bulletin board information	– Web Research – Reviewing Class Notes
		• Verifying		– Recall previous queries – Double checking information		
Communication						
	artificial	• Initiating	– S.O.S. Emergency	– Introductions	– Team building – Collaborative work	– Chatting (public or private)
		• Participating	– Instant messaging	– Queries	– Event planning – Assassins game – Social Planning	
	informal	• Broadcasting		– Information exchange – Scheduling	– Posting information to bulletin board – Advertising	
	formal	• One to One communications with an individual • One to Group communications with select group, team or family • One to All Possible broadcast communications with unknown people				
Creation						
	Work	• Recording	– Remember this! – Add a todo or call list		– Class note taking – Meeting	– Generating messages
		• Synthesizing		– Forwarding x to y	– Filling out survey – Registration	– Summarizing lecture
		• Generating			– New ideas – Adding information to existing projects	– Mobile tool building

Figure 1—Distraction Matrix

Time

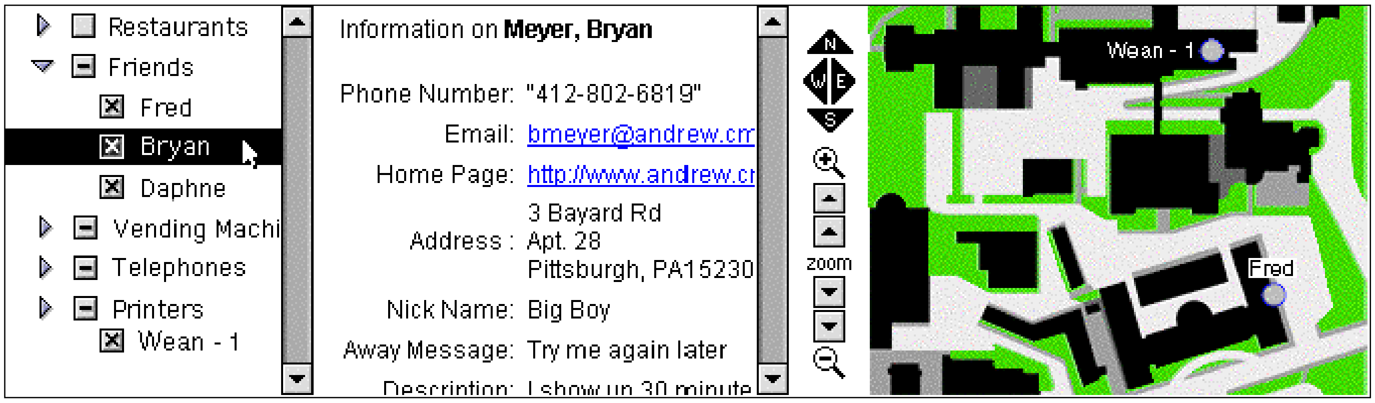
		Snap	Pause	Tangent	Extended
Information					
Active	• Receiving • Notifying • Monitoring • Serendipity • Seeking	– Receive request for help – Completion of task notification	– User searches through list of possible solutions returned by system – Finding someone to help	– User tries suggested solutions	
Passive	• Browsing • Finding • Verifying				
Communication					
Inform Artificial	• Initiating • Participating • Broadcasting				– Expert and user collaborate
Formal	• One to One – communications with an individual • One to Group – communications with select group, team or family • One to All Possible – broadcast communications with unknown people				
Creation					
ork	• Recording • Synthesizing •		– User initiates query		

Figure 2—Distraction Matrix for Portable Help Desk

Time

		Snap	Pause	Tangent	Extended
Information					
Active	• Receiving • Notifying • Monitoring • Serendipity • Seeking	– "Fred is coming" – Print queue alert	– Seek Resources	– Accessing personal data	
Passive	• Browsing • Finding • Verifying	– Check print queue	– Find Person		
Communication					
Inform rtifici	• Initiating • Participating • Broadcasting				
Formal	• One to One – communications with an individual • One to Group – communications with select group, team or family • One to All Possible – broadcast communications with unknown people				
Creation					
rk	• Recording • Synthesizing	– Requesting event notification – Selecting printer from suggested list			

