

**Will You Help Me:**  
Enhancing personal safety and security utilizing mobile phones

by

Jae-woo Chung

Submitted to the Program in Media Arts and Sciences,  
School of Architecture and Planning  
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2006

© Massachusetts Institute of Technology 2006. All rights reserved.

Author.....  
Program in Media Arts and Sciences,  
August 14, 2006

Certified by.....  
Christopher Schmandt  
Principal Research Scientist  
M.I.T. Media Laboratory  
Thesis Supervisor

Accepted by.....  
Andrew Lippman  
Chairperson  
Department Committee on Graduate Students  
Program in Media Arts and Sciences



# **Will You Help Me:** Enhancing personal safety and security utilizing mobile phones

by

Jae-woo Chung

Submitted to the Program in Media Arts and Sciences,  
School of Architecture and Planning  
on August 11, 2006, in partial fulfillment of the  
requirements for the degree of  
Master of Science in Media Arts and Sciences

## Abstract

This thesis describes the design and development of a system that is aimed for enhancing safety and security using cellular phones. This system has two main components: a master phone application to assist people who need to take care of their loved ones, and a slave phone application to provide help to care-recipients who need attention from their caregivers. This system applies location awareness (GPS), awareness of social activities (communication activity and proximity with close peers,) and peer-to-peer data communication as its core technologies.

There are three sub-components that are implemented the system: First component is for providing a set of information in order to enhance awareness of crime around users' surrounding areas. This component is used to assess risk on users themselves as well as their property. The second component is a sub-system that is dedicated for detecting a possible abnormal transit behavior. Consequently, the system alerts this abnormality to both the system users and their caregivers. Third component detects nearby phone to cultivate social activities with the system user's peers. The purpose of finding contacts is to get remote help from friends of the care-recipients.

All the functions provided by the system fall into the gray area in between the state in which care-recipients are completely safe, and that in which care-recipients had an accident severe enough to require help from authorities.

Thesis Supervisor: Christopher Schmandt  
Title: Principal Research Scientist, M.I.T. Media Laboratory



## Thesis Committee

Thesis Advisor: \_\_\_\_\_

Christopher M. Schmandt  
Principal Research Scientist  
M.I.T. Media Laboratory

Thesis Reader: \_\_\_\_\_

Andrew Lippman  
Senior Research Scientist  
M.I.T. Media Laboratory

Thesis Reader: \_\_\_\_\_

Joseph Dvorak  
Research Affiliate  
Motorola Scientist in Residence



## Acknowledgments

First of all, I thank my advisor Chris Schmandt who gave me an opportunity to be a Media-Labber at MIT, and admitting me as his Masters and Ph.D. student. I especially thank him for believing in me and for giving me some amount of autonomy while at ‘The Lab’. I am also thankful for him helping me out in numerous ways with my research and in my development of the route prediction algorithm in this thesis. For almost one and half years (perhaps more), he tested our system and gave me feedback everyday, whether it was hot days in summer or very cold and snowy days in winter. He was testing whenever he was en route to the lab or home, walking or bicycling! How many advisors would do this? But that is only one example. I was indeed motivated by the efforts that he made for this project. Thanks, Chris!

Also, I would like to thank my thesis readers, Andy Lippman and Joe Dvorak who gave me a lot of encouragement and helped me to think about the direction of my thesis topic. When I first met Andy to ask him to be my thesis reader, he gave me lots of encouragement. Also, Andy advised me to see my project within a bigger context and to develop my ideas further. This helped me refine my thoughts and my project in many ways.

Without Joe’s help I could not have done this project. First of all, he provided us with some Motorola phones and he included service too! Also, he motivated me to choose this very interesting topic in the thesis. I often visited him for a discussion and brainstorm about my research. This collaboration helped me to develop my ideas as well as to write this thesis. I hope to interact with him more in the future.

I also would like to thank my colleagues in the Speech Interface Group, Rachel Kern, Stefan Marti, Chaochi Chang, Paulina Modlitba, Matt Adcock, and Jeff Goldenson. Especially I like to thank Rachel for helping me to correct my written English. Also, she was my co-author on the paper “Common Sense Translation Assistant” at CHI2004. Although I made a ‘contract’ with her to exchange my programming skills with her English, I think she helped me more than I was able to help her. Thank you, Rachel. Also, I would like to thank Matt Adcock for helping me correct much of the writing in this thesis. He spent a lot of his time reading my thesis and correcting it. I don’t have any

'contract' with him but I promise to help him his projects in the future. As I submit this thesis, Chaochi is interning in India, but as two Asians who study outside of their home countries, we have shared a common culture-shift and background. I thank him for his companionship. Thanks to Stefan for instructing me in the 'ways of the Media Lab' when I first arrived. To Paulina, I give thanks for helping me with my thesis proposal and for being a guinea pig for some of my early experiments. Jeff gets thanks for proofreading my thesis but also some big thanks to him for letting me borrow his bike to conduct my experiments. It seems like I have made a good use of all my resources and this thesis has become a real group effort! Thanks guys.

I also want to thank my previous academic advisors, Marilyn Tremaine and Robert Rowe, who gave me the support I needed to continue my studies at the Media-Lab. Marilyn opened the door for me to do research in the human-computer interaction domain, and recommended that I consider further study at MIT. Also a Ph.D. graduate of Media-Lab, Robert advised me while I was studying interactive music in NYU. He also recommended to me that I should study at Media-Lab.

