Location Based Reminding System

Jacob Christensen, Jai Modi
Computer Science and Engineering
University of Washington, Seattle, WA
{jacoblc, modijai}@cs.washington.edu

ABSTRACT

Time based reminders are well established and are used extensively. However, in our daily lives, it is often location that serves to remind people of certain tasks. The location based reminding system is a project that will trigger reminders based on a user's current location. The user interface for setting and receiving reminders is very easy to use. A user manages locations and reminders into the system from a simple web-browser interface. We present the use of a LCD wrist-watch and the Personal Server. The Personal Server uses WiFi beacons to determine the user's current location, and issues reminders to the wrist-watch worn on the wrist of the user.

1. INTRODUCTION

A consistent problem faced by many people is forgetting to do a certain task when they are close to a particular location. A traditional handheld device only has the ability to remind within the context of time and requires the user to keep the bulky form factor readily accessible. For our embedded systems capstone class, we are attempting to solve this problem by building a location based reminding system designed to be reliable and accurate, in addition to utilizing a convenient, familiar form factor.

The interesting thing about this problem is that it is very common and is faced by most busy, forgetful people. It isn't a problem that affects a specific group of people, so it is important to consider a variety of potential users. A good solution to this problem would work both indoors and outdoors, and provide a simple user interface for managing reminder locations.

Our system consists of a Personal Server device as well as a LCD wrist-watch. Ideally, the Personal Server can perform location computations in an unobtrusive place, such as a backpack, and the watch can be worn on the wrist. The Personal Server was chosen for this project because it can perform the necessary computations quickly and can also act as a web server that provides a user interface. The LCD wrist-watch can easily replace the common watch in addition to providing much more functionality.

Our system is very easy to use and interface with. Users can add locations and reminders by simply opening a web page hosted by the Personal Server and entering the information in a web browser. As reminders are added, the Personal Server continuously determines its location using nearby WiFi beacons and compares it with the added locations. When the Personal Server is within the distance specified by the user of a location, it will send a reminder message to the user's wrist-watch. The wrist-watch will then display the location name and message on the LCD screen. Once a message is displayed, the user has the option of ignoring the reminder, or deleting it from the system by pushing a button on the watch.

A potential scenario for the use of this system would be to remind for simple tasks such as mailing a letter at the post office, stopping by the bank to deposit a check, or picking up some milk at a grocery store. Consider the following example: you need to return a book you borrowed from a university library by a certain date, and you can never remember to drop it off when

you're close to one. Using our system, you can set a reminder to display a message on your watch when you're within walking distance of a library so you can drop off your books, and save yourself the cost of overdue fees.

In this paper, we describe the design and implementation of the Location Based Reminding System, followed by the evaluation of our prototype. We also discuss possibilities for extending this project in the future.

2. RELATED WORK

Similar research has been done in the past on context and location awareness directly related to reminding. The Stick-e Note Architecture proposed an application that uses a mobile computer attached to a GPS receiver. This computer is then used to display information on various attractions as a visitor walks around Disney World and encounters digital Stick-e notes. This project also mentions that using mobile devices such as palm-top computers or PDA's are ideal candidates to exploit the technology of location based reminding. In fact, location is just a small portion of the different types of contexts a computer can utilize for reminding; others include the presence of objects or people and environmental conditions [1].

Another project previously worked on was MemoClip: A Location-Based Remembrance Appliance. The MemoClip is a small, wearable location-aware device that would beep to remind users of their tasks based on some location. The MemoClip consists of a microcomputer (with sensors and an LCD display), infrared LocationBeacons (solar panel-powered, connectionless devices that are installed at places of interest) and a programming device that is used to program both the MemoClip and the beacons. The MemoClip is pinned on the user's shirt, and when the user moves, sensors on the MemoClip detect this motion and send out requests for location information. If a nearby beacon receives these requests, location information is sent back to the MemoClip. This location information is compared to the database, and if a match is found, the MemoClip beeps and displays a message on the LCD [2].

The Positional Pilot was a project that was worked on by a couple of students at the University of Washington for their embedded systems capstone class offered in winter 1998. The project implemented a personal reminder system by using a PalmOS 2.0 application that combines GPS information from a receiver with a set of reminders for certain locations. The user would be able to set off an alarm to go off at a particular location, and the next time he/she is in the vicinity of that location, a reminder would be displayed on a PalmPilot [11].