Figure 3—Distraction Matrix for Matchmaker



Restaurants
Friends
Fred
Bryan
Daphne
Vending Machi
Telephones
Printers
Wean - 1

Information on **Meyer, Bryan**

Phone Number: "412-802-6819"
Email: bmeyer@andrew.cm
Home Page: http://www.andrew.cr
Address : 3 Bayard Rd
Apt. 28
Pittsburgh, PA15230
Nick Name: Big Boy
Away Message: Try me again later
Description: I show up 30 minute

Figure 4—Portable Help Desk user interface

User: "Locate Fred."
Speech-PHD: "Fred is located in Hamburg Hall"

User: " What is Bryan's phone number?"
Speech-PHD: "Bryan's phone number is 412-802-6819"

Figure 5—Speech Portable Help Desk transcript

expert user may depend on many factors such as technical expertise, friendliness, proximity, and availability. The Matchmaker infers expertise and skills by observing an expert's track record rather than by asking him explicitly. This application utilizes temporal context in determining the availability of an expert and spatial context in determining the distance between the user and expert.

For large projects and design groups, no single individual has the expertise to perform every task. The Matchmaker system connects a user's query with an expert who is located near the user, who is available, and who has a profile listing the skills needed and has a history of answering similar questions. By choosing an expert who is near the user initiating the question, the expert's time is not used for moving to the user. After contacting the expert with the question, the Matchmaker system then requests feedback from the chosen expert to determine if they are best suited to the question. The database updates its profile of the queried expert in order to increase the accuracy of choosing experts. The Matchmaker system reduces the time and increases the efficiency of locating help.

The Matchmaker system has been instantiated, enabling users to efficiently contact CMU's School of Computer Science Computing Support Group (CSG) to resolve their queries. The CSG maintains an extensive database of previously answered requests; this information enabled the system to generate profiles of the CSGs experts. Figure 3 shows some of the activities supported by the Matchmaker system.

Figure 6 shows the Matchmaker system architecture. The user's query gets sent to the server, along with the location of the problem. The server sends the query to the Information Retrieval partition that searches the database for similar past queries, experts who have answered these queries and experts with similar knowledge. The returned list of experts is sent to the Matchmaking partition. The location of these experts and their schedules are compared to the context of the query. The chosen expert is then notified of the request.

2.1.3 *Privacy Guard*

While PHD offers a valuable service to collaborating work groups, the location sensing ability of the system is also a liability. As such, Privacy Guard has been developed to allow users to protect their information. Privacy Guard enables basic privacy policies in addition to advanced expressions describing users, groups and time periods where the user's location can and cannot be reported. The architecture of Privacy Guard is illustrated in Figure 7.

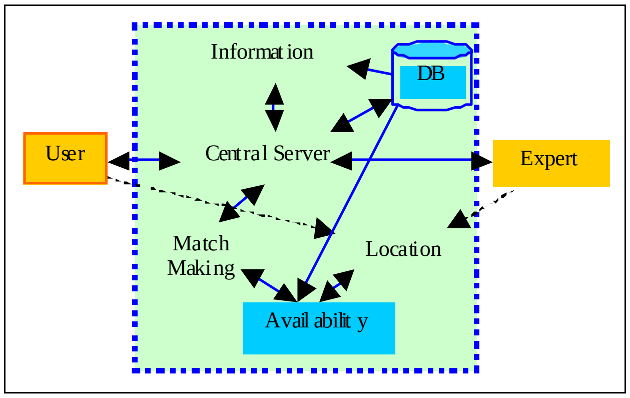


Figure 6— Matchmaker system architecture

Location information and permissions are securely sent to a central server. When the server receives a query for a user's location, the server compares the client with the permissions of the target. The client then receives the location of the target or a refusal to answer the request.