Lastly, and at the most, I thank my family for all their support. Without them, I could not come to MIT and continue my studying.





# Table of Contents

<b>1. Introduction</b>	<b>13</b>
1.1. Problem	13
1.1.1. Awareness and Detection	14
1.1.2. Providing Remote Help	15
1.2. Proposed Solution	16
1.3. System Description	17
<b>2. Related Work</b>	<b>19</b>
2.1. Monitoring System on Mobile Phones	19
2.2. WatchMe	20
2.3. Systems for Awareness of Crimes	22
2.4. Reality Mining	24
2.5. Route Finding Algorithm for Road Traffic	25
<b>3. Where are you? – The Risk Meter</b>	<b>29</b>
3.1. Scenario	29
3.2. User Interface	30
3.2.1 <i>Contella</i>	30
3.2.2 Risk-Meter	33
3.3. Trust and Communication	35
3.4. System Architecture	37
<b>4. When you are going? - Route Detection Algorithm.</b>	
4.1. Scenario	41
4.2. User Interface	42
4.2.1. Contextual information	43
4.2.2. Settings	44
4.2.3. Communication	45
4.2.3.1 Alert Messaging	45
4.2.3.2 Remote Assistance	46
4.3. Design consideration on Route Learning and Detection Algorithm	49
4.3.1. Endpoint Detection	50
4.3.2. Route Identification	51
4.3.3 Route Prediction	53
4.3.4 Abnormal Transit Detection	54
4.4. System Architecture	59
4.4.1. Data Structure	59
4.4.2. System Component	61

<b>5. Who is around you? - Finding Contacts Around You</b>	<b>65</b>
5.1. Scenario	65
5.2. User Interface	66
5.3. Design consideration of the system	69
5.3.1. Data logging and implicit information exchanges between phones	69
5.3.2. Search Process	70
5.4. Communication process between Master phone and Slave phone	71
5.5. System Architecture	73
5.5.1. Data Structure	73
5.5.2. Log Handler	75
5.5.3. Bluetooth Communication Handler	76
<b>6. Evaluation</b>	<b>79</b>
6.1. Where are you – Risk Meter	79
6.2. Where are you going – Route Detection	82
6.3. Who is around you – Finding contacts	90
<b>7. Conclusion</b>	<b>97</b>
7.1. Contribution	97
7.2. Future Work	99
<b>Appendix A</b>	<b>101</b>
Route Templates Collected during the Experiment	

## **Bibliography**



# Chapter 1

## Introduction

Although mobile phones were originally simply “telephones”, the current reality is that today’s mobile phones are really computers. As phones become more capable of independent action through programmability and data-processing capability, they can do a great deal more than merely transfer voice data. Mobile phones now come equipped with video cameras and music players, as well as software for personal schedulers, all of which use a powerful processor and a large memory. People are already carrying their phones everywhere and taking advantage of the additional functionality they provide.

By coupling the phone’s computing and data communication capabilities, it is possible to create mobile peer-to-peer interactions that have previously only been achievable with desktop computers. These new phone capabilities allow us to access other phones remotely, as well as to program and command remote phones to execute functions. The ability to execute commands on a phone was previously only available when the phone was physically present.

Additionally, these phones are capable of conveying location and notifying users of proximate mobile phone devices. To meet the FCC’s Enhanced 911 (E911) requirements, all mobile phones are mandated to be equipped with devices that are aware of location, such as GPS<sup>1</sup> [7], or triangulating cell-tower technology, also known as mini-GPS. In addition, more and more phones are equipped with Bluetooth, which is capable of scanning nearby Bluetooth devices. This helps us to be aware of devices that are physically close to us. With these new phone attributes, how can we use phones to enhance security but prevent intrusions of privacy?

### 1.1 Problem

The expectation that mobile phone users are able to answer phone calls most of the time lets us easily check on our loved ones’ well-being by simply calling them. This convenience lets us use mobile phones to call our friends and family to get or to provide

---

<sup>1</sup> Global Positioning System.

help. However, in an accident or emergency situation, circumstances may not be favorable, and we may not be able to call our guardians or care-givers to request help. Although our care-givers may be willing to provide help to us, they are unlikely to know exactly when we need help or require attention unless we explicitly contact them. This raises the question of how to increase awareness of our loved one's situations, and be able to provide help remotely.

### **1.1.1 Awareness and detection**

Let us consider a system that detects the situations in which attention is required from one's care-giver. The system would automatically inform the care-giver that his or her dependents require help. This type of system already exists in the case of domestic fire and burglar alarms. In the area of security and health care, industry and academia have actively conducted research [14], [20] using various external sensors such as visual monitors, microphones, motion detectors, and proximity sensors. Although increasing the number of sensors may result in increased awareness, these sensors may not always be available, particularly in the case of outdoor activities.

Despite limitations in the number of sensors that a pedestrian can carry, mobile phones equipped with GPS receivers enable us to track one's location. This functionality can be used by pedestrians to identify where they are on a map. Also, it is possible for parents to use a GPS service for remotely monitoring their children while they are en route to school, for example. [20] Through a GPS-equipped mobile phone, a child's GPS coordinates are periodically tracked, and his/her parents can see on a map where the child is located.