Our project is very similar to the MemoClip and the Positional Pilot projects described above, as it focuses mainly on the context of location-based reminding. Similar to our project, the MemoClip beeps and displays a message on an unobtrusive device when the user is close to some location. However, the Positional Pilot displays messages on a Palm Pilot, which is slightly an interfering device since it has to be removed from a backpack or pocket. We feel that the watch is a much more accepted device that users are already comfortable wearing, and it is easier to interact with a device on the wrist rather than the lapel or a Palm Pilot. The MemoClip reminding system is also limited by the range of the infrared beacons, and the Positional Pilot uses GPS which does not work indoors. Our WiFi based system will allow for a much larger coverage area with comparable accuracy.

3. IMPLEMENTATION

Architecture

Figure 1 shows the basic architecture of our system. Most of the major components are installed on the Personal Server designed by Intel Research. These components include Place Lab (software used to compute current location), Crossbow Mica2 mote (micro-controller used for radio communication to the wrist-watch), Apache Web Server (http server),

and a SQLite Database (stores location and reminder information). In addition to the Personal Server, our system also uses a LCD wrist-watch called the DisplayMote, designed by Intel Research and the University of Washington. The DisplayMote serves as a wrist-watch, and is capable of displaying reminder messages. A web browser is also used to enter location and reminder information, and the entire system relies on 802.11 wireless hotspots and Place Lab to determine location information. Each component is described in detail below.

device without any traditional input/output capabilities such as a keyboard or a display screen. The Personal Server was built by Intel Research Seattle, and it runs Linux [9]. The Personal Server has an Arm XScale processor operating at 400 MHz. It has a Compact Flash 802.11b wireless networking card which is used for wireless communication and a Compact Flash memory card for data storage. It communicates with the DisplayMote via radio active message packets on a

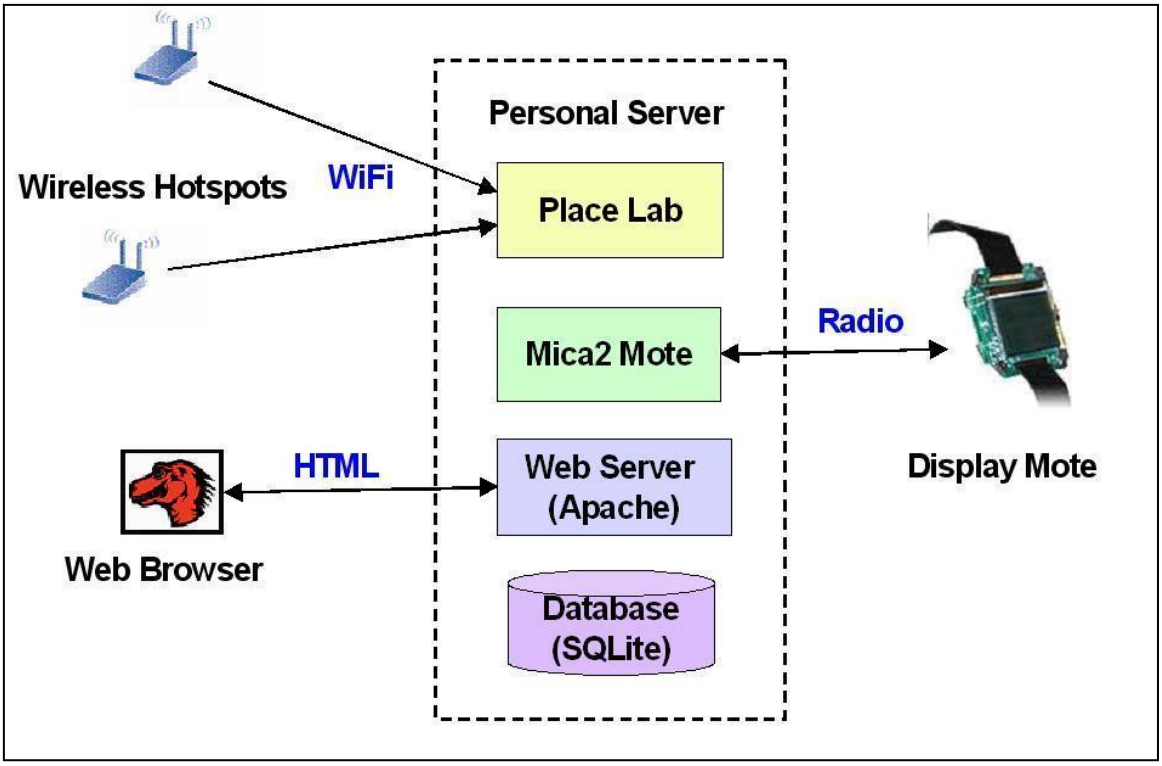

Figure 1: Basic architecture of location based reminding system

Components

The location based reminding system is fairly a simple concept, but consists of many components. The biggest challenge we faced was getting the components installed on the Personal Server and running them reliably. The following is a description of the components and their implementation in our project:

a) Intel Personal Server: The Intel Personal Server is a mobile handheld Crossbow Mica2 mote that has the TinyOS program TOSBase installed.

b) Web User Interface: The Personal Server is running an Apache Web Server. To add locations and set reminders for our system, the user would have to go to a website and enter in this information. Keeping in mind that ease of use was an important project goal, we have designed the web user interface to be very user friendly.

Figure 2 shows a snapshot of the web user interface of our system. Users have the ability to manage both locations and reminders via this web interface.

To add locations, a map must be selected, after which a point on the map must be clicked. Once this is done, the latitude and longitude of that point is displayed in textboxes. After this, the user must provide a name and category for the location selected, and click the submit button.

a category is selected, then a reminder will be added for every location under that category. Once the selections are made, users enter additional information about the reminder. They specify how many minutes away they must be by either foot or car for a reminder to be issued and the message to be displayed on the wrist-watch. The time for reminding that is entered, is then converted into a distance by multiplying it by different a scalar

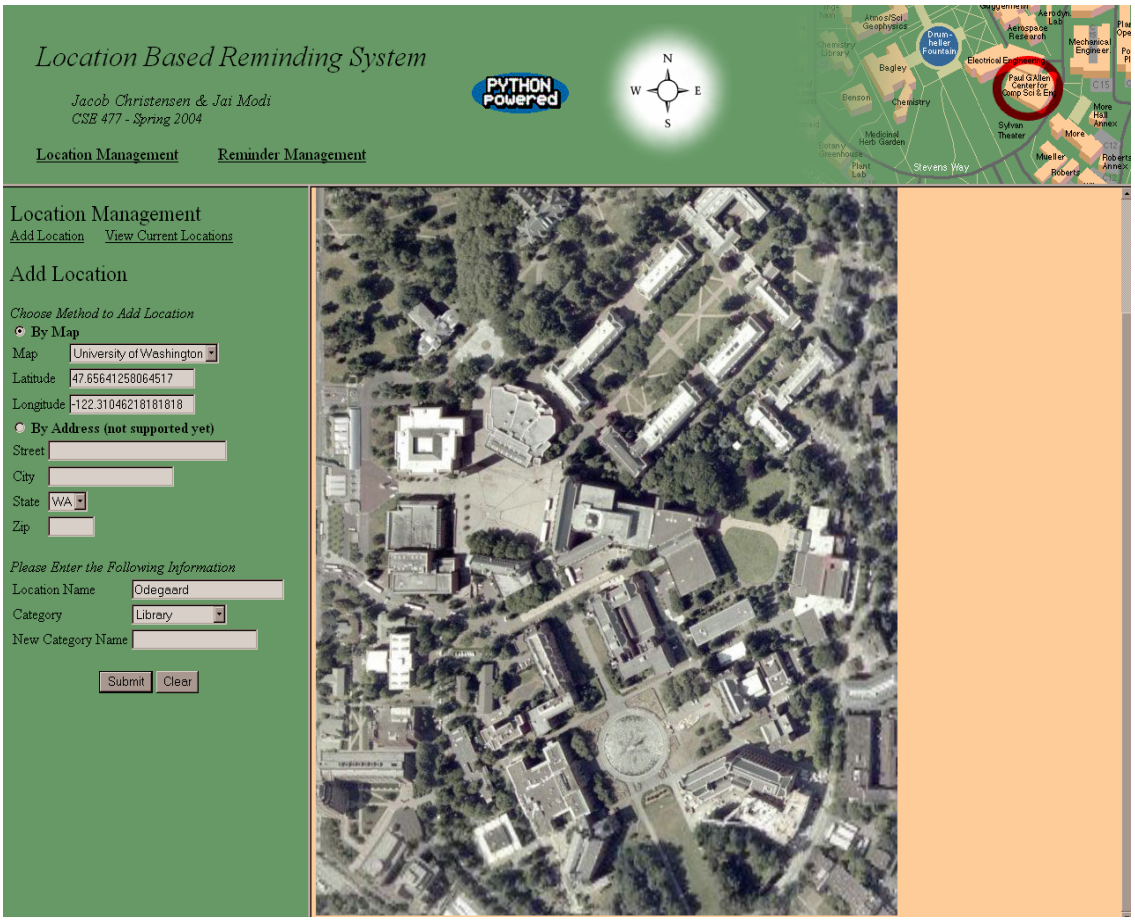

Figure 2: Snapshot of web user interface

To set a reminder, the user has the option to select either individual locations, or entire categories. After that, any number of locations or categories can be selected by simply selecting a checkbox corresponding to a location. If

constant, depending on whether car or foot was chosen.

The web interface was created with CGI scripts using the Python programming language. The web pages are all dynamically generated by

querying a database, and data is inserted into the database when a location is added or a reminder is set. Initially, we had planned to implement the web interface using Perl CGI scripts, however we were unsuccessful in installing the Perl modules needed to access the database on the Personal Server. Noticing that the Personal Server has much greater support for Python, we decided to switch to Python [5,6].