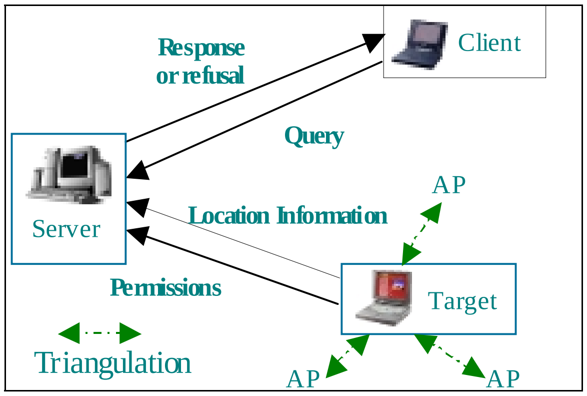


Figure 7— Privacy Guard architecture

2.1.4 *Context Aware Agents*

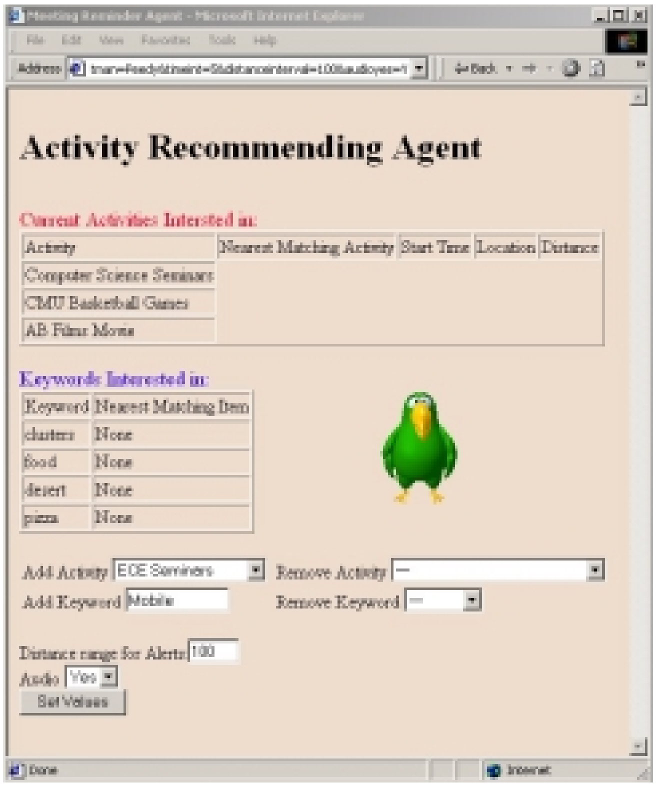
Busy groups tend not to have abundant time to browse their calendars, check for new e-mail or read bulletin boards. Context Aware Agents address this by delivering relevant information to the user when it is needed. Appointments, urgent e-mail and interesting events on a public calendar are shown to the user when they are not engaged in more important tasks. These proactive agents deliver information to the user instead of the user polling the relevant sources. These proactive agents monitor public and private calendars and e-mail accounts. The goal is to provide intelligent calendar management, including scheduling and resolving conflicts with other user's calendars while accounting for the location and available resources for the meeting. These agents are services that are derived from the temporal awareness services including public and private calendars.

The following three Context Aware Agents have been implemented.

– Notification Agent alerts users if they are passing within a certain spatial distance to a task on their to-do list. If a user is walking near their mailbox, the agent alerts the user if they have a package to collect.
– Meeting Reminder Agent alerts the user if they are likely to miss a meeting. The system determines time to a meeting including travel time from their current location.
– Activity Recommendation Agent recommends possible activities/meetings that a user might like to attend based on their interests. For example, the user has set his Activity Recommending Agent to inform him when free food is available. As the user walks through a building, the system identifies a meeting with free food upstairs. The user is notified.