However, there are a few problems with using the currently existing GPS services to track loved ones' locations for the sake of their security and safety. First, the interface for monitoring position is based on plotting a dot on an electronic map. A moving dot on a map shows information about where the subject is located, but does not give detailed information about what type of situation he/she may be in. Second, the existing services compromise the privacy of the person who is being tracked with GPS. Third, current GPS

systems do not detect abnormal situations. Caregivers are constantly required to focus their attention on the map to assess the situation themselves while their kids are on the way to school, for example. A dot on a map can, at best, provide psychological satisfaction to a concerned parent, but the overall system cannot automatically detect an abnormality in a child's routine, or alert the parent in the case of an unusual situation.

Thus, for a GPS tracking system to be more useful, the system should learn a traveler's regular transit pattern and should be able to discriminate abnormal routes from a user's normal path. In addition, the system should be aware of a traveler's current location and should assess any potential risks in their current surroundings. Increasing awareness of potential crime in various neighborhoods is an important element of enhancing the security and safety of each individual. As the system assesses the potential risk of a pedestrian, it should alert the pedestrian's caregivers whenever the risk exceeds a safe level.

### **1.1.2 Providing remote help**

Increasing the overall security of an individual requires not only monitoring but also managing potentially risky situations. When a risky situation is detected, it is reasonable to think about how to solve it. When we think about providing help to our dependents, one way to provide help is to go to their physical location. However, when our dependents are located too far away for us to get to them in a reasonable amount of time, another solution is to find someone in close proximity to them who can provide assistance. This is not only helpful to our dependents, but it also provides a way to further assess the seriousness of the situation by adding an alternative avenue of communication. This could be especially useful if our dependents' communication channel is temporally unavailable.

We are living in a society where an individual might have multiple relationships with multiple social groups. This makes the task of finding someone to help our dependents difficult, because the most appropriate person to provide help may not be in a social group for which we might have contact information. When our dependents are not living in the same town but we still need to take care of them remotely, it becomes harder to find someone with access to our dependents' social contacts.

In addition, personal relationships are often regarded as private; thus, when we try to access the contact information of someone with whom we don't have a direct relationship, we are attempting to find a communication channel which has not been explicitly allowed previously. Thus, in designing a system for finding someone who can help our remote dependents, we need to find a way to protect the privacy of both our dependents and their peers. Additionally, we need to think about how we might find a contact person in close physical proximity to our dependents, and how to communicate with them successfully.

## **1.2 Proposed solution**

To solve the previously stated problems, this thesis addresses an approach which is described in three sub-themes: where you are, where you are going, and who is around you.

“Where you are” tackles an issue related to a pedestrian's current location. We collected a year's worth of crime logs around the MIT campus to measure personal safety risk. These logs were saved on a geographic information system (GIS) database server to render crime statistics and maps in real-time at a user's request. Users can request a crime risk analysis by providing the server with their current location and information about their commonly transported valuables. Based on the GPS coordinates, the time of a request, and a user's information about his/her valuables, the system will calculate the current risk of theft or damage for each valuable and provide this information to the user.

“Where you are going” tackles the detection of abnormal locomotion patterns while a pedestrian is moving. Based on the user's average elapsed traveling time between two known locations, the system will detect and determine certain abnormalities such as deviation from average transition time and unusual destinations. To achieve this, the system requires users to bookmark locations and specify the desired routes to be detected. A simple interface, such as a button to initiate the system's learning of a new route, is provided to train routes before detecting any abnormalities. Data is initially collected during the training, and later used to check if users are on the normal path heading to their destinations. The algorithm in the system is restricted to detect recurring routines,



such as going to school, to the office, and back home. Although deviations from normal routes do not always imply that an incident has occurred, it is useful to provide this information to caregivers as an awareness of their dependents' possible abnormal situations.

“Who is around you” tackles the discovery of our loved one’s social peers who can potentially provide help in an emergency, and help a remote caregiver to assess an unusual situation. Because the nature of social relationships is dynamic and changes over time, it is hard to find contact information for our dependents’ peers. By investigating the communication frequency between users and each of their peers via phone logs, and examining the time that users spend in close proximity to each of their peers, the system determines who the best possible individuals are to call and ask for help. In order to accomplish this, the system keeps track of communication logs, Bluetooth MAC Address (BTID) logs, and phones’ location logs. Periodically-scanned BTIDs are used to examine the proximity of co-located peers. Since a GPS-enabled phone can locate its position through book-marking certain locations, e.g. “at home”, the system can further search for a person who actually visited the user’s home, by examining the events in Bluetooth logs and location logs. These logs are only kept in an individual user’s phone and other phones cannot directly access this information. Finally, the process of finding a possible remote contact person is determined inside the phone which is running the application, but the name and the contact information are completely hidden from other phones, which are requesting contacts, for the sake of protecting the privacy of everyone involved.

### **1.3 System description**

Although this thesis divides a proposed solution into three small themes, overall, the system is divided into two parts: the cellular phone application, *Contella*, and the spatial database system.

*Contella* is an application platform that allows us to store contact information and to exchange the information between other *Contella* phones via Bluetooth and cellular phone communication network in peer to peer manner. Based on *Contella*, three functionalities for client side services, “Where you are”, “Where are you going” and

“Who is around you” are developed. We used J2ME<sup>3</sup> on Motorola’s i860 and i870 phones as our main development platform for *Contella*.