c) **Database:** In order to store and retrieve information atomically in an organized manner, we decided to store information in a database on the Personal Server. We chose to install a SQLite database: an embeddable, lightweight, relational database [4]. The database schema, as shown in Figure 3, consists of 4 tables: Locations, Categories, Reminders, and Maps. The database is accessed using PySQLite, the Python driver for SQLite, and also JNI from within Java. We had initially planned to install a MySQL database on the Personal Server, but ended up using SQLite since it is much lighter and easier to install on the Personal Server. One disadvantage to using SQLite is its lack of data types. This initially caused trouble when interacting with the database in Python and Java.

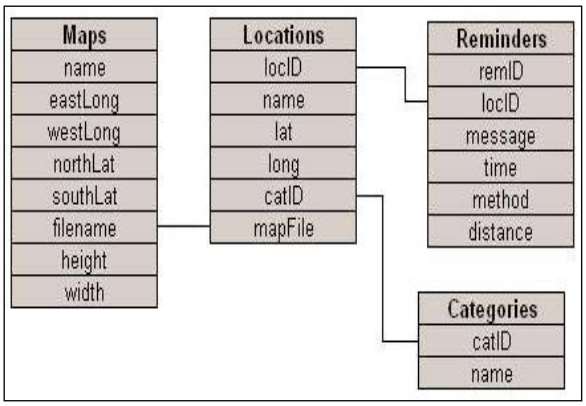

Figure 3: SQLite Database Schema

d) **Place Lab:** Place Lab is the software that facilitates user positioning for location enhanced computing applications. It was created by Intel Research, with the motivation being the widespread proliferation of WiFi technology. Place Lab exploits the fact that each WiFi access point has a globally unique MAC identifier that the access point periodically broadcasts. Combining this information with GPS that maps MAC identifiers to latitude, longitude coordinates, Place Lab allows client devices to passively listen to nearby access points and determine their own location based on the location of the WiFi access points it can hear [3].

Place Lab was installed on the Personal Server, and is running a WiFi Spotter and a Beacon Particle Filter Tracker that gives periodic estimates of the current location. With this data, we can calculate the distance between the estimated location and each of the reminders that have been set. If any of those locations are within range, reminders are issued to the DisplayMote. The application that does all this computing was written in Java.

e) **DisplayMote:** The DisplayMote is a microcontroller with an LCD display. TinyOS, an open-source operating system designed for wireless embedded systems is used on the DisplayMote [10]. The TinyOS application running on the display mote receives radio signals sent by the Personal Server. When such messages are received, the application displays the message on the screen and flashes a blue LED on the side of the watch.

Figure 4 shows an example of the display mote with a reminder message displayed. Once such a message is received and displayed on the mote, the user has two options: ignore the reminder or delete the reminder. If the reminder is ignored, another reminder is issued the next time the user is within range of the current location, and a message is displayed again. This is implemented by adding the ignored reminder to an ignore list, and

subsequent messages received for this reminder are ignored.