Figure 8 shows an example user interface for the Activity Recommendation Agents. The user defines their interests enabling the agent to recommend upcoming activities. Interests are categorized by activity and keywords. This interface displays upcoming recommendations or the user is automatically notified of activities.

The focus of this work is the development of context aware agents that function as services. This enables simpler applications to utilize agents' context awareness.


Meeting Reminder Agent - Microsoft Internet Explorer

Activity Recommending Agent

Current Activities Intersted in:

Activity	Nearest Matching Activity	Start Time	Location	Distance
Computer Science Seminars				
CMU Basketball Games				
AB Films Movie				

Keywords Interested in:

Keyword	Nearest Matching Item
clusters	None
food	None
desert	None
pizza	None

Add Activity: ECE Seminars
Remove Activity: —
Add Keyword: Mobile
Remove Keyword: —

Distance range for Alerts: 100
Audio: Yes

Figure 8— Activity Recommendation Agent user interface example

3 EXPLORATION OF PERVASIVE COMPUTING ENVIRONMENT

The major goal of our architecture is to enable users to move their work seamlessly between devices. Input and output modalities may transition between visual and auditory user interfaces. This architecture has been implemented to address issues with pervasive computing and wireless infrastructures. The architecture moves the application to a network-connected server, leaving only a minimal interface on the client device. Any device implementing the interface can then reattach the server running the application. The wireless infrastructure introduces bandwidth limitations and intermittent operation. The architecture includes elements that preserve the flow of data between the server and the devices. Optimizing data for the capabilities of a device maximizes performance of mobile devices.

3.1 *Original Architecture*

The original architecture describes four layers as shown in Figure 9. The bottom of the figure has a range of mobile and fixed devices. They are not required to be homogeneous in hardware architecture or operating system. The second layer contains device proxies. For every device there is a device proxy. These represent a transcoding layer present for each device. The third layer is the user proxy layer. Every user has their personal user proxy. Applications and a user's state can be stored in this layer. The fourth layer represents the services layer. Shared applications, utilities and servers are implemented here. All requests between layers are made in hypertext transfer protocol, HTTP. Data structures such as an integer, character and string may be sent and received in these requests. Each request includes user identification and device identification.

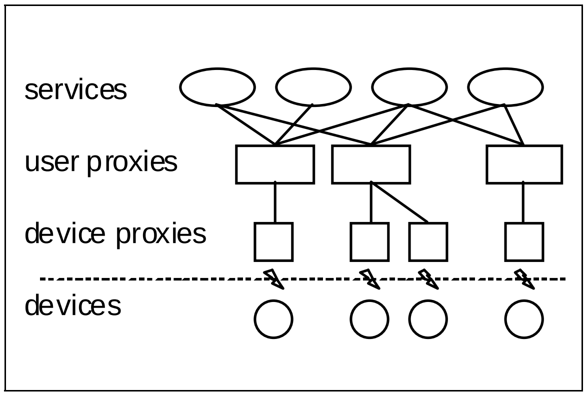


Figure 9—IBM's original architecture

IBM's implementation of this architecture is in Java. Devices execute the service manager. The service manager prepares requests and interprets responses for client applications running on the device. When sending a request, user identification and device identification are included in the HTTP request. The device proxies use WEBI, an HTTP proxy developed by IBM. This proxy intercepts user's requests, passes them through a series of user specified filters and forwards the transcoded requests and responses. The user proxy receives a request, starts an application or forwards it to a service. If the requested service is not well known or defined in the user's preferences, the user proxy invokes the Service Location Protocol to locate the requested service.