*Contella* is further divided into a master phone application and a slave application. In the thesis, I refer to the phone that contains the master phone application as “the master phone” and the phone which contains the slave application as “the slave phone.” The master phone has the capability to remotely access and command slave phones when necessary. In order to communicate between master and slave phones, trust must be established between them. The caregivers use the master phone to detect abnormal situations and search for contact points of their care-recipients via remotely commanding the slave phones. The slave phones are used by care-recipients and get help from master phones.

The spatial database system is essentially a subset of a Geographical Information System (GIS) database. This GIS database is the server side of the service “Where are you” The Server contains GIS information as well as crime logs. The crime logs are used to assess whether the slave phones are in a dangerous zone. The information from the crime logs is used on both master and slave phones to increase the awareness of crime history in the area where the users are located.

All in all, the design focus in this thesis is to increase the availability of caregivers by providing adequate information to caregivers, detecting possible problems related to location, alerting both caregiver and care-recipient, and providing the capability of searching for possible contacts who might offer help to both the caregiver and the care-recipient. All the functions provided by the system exist in the range from care-recipients being completely safe, to care-recipients having had an accident severe enough to require help from authorities.

---

<sup>3</sup> Java Platform, Micro Edition (J2ME) is a collection of Java APIs for the development of software for resource constrained devices such as PDAs, cell phones and other consumer appliances.

## **Chapter 2**

### **Related Work**

#### **2.1 Monitoring System on Mobile Phones**

As mobile phones become one of the best candidates to be used as personal security tools, the telecommunication industry and mobile phone manufacturers have developed and provided a few services towards the goal of increasing safety and security both for the individual, and for entire families.

Additionally, in the area of home security and health care, companies and academic institutions are actively conducting research on new solutions. For example, the project called “Wellbeing at Home” [18] uses various external sensors to monitor one’s well-being, such as physiological sensors including heart beat monitors, external visual monitors, and various proximity sensors. However, for someone participating in activities outside of the home, these sensors are not always available.

Although increasing the number of sensors in the home may be useful for increasing awareness of one’s well-being indoors, few companies around the world provide services such as LG Telecom’s “Peace of mind Service” [17], which is available in South Korea. This service is used for monitoring family members while they are en route to school, to the office, and to their homes. However, the service fails to increase the availability of the caregivers of those being monitored. “Peace of mind Service” allows service subscribers to monitor a family member’s location through GPS coordinates that are sent from their family member’s phone via Simple Messaging System (SMS) and shows their location on a map in a phone. For example, parents can monitor their child’s progress towards a destination by looking at a map on their phone which updates its information every five minutes. However, this solution is not really sufficient for detecting an abnormal situation. Rather than trying to detect an abnormality in someone’s routines, and trying to increase the availability of caregivers in an unusual situation, this system only provides psychological satisfaction to caregivers while monitoring their charges. The care-recipients, however, do not receive any benefit from

the system, and are not aware of whether their caregivers are monitoring their safety until they arrive at their destination.

Another service that is provided in the US is “Teen Arrive Alive” provided by [teenarrivealive.com](http://teenarrivealive.com) [21]. GPS and call-in voice services are used in combination to locate young teens or other family members. Subscribed users can track locations of their registered teenagers by accessing the [teenarrivealive.com](http://teenarrivealive.com) website or by calling their Locator Hotline service. After speaking their child’s cell phone number, along with their private pin number, the system provides information about the child’s last known location as street address, the current direction of travel, and the speed of travel. However, the system requires that the server retrieve GPS information constantly, which could be a potential privacy problem. Moreover, the information that the service provides to subscribers is standard GPS information with no additional attributes. For example, although the system provides subscribers with information such as the current speed and travel direction of their child, it still lacks a built-in smart detection derived from a child’s history of transit patterns, and therefore, it cannot alert subscribers when an abnormality is detected along their child’s route.

## **2.2 WatchMe**

Will You Help Me was inspired by the WatchMe [12] project carried out in the Speech Interface Group at the MIT Media Lab. WatchMe is a personal communicator in the form of a wristwatch. The goal of WatchMe is to keep an intimate connection with friends and family through text, voice instant messages, and synchronous voice connectivity. Through embedded GPS in the WatchMe device, the watch can track a wearer’s location, and use the data collected to predict the wearer’s destination and arrival time. Users are allowed to name frequented locations, and the system tracks position via GPS to compute normal routes of travel between two named locations. Then in real-time, it reveals a user’s current location such as “At home”, “On the way to work, arriving in 10 minutes” or “Left home 5 minutes ago.”

The interface of the system presented here that shows a graphical representation of context information is directly inherited from WatchMe. As will be described in later

sections, WatchMe uses a set of icons and text to convey a user's current transit behaviors such as moving, at a particular location, going to a particular location at a certain time, or departed from a particular location at a certain time. The information is displayed such that users only need to glance quickly at their watch to learn about another user's current situation based on location information.

In addition, WatchMe is enabled to learn about a user's route. Routes are defined as trajectories between labeled locations. In the system, routes between two known endpoints are automatically extracted from the GPS data. A route template is calculated from the various instances of a given route; this involves dividing the route into chunks of 100 meters of traveled distance and finding an average latitude and longitude for each chunk. The average elapsed time from the starting point and the average speed are also included in the template.