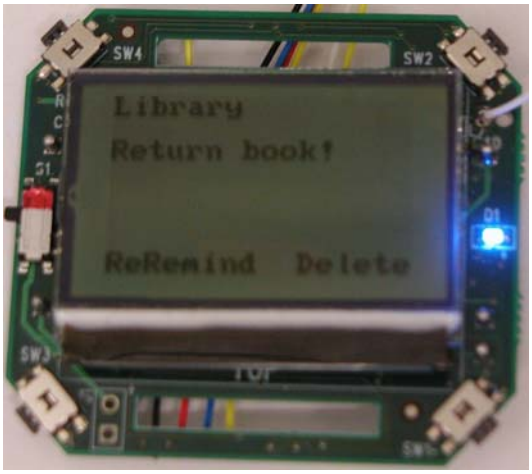

Figure 4: Example of DisplayMote with message and flashing LED

This ignore list is then cleared periodically. If the reminder is deleted, the display mote sends a message back to the Personal Server to delete the reminder from the database. The display mote application is also capable of receiving several reminders at the same time. The messages are then displayed on the screen by cycling through a list of reminders. This is necessary to allow the user to reasonably receive and view several reminders at the same time if multiple reminder locations are all within range of the Personal Server.

4. EVALUATION

The location based reminding system was tested out thoroughly, and it seems to be working well. The web user interface has been tested comprehensively, and it is working reliably at all times. To test the location reminding, several locations at the University of Washington, Seattle campus were added into the system. After that, reminders were set for these locations with a time range of 1 minute by foot (converts to 80 meters), and a message to be displayed. Next, we took the Personal Server and the wrist-watch and walked out on campus. Figure 5 shows a map of the University of Washington campus. The red spots mark the locations where reminders were set. The blue spots mark the locations where the reminders were received the first time on the wrist-watch. The numbers near the spots indicate the order in which the reminders were received; for example, the number 2 near the blue spot indicates the where the reminder was received for the reminder set at the red spot marked with the number 2, and so on. The black dotted line shows the path that was taken with the Personal Server and wrist-watch. We were successful in getting all reminders on the DisplayMote within a reasonable distance of about 80 meters as we approached the target location.

We were also successful in getting two-way radio communication between the display mote and the Personal Server. The buttons were tested out and they seem to be functioning as expected. Deleting reminders sends a message to the Personal Server to delete the current reminder from the database. Ignoring a reminder simply adds the current reminder to an ignore list, so all subsequent messages with the same reminder are ignored for some period of time. In the event of multiple simultaneous reminders, the DisplayMote successfully iterates through each of them, giving the user an opportunity to respond.

Reliability and accuracy were two important criteria for the project's success. If the system works well at times and not at others, then the design needs to be reconsidered.

As we tested our system, all of the reminders were given and were deleted reliably. False reminders were never issued, and correct reminders are constantly issued as long as we remain at the location. The web user interface is slightly unreliable after long periods of time, because the wireless connection seems to timeout. However, we were satisfied with the reliability of all of the hardware involved. The personal server was calculating and issuing reminders frequently, at approximately five seconds, and the only complaint we had about the DisplayMote was the poor quality of its display in moderate sunlight. Radio communication between the DisplayMote and the Personal

Server was reliable enough because any lost packets would likely be sent again within five seconds during the next location calculation iteration. Overall, the entire system is functioning reliably, and we are successful in receiving reminders at different locations.

Evaluating the accuracy is somewhat subjective for our reminding system. Much of the accuracy depends on the ability of Place Lab to estimate the current location. Both indoors and outdoors, Place Lab runs correctly as long as we are in an area with WiFi beacons. Because the number of wireless networks is growing, this method of determining location will get even better with time. However, as it relates to our project, typical reminders will be on the order of at least hundreds of feet. The average walking pace is approximately 80 meters per minute. Place Lab currently gives us at least this granularity; hence it is a very effective method to use.