3.2 *Revised Architecture*

The IBM architecture allows user preferences and long-term state to be stored in the user proxy or in a service. The decision was made to create a unified database as a service. A fifth layer was added to the architecture above the services, see Figure 10. The new architecture is called Handy Andy, from the name of the wireless network project on the CMU campus, Handheld Andrew. All services and user proxies are granted access to the database based on the privileges of the user authenticated to them. This prevents common data such as user name, address and contact information to be duplicated across systems. Updating this information can be done with a single application. Conflicts pertaining to the format of stored information became apparent, as competing applications had preferred data sizes and types. By using an SQL database, stored procedures could translate actual stored data into any format requested by a service.

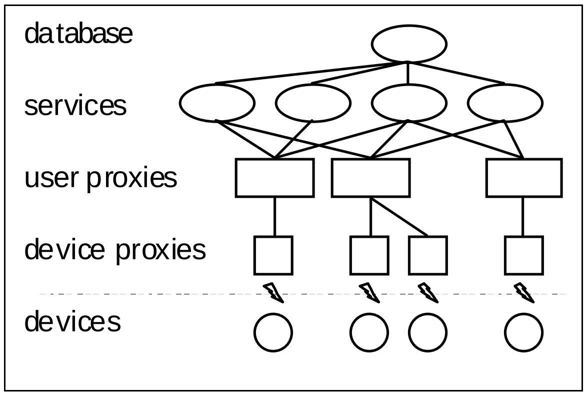


Figure 10—Handy Andy architecture

Idealink is a virtual meeting space tool optimized for small screens of portable devices. The user interface is a shared whiteboard, which can be archived for later review. The tools of the interface are designed to utilize a minimum of screen area while preserving a whiteboard metaphor. The system is implemented within a client/server architecture. This could be accomplished with an application running on the target devices and a server that distributes screen updates to the clients as shown in Figure 11. The Handy Andy architecture enables additional features and ease of implementation within pervasive, wireless environments. Figure 12 shows the architecture elements in use by a typical Idealink session within the Handy Andy architecture.

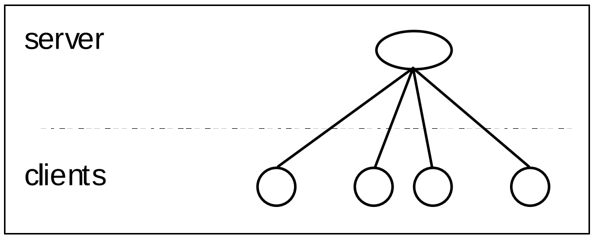


Figure 11—Basic Idealink architecture

The Handy Andy architecture allows the system to be more flexible. Problems inherent in wireless networks, such as broken connections and limited bandwidth, are automatically dealt with. Each user has one or more devices running the Idealink user interface. The devices may utilize color or black and white displays; their screen sizes could range from a watch size liquid crystal display to a wall size projected image. The device proxy instantiates filters that scale the size of screen updates, and it adjusts the color depth according to the device properties. The devices do not use valuable clock cycles and battery power for these operations. If the communications channel between the device proxy and the device is broken, the device proxy caches updates until the connection is reestablished. If the user wishes to use a different device to participate in the Idealink session, they can start the Idealink user interface on any of their devices, and continue the meeting. The user proxy knows what meeting is taking place by the user's calendar, this allows the system to automatically negotiate who is to be included in the Idealink session. The user proxy stores preferences, including the tool palette layout, keystroke combinations selected by the user. The Idealink service combines each user's additions to the session and distributes these updates to each client. At the end of the meeting, the service archives the session in the database.