The canonical route templates are used to predict where the user is going and to estimate the time to arrival. By trying to align the user's current progress to the saved route templates, the route the user is currently traveling along is identified. The alignment is a piecewise comparison of the route segments, generating a distance error for each segment, averaged into a total route error. It is compared to all the canonical routes initiating at the same source. The predicted route is the one with the smallest aggregate error, below a threshold of 150 meters. The route templates provide a means to predict the traveled route, and based on the average time required to reach the current segment, they are able to predict an estimated time to arrival.

Although the route tracking algorithm in Will You Help Me is different from WatchMe's, we carefully studied the behavior of the algorithm used in WatchMe and extracted some of the features that would work better on an optimized algorithm that is dedicated to route detection. This eventually allowed us to successfully develop an algorithm that could be used on a mobile phone with limited memory and computation power. Also, we discovered some disadvantages to using a server for the sharing of context information among peers. The main disadvantage is that when using a server architecture, private information is not sent directly to the user's peers, but rather must be rerouted through server, which has the potential for exposing private information to third parties. In addition, because users are required to subscribe to a service or a server,

the scalability is limited. If peers in a group are using different services for sharing context information in order to share information among all the peers, each one needs to subscribe to each service. If we can share context information directly between peers, then no central server is required. This is the key reason for us to consider a stand-alone version of WatchMe in mobile phones.

The application used in Will You Help Me is different from WatchMe. Although the two systems have the similar concepts of allowing users to share context through mobile devices, and both use location awareness for tracking and predicting users' behaviors, the systems are used in different contexts. While WatchMe is more focused on helping friends and family members to stay connected, Will You Help Me is concerned with enhancing safety and security utilizing mobile phones.

### **2.3 Systems for Awareness of Crimes**

According to G Tomas Kingsley [8], during the 1990s, nonprofit institutions in several cities, including Boston started a project to construct a computer-based information system that would provide a variety of information about conditions and trends at the neighborhood level. The city could use this system to identify patterns of problems as well as opportunities. These databases covered an extensive array of social welfare issues including crime. However, the goal of the project is to democratize information; the information was provided to community leaders so that institutions and residents could build their capacity to enhance decision-making by using the data. Unfortunately, the data access of these databases was a cumbersome process. Information requesters could access and manipulate the database directly to get simple facts from terminals at institutes in some cities, but they needed to ask system staff to get any tables, maps, or analysis of the data.

Another system, the Automated Regional Justice Information System (ARJIS) [1], developed as a web-based network of criminal justice agencies in San Diego County, is a complex criminal justice enterprise network utilized by 50 local, state and federal agencies in the San Diego region. The goal of the system is to promote better sharing of

information between police officers and the public. An article entitled “Automated Information Sharing?” [22] from the NIJ (National Institute of Justice) Journal in January 2006 reported a study showing that sharing crime information increases public safety as well as the safety of officers. The San Diego Sheriff’s Office (SDSO) reported that this information increases effectiveness and job performance as well as public safety, in general.

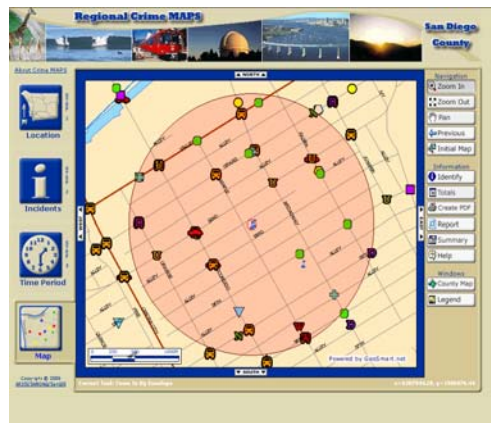


Figure 2.1 Screenshot of map service provide by ARJIS

The system provides a large range of services including an emergency digital information service that delivers official information about emergencies and disasters to the public. Other information includes the 10 most wanted criminals, online warrants, who’s in jail, crime statistics, and a graphical tool that allow users to browse crime information in the forms of a map, a statistical bar graph, or simple number figures. Users can access ARJIS to create maps based on crime types, period and region as shown in Figure 2.1. Although it provides rich information on the website available through any standard web browser, it has yet to provide any interface that can be accessed and viewed on a mobile phone.

The website Chicagocrime.org provides a user-friendly website that allows users to access a crimes database. We found that this database and its interface are the most advanced type of crime-searching tools. One of the most sophisticated features in the site is called “Crime along a route”. It is a map service developed with the Google map development toolkit with their crime log information. The service lets users create custom views of crimes from auto theft to bribery on maps. Users can draw a route by

clicking multiple points on a map, and then they can choose conditions to plot crime instances based on crime type, date, and time.