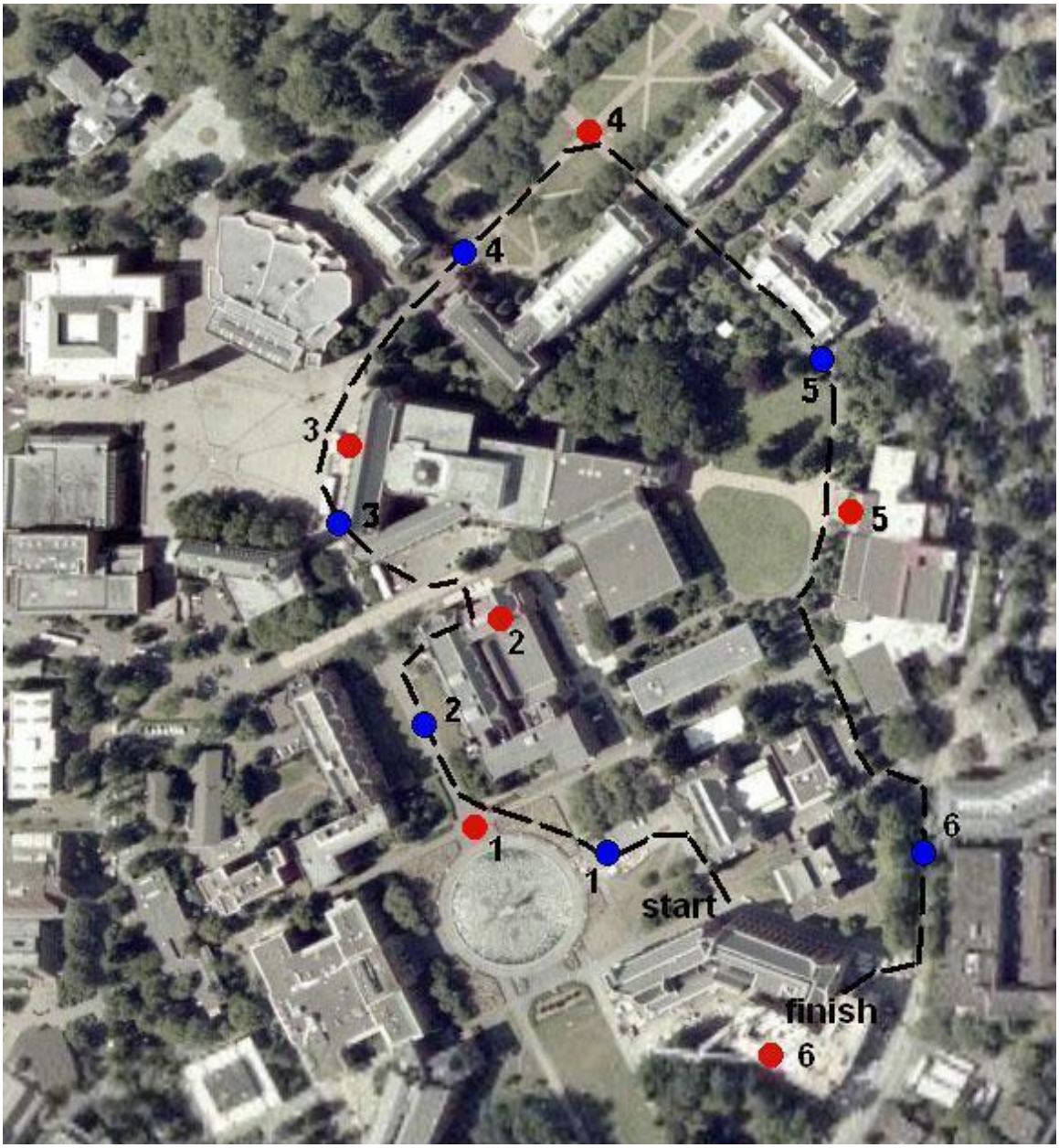

Figure 5: Map of University of Washington campus

5. FUTURE WORK

Although our results seem promising, a lot more work can be done to extend the location-based reminding system. Here are a few examples:

a) **Wrist-Watch Text Entry:** Currently, our project only allows users to add locations and reminders from the web interface. Users might want to have the option of adding reminders when they are currently at some location, in order to remind themselves the next time they are close to that place. We currently have functionality for sending messages back to the Personal Server for adding locations and reminders, but still need the ability to enter text on the wrist-watch to make it practical. With wrist-watch text entry, users will be able to add locations by simply entering reminder information on their watches when they are currently at some location, rather than through the web interface.

b) **Adding Location by Address:** Our web interface presently supports the addition of locations by choosing places from a map. Another way to add locations could be to enter the address of that location. This option might require the use of web services from map finding website such as Maporama, in order to convert an address into latitude and longitude [8]. Nevertheless, it might be easier for some users to add locations by address instead of pointing out locations on a map.

c) **Selecting a Region Instead of a Point:** Our web interface currently allows user to click on points on a map in order to add locations. A better way to allow users would be to select a polygon such as a square or rectangle or a circle as their location. Once a region has been selected, the system would need to check if the current location is contained inside the selected region. This method might be more effective for certain locations that span a wider area.

d) **Reminding based on Time:** Another way to extend this project would be to add time based reminders along with location based reminders. Users might want to be reminded based on their location only at particular times in the day. The addition of time to location based reminding will be able to serve this need.