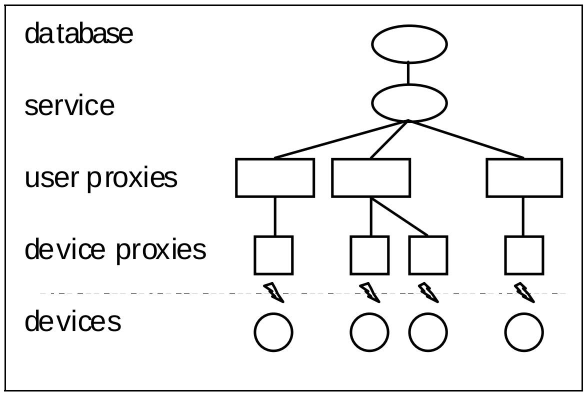


Figure 12—Idealink architecture with Handy Andy

3.3 Location Sensing Service

The Location Service generates a key parameter of context information. To determine the location of a user, their computer measures the signal strength to all available wireless access points. The location sensing compares measured signal strengths to recorded training signal strengths to determine the location of a user. For every location, there is a unique reading of signal strengths gathered from a group of access points. For this training, the user's location is manually input into the computer and about 17 samples are taken and averaged. A table is generated, recording what signal levels to expect at different locations. This only has to be done once and can be saved for use in later sessions and on other platforms. During use, measured values are compared to those in the table and differences are computed. The entry with the smallest difference is taken to be the current position. [8] This architecture is illustrated in Figure 13. The client requesting the location of a target sends their request to a server. The server may use a caching mechanism to answer the request, or send the request to the target user. The target user's computer determines its location and sends the results to the server. The server completes the transaction by sending the location of the target to the client.

The method utilized to determine the location from signal strength information is significantly more accurate than standard GPS [8]. This accuracy is reflected in Table 1.

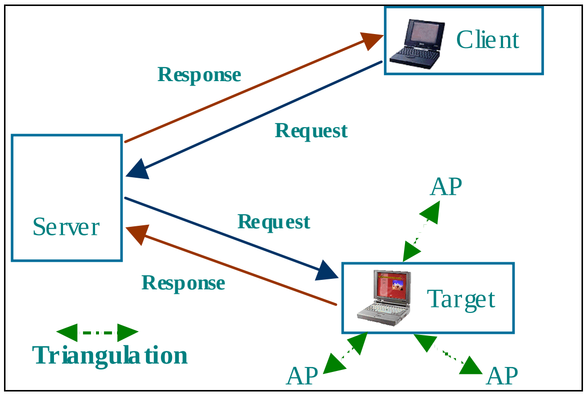


Figure 13—Client-side location service architecture

Accuracy	Strength (dBm)	Distance (feet)
68.6 %	+/- 0.939	+/- 5
95.4 %	+/- 1.146	+/- 10
99.9 %	+/- 2.817	+/- 15

Table 1—Accuracy of location measurements

3.4 Client/Server Speech Issues

Speech-PHD requires significant computing resources for the automatic speech recognizer (ASR) and for text-to-speech conversion. When in development, no mobile device had the needed computing power, memory and non-volatile storage. CMU's Sphinx ASR [10] and the Festival Text-to-Speech software were used. Placing the ASR and text-to-speech software on a server solved the resource limitations, but introduced network latency. Latency was measured from the end of the user's query until the system began its response; this is listed in Table 2. Transferring the entire speech utterance for both the query and the response required almost 5 seconds. By modifying Sphinx to stream the user's query, and marinating Festival's behavior of transferring the entire response utterance, latency was reduced to 2 seconds. By modifying Festival to also stream the speech utterances, the delay was reduced by a total factor of 25.

Latency (sec)	Transfer File Query	Streaming Query
Transfer File Response	5	2
Streaming Response	-	.2

Table 2—Speech-PHD network speech latency

3.5 Lessons Learned

The Handy Andy architecture provides a useful framework to develop persistent applications. The architecture is extremely broad in its description, allowing applications to be implemented in a very portable manner.

A successfully implemented device proxy has the ability maximize the usefulness of a device while offloading expensive conversions to a network side server. Simple applications only need to be implemented in the user proxy or as a service.

There was a tendency to make the device proxy do more than it was capable of. Speech and user interface adaptation were two filters that were explored. Current automatic speech recognizers are not accurate with unlimited vocabularies. Most ASRs require the knowledge of the user's language in the form of a language model. This would have to be provided by the application and therefore would no longer be transparent to the application programmer.