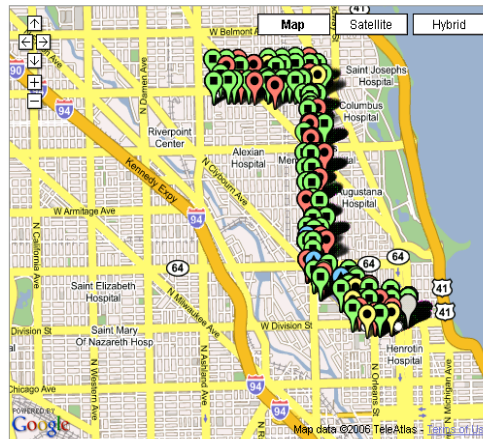


Figure 2.2 Screen shot of map service provide by Chicagocrime.org

Figure 2.2 is a result of using Chicagocrime.org to plot crime incidents from Nov 16, 2005 to July 10, 2006 with all types of crime. This map can be useful when a user plans to travel between one location and another, but the system doesn't provide automatic route planning based on the crime statistics. In addition, although users may be interested in incidents of crimes, it might be more useful for users to be aware of crime tendencies derived from statistical facts along the route. Finally, the site does not provide any interface for supporting mobile users to access crime information.

## 2.4 Reality Mining

Reality Mining [4] was a research project conducted by Nathan Eagle as a part of his Ph.D. work in the Human Dynamics group at the MIT Media Lab. His work's primary goal was to collect machine-sensed data from the real world, and to use the data to recognize social patterns in people's daily activities. This could be used to infer social relationships, identify social significant locations, and model group dynamics. In order to utilize Bluetooth's ability to detect nearby devices, he deployed 100 Nokia Bluetooth-equipped phones among Media Lab and other MIT communities to collect information representing proximity, location and time. He conducted this experiment for six months and collected over 500,000 hours (~60 years) of continuous data on daily human behavior.



From the study he discovered that although individuals have the potential for a relatively random pattern of behavior<sup>4</sup>, identifying and recognizing people's everyday routines and behavior patterns can be easily achieved; from the daily routines of driving home from work to weekly patterns. In addition, by comparing the collected data and surveys from users, he validated that the reported frequency of (self-report) interaction is strongly correlated with the number of logged BTIDs ( $R = .78$ ,  $p = .003$ ), and that the self-reported data has a correlation with the proximity data ( $R = .74$ ,  $p < .001$ ). The results strongly support the belief that Bluetooth detection is effective for measuring interaction in social activities among individuals.

Although this method is very useful for measuring and predicting social activities with nearby social peers, Bluetooth scanning only provides information about proximate Bluetooth IDs. Given only this BTID information, we cannot sufficiently identify the owner of each device as well as his/her peers. Thus, this approach is still left with the problem that a human has to collect additional identification information and identify each Bluetooth ID with each corresponding detected devices' owner. In addition, as Eagle mentioned in his paper Reality Mining [4], the computation for analyzing log data to infer social relationship and finding patterns of human behavior is not meant to be computed in a mobile phone. Although his work did not provide an optimized algorithm for mobile phones to do such tasks, his thesis described a simple algorithm for mobile phones for detecting a user's social interaction with his/her peers through BTID and location via GPS.

## **2.5 Route Finding Algorithm for Road Traffic.**

There are a number of system that support route planning, which use traffic congestion information to find optimal routes from the input of a user's defined destinations. This is a similar solution to the problems in data communication networks between routers and hubs, and in electronic power supplies through cable links between transmission towers; all of these solutions attempt to maximize the efficiency in time and energy to transmit something, whether it be vehicles, data, or electricity. Here, we looked

---

<sup>4</sup> The study employs a simple model of behavior for analyzing individual in three states: home, work, and elsewhere.

at systems and algorithms that relate to a route planning system, which is most relevant to what this thesis covers.

TrafficMaster [6] is a route planner system developed by John Fawcett and Peter Robinson in 2000 at Cambridge, U.K. as part of their research project. The system recommends routes that reflect the congestion anticipated in the future during the time of the journey. In addition, travelers are notified if the predicted congestion at the time of travel makes an alternative route preferable. The system collects congestion information from various sources, such as speed detectors and monitors mounted on bridges and major roads. It accumulates historic information, and also includes a live data service that informs its clients of unexpected congestions. The historical information includes location direction of traffic flow as detected by sensors, as well as average car speed. In addition to data from sensors, it also uses map information to find an optimal route between pairs of points for a given departure time by accounting for the connectivity and capacity of each road.

The system uses the Fast Lee [3] routing algorithm. Similar to Dijkstra's and Bellman-Ford's routing algorithms, Lee's algorithm finds the shortest path between two points using a trick known as wave propagation and back-trace. The algorithm simulates an inkblot spreading out over a piece of paper, centered on the start point. The area covered represents vertices already explored. The real time is known upon the traversal of each edge, so the algorithm can calculate the correct value of the delay, which is derived from accumulated heuristic data. When the destination vertex is reached, the algorithm traces back to the start to find which path leads from the start to the destination. Because vehicles continue to move along the road while the system is calculating routes, this enables an advantage for providing the most current information to users - the algorithm traces back from the destination to find the latest position of the car before returning the result.

There are a few resemblances between the TrafficMaster system and the route detection algorithm in this thesis. TrafficMaster calculates a route's condition based on a heuristics approach of using congestions. Traffic congestion is not static; it changes over time, fluctuating from morning rush hour, to very little congestion during the middle of the night. It is reasonable to apply this heuristic in the algorithm because the average

departure time is known. Similarly, our route detection algorithm predicts routes and destinations based on a user's historical transit seed and usage of routes based on the time of day. In addition, the detection of abnormality reflects the current detection sensitivity; this sensitivity is based on risk information which changes over time.