One of the most important points we learnt working on this project was that it is much wiser to first look at the technology available to you, and then plan on the design of the implementation and tools that you will use. This is especially important if you have limited time to complete a project. We initially assumed that we would be successful in installing some software on the Personal Server, and hence came up with a design to use Perl for the web interface and MySQL for the database. However, we ran into problems with this initial plan costing us many hours, and soon had to switch over to Python and SQLite as they were better supported for the Personal Server.

6. CONCLUSION

Location based reminding has tremendous potential in the future. In this paper, we have presented a prototype of a location based reminding system what was designed, implemented and tested in a short period of 10 weeks. Our project consisted of several components, and getting them all to work individually and together was a challenge for us. Due to limited time and resources, our system is simple, but with better technology and more time, our foundation for location based reminding can be extended in several possible directions. In our busy lives, location based reminding can help people by reminding them to do certain important tasks depending on their location. This is often more necessary than reminders within the context of time. The results from our experiments also show that using Place Lab to determine location from 802.11 wireless hotspots turns out to be quite accurate for a location based reminding system. Moreover, our tests show that our system is dependable with reminders being triggered off reliably, and with two way radio communication between the DisplayMote and the Personal Server. Most

importantly, our prototype is simple to use, allowing users to interface with the system by a web-browser and a convenient wrist-watch form factor.

7. ACKNOWLEDGEMENTS

We would like to thank Dr. Gaetano Borriello, the instructor for our class, for his support, guidance and ideas to help us along with our project and to ensure we were able to complete a prototype within a short period of 10 weeks. We are extremely grateful to Waylon Brunette, the teaching assistant for this class, for the countless hours he spent helping us with the technology we needed to complete this project. We are also thankful to Cameron Tangney and Alan Liu for their help in installing Place Lab on the Personal Server, and Trevor Pering for his help with general questions on working with the Personal Server. We are also very thankful to several of our classmates for help received along the way, but special mention must be made of Philip Grin and Jerry Fu for their contribution with the display motes, and Tom Anderl and Karl Yerkes for assisting us with special features of Linux running on the Personal Server.

8. REFERENCES

[1] Pascoe, Jason. The Stick-e Note Architecture: Extending the Interface Beyond the User. International Conference on Intelligent User Interfaces. 1997, 261-264.

[2] Beigl, Michael. MemoClip: A Location-Based Remembrance Appliance. Personal and Ubiquitous Computing. Volume 4, Issue 4, 230-233.

[3] Place Lab (Intel Research Seattle), <http://www.placelab.org/>

[4] SQLite: An Embeddable SQL Database Engine, <http://www.sqlite.org>

[5] The Python Programming Language, <http://www.python.org>

[6] Lessa, Andre dos Santos, Python developer's handbook, 2001.

[7] IPKG Find: Search Engine for pre-built packages, <http://ipkgfind.handhelds.org>

[8] Maporama, <http://maporama.com>

[9] Intel Research Seattle, <http://www.intel-research.net/seattle/>

[10] TinyOS, <http://webs.cs.berkeley.edu/tos/>

[11] Clifford Chu and Alex Pescaru. Positional Pilot: GPS Enabled Task Finder.

9. APPENDICES

a) Personal Server Installation: The root flash on the Personal Server has very limited free space, so all components such as Place Lab, the database, and the www directory of the web server must be stored on the compact flash card. To install IPKG packages on the compact flash card, the following must be done:

i. Edit the /etc/ipkg.config file by adding the following line:

dest cflash /mnt/cf1

ii. To install the ipkg package, type the following command:

ipkg –dest=cflash install <pkg_name>, where pkg_name is the name of the ipkg package to be installed.

b) Resources/Code: All the resources used, and source code written for this project are linked of the following webpage. <http://www.cs.washington.edu/education/courses/cse477/CurrentQtr/projectwebs/cse477f/Submit/>