The database enabled data to be shared among users and services. The interface for the data could be customized for each service, limiting issues with proprietary application protocol interfaces, APIs. The data inherited the security model of the database, allowing user permissions to be specified and enforced. While this was convenient for programming, it introduced a single point of failure for the system. The database limited performance and exposed all shared data to security risks.

4 Future directions

Proactive agents in the shown system are not able to access system level functions such as starting new applications on the behalf of the user. System level security must be addressed in such a way that enables user beneficial proactive agents while protecting the user's system.

While context information is useful for generating more intelligent behavior in systems, this information is a liability for the users of the system. Location information is a prime example. The security of this information must be addressed at all levels of the system, including the architecture, protocols, inferred preferences and user specified preferences.

The location service is not optimized at this time. Requiring the tracked client to return it's current location for every request uses power and compute cycles of the limited mobile devices. Ideas for increasing the efficiency and scalability of the location service include caching and predicting the location of users.

5 Conclusions

In this paper we report on a novel framework for context aware computing, and categorization of developed applications within its spatial and temporal domains. Three context aware applications have been developed that utilize the services from the context aware framework. Two different user interfaces, visual and audio, have been implemented and contrasted for the same application. The paper introduces a generic architecture for pervasive computing, explores and refines its design space, and proceeds towards specific instances and evaluation of the architecture. A Distraction Matrix has been defined as a metric of user's attention resources, as pervasive computing environment that causes minimal distraction, needs to be context aware.

6 Acknowledgements

We would like to acknowledge the funding support of IBM Research, National Science Foundation grant 1394/99, and the Defense Advanced Research Project Agency. We would also like to acknowledge the students from the Rapid Prototyping of Computer Systems and Mobile Computing courses for their contributions to the project. We thank Mike Karasick for his continuous support to the project.

7 References

[1] Anind K. Dey, Daniel Salber and Gregory D. Abowd. In the *Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE '99)*, Dublin, Ireland, December 13-14, 1999. pp. 114-128.

[2] Anind K. Dey, Daniel Salber, Masayasu Futakawa and Gregory D. Abowd. GVU Technical Report GIT-GVU-99-23. Submitted to the 12th Annual ACM Symposium on User Interface Software and Technology (UIST '99), June 1999.

[3] Kortuem, G., Segall, Z., Bauer, M. "'Intelligent' Enviroinments." *Second IEEE International Conference on Wearable Computing* (ISWC), October 1998.

[4] Van Laerhoven, K. and Cakmakci, O. "What shall we teach our pants?" In Proc. of the fourth International Symposium on Wearable Computers, ISWC 2000, Atlanta, ISBN 0-7695-0795-0; IEEE Press, pp.77-83.

[5] Mann, Steve, "Humanistic Intelligence: 'WearComp' as a new framework and application for intelligent signal processing." Proceedings of the IEEE, Vol. 86, No. 11, November, 1998, pp 2123-2151.

[6] Mann, Steve. Humanistic Intelligence. Proceedings of Ars Electronica, Sep 8-13 1997. Invited plenary lecture, Sep. 10, http://www.wearcam.org/ars/hi.html.

[7] Smailagic, A., Siewiorek, D., "User-Centered Interdisciplinary Design of Wearable Computers", ACM Mobile Computing and Communications Review, Vol.3, No.3, 1999, pp 43-52.

[8] Small, J., Smailagic, A., Siewiorek, D.P., "Determining User Location For Context Aware Computing Through the Use of a Wireless LAN Infrastructure", Submitted to ACM Mobile Networks and Applications, 2001.

[9] Starner, Thad, Schiele, Bernt and Pentland, Alex. "Visual Context Awareness in Wearable Computing." *Second IEEE International Conference on Wearable Computing* (ISWC), October 1998.

[10] http://www.speech.cs.cmu.edu