The route detection, however, does not involve finding new routes based on map information. Rather, it tries to predict where the user is heading based on the user's transit behaviors. This is unlike the TrafficMaster, which requires destination information to compute an optimal route between one's current location and the given destination. However, our route detection algorithm does not require destination information because the destination itself is the question that it tries to answer. In addition, the route detection algorithm does not have any street information other than what the users have provided through training routes on the system.

Although the scope of this thesis does not cover finding routes between two locations, the future work of the project involves using an algorithm that tries to compute optimal routes between a current location and a predicted location. Often, multiple routes exist between two locations, and people may have to make decisions about which route to take. These decisions will be supported by the system, as it will compute the conditions of each route, taking into account the travel time of day, predicted risks, total expected travel time on the routes, and location of people near the routes, who are peers and caregivers of travelers, and who may be able to offer help if needed. This future work will be described in more detail in the later concluding chapter.



## Chapter 3

### Where are you? – The Risk-Meter

Increasing people's awareness of potential crime in various neighborhoods is the first step towards enhancing the security and safety of each individual. Walking down an unfamiliar street for the first time, or trying to park a car or bicycle in a strange neighborhood can make one particularly vulnerable to crimes. *Contella* offers users the ability to learn about the safety of their surroundings in an attempt to more deeply inform their daily decisions.

Our approach to enhancing the awareness of crime was to build a system that measures a potential risk and provides this information to the concerned user. This system was built on top of an already existing system that enables simple GPS detection of position. As mentioned in the introduction, we attempted to measure the risks to personal safety based on a year's worth of crime logs around the MIT campus in Cambridge, MA. This information was collected and saved on a Geographic Information System (GIS) database. Our system renders crime statistics from the database in real-time at the user's request. Users can request a real-time risk analysis by providing the server with their location and information about their commonly transported valuables. Based on GPS coordinates, the time of a request, and users' information about their valuables, the system can calculate and communicate any potential risk to its users.

The goal of the system is to provide a simple mobile phone interface to retrieve just-in-time risk metrics at the click of a button. The following scenario describes how the system could be used.

#### 3.1 Scenario

Scenario 1:

Ann is a new freshman at MIT. She lives in an apartment off-campus, and it takes her about 20 minutes to walk to her lab. She likes to walk alone but she is very concerned about her safety and security. Whenever she feels unsafe, she grabs her cell phone and checks the Risk-Meter to see how safe her surroundings are. At times, she works late at

the lab and before she leaves, she checks the Risk-Meter to decide whether it is safe for her to walk home at night or whether she should take a shuttle bus which stops close to her house. She also uses the Risk-Meter to protect her belongings. When she rides her bike and needs to find a new place to park it, she checks her Risk-Meter to assess the prevalence of bike theft in her immediate vicinity. Sometimes she sees on the Risk-Meter that the risk to bikes is high in certain neighborhoods, so she tries to find a safer place to leave her bike. If she can't find one, she keeps her eye on her bike whenever she can.

Like most parents, Ann's parents are concerned about her safety and security. Because she moved far away from home to study, her parents are unfamiliar with her new city, which adds to their concern. Ann allows her parents to check her Risk-Meter remotely at the beginning of the school year, and her parents check Risk-Meter from time to time to see if she is all right. Although the Risk-Meter only measures risk based on historical crime statistics and does not report Ann's actual safety, psychologically it gives her parents a connection to their daughter.

## **3.2 User Interface**

The interface of the Risk-Meter consists of a menu for configuring and requesting risk analysis, and a graphical interface for visualizing the measured risk. This interface is a part of *Contella*, a phonebook client application on which the Risk-Meter was developed. We used J2ME's standard API on Motorola i870 iDEN phones to build a graphical user interface for *Contella*. In the following sections, first I will describe the basic *Contella* interface, and then I will illustrate the interface for the Risk-Meter, which is a part of the larger *Contella* system.

### **3.2.1 Contella**

The main menu that the user sees when *Contella* is first launched is a contact list. The user may use the phone's standard arrow keys to browse the phone book list. We can see the layout of the keypad shown in the Figure 3.1(b).

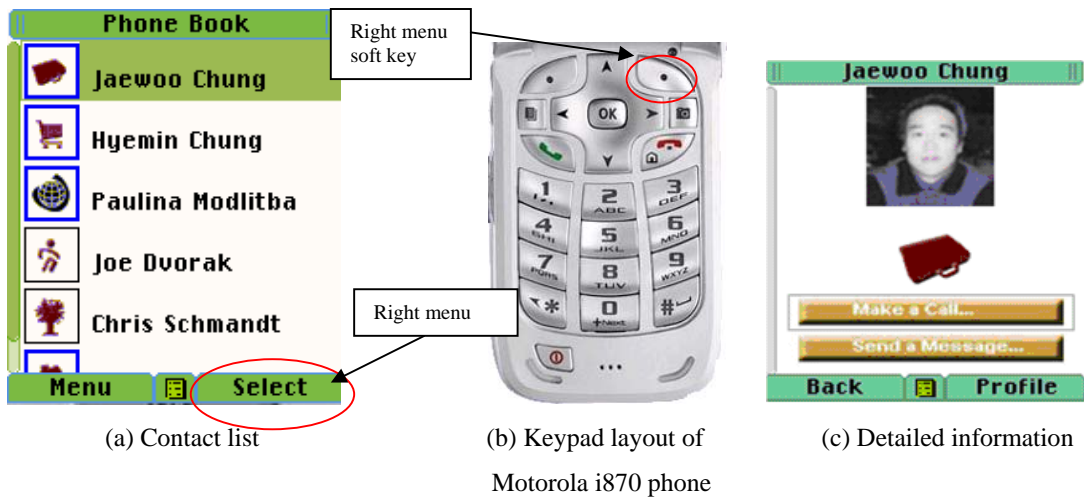


Figure 3.1 *Contella's* contact list and individual contact's detailed information screen

Boxes positioned on the left side of each entry are context icons. These icons can represent a number of different things, including each contact's current location (office, supermarket, park or home icons), whether a contract is currently moving (walking icon), and if a contact's status is unknown (globe icon).

*Contella* shows detailed information about the highlighted contact ("*Jaewoo Chung*" in <Figure 3.1(a)>) when the user presses the right menu soft key or the OK button on the keypad. The detailed information includes an image of the contact, the contact's contextual information, and buttons for communication with this contact.

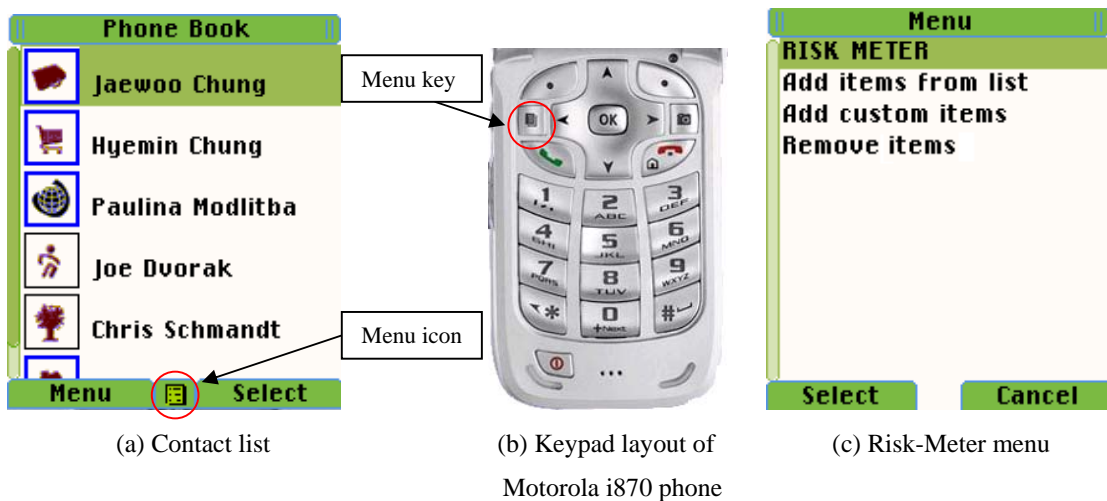


Figure 3.2 *Contella's* Contact list screen and keypad layout of mobile phone

By pressing the menu key (see Figure 3.2(b)), the Risk-Meter Menu will appear on the mobile phone screen (see Figure 3.2(c)). The following table describes the functions on the menu:

Name of item	Description
Risk-Meter	Shows risk of selected items
Add items from list	Lists a preset of at-risk items such as car, bicycle, and laptop
Add custom items	Enables user to type in a name of an item through mobile phone keypad
Remove items	Lets user remove items

Table 3.1 Menu description of Risk-Mete

As previously mentioned, users can set up a trust scheme, allowing other trusted users to have permission to access information. A master phone can access information from a slave phone, while a slave phone can request certain services from a master phone. Details for setting up the master and slave trust scheme will be covered later in this chapter.

The following figure illustrates how to access a slave phone user's risk information. As shown in the Figure 3.3(a), when the user selects a contact from the list to be a slave, the border of the context icon becomes blue.



Figure 3.3 Screenshot of trusted user and non trusted user



When users select a contact from the list, for example “Jaewoo Chung” whose context box border is blue, users can see a list menu on the bottom of the screen, as shown in the Figure 3.2(b). If the border is not blue, meaning that the contact is not set up as a slave, the list menu will not be shown at the bottom of the phone screen, can be seen in Figure 3.2(c).

Selecting the list menu will bring up a hidden list of commands that allow the user to access information and request tasks relating to the already selected contact. This list is shown in Table 3.2.

Name of function	Function description
<b>Risk Meter</b>	Requests and displays remote contact’s Risk-Meter
<b>Manage Location</b>	Add or delete remote contact’s labeled location
<b>Manage Routes</b>	Set a route for detecting transit patterns

Table 3.2. Commands that allow users to request tasks or access information from slave phones

### 3.2.2 Risk-Meter

For the design of the crime statistic’s visualization, we considered two kinds of graphical representations: a map and a bar graph. As seen in Figure 3.4, the map allows us to easily grasp the overall distribution of risk. However, the small screen of most mobile phones does not provide a fine enough resolution to easily communicate detailed crime information. In addition, it is hard to display multiple properties on a phone screen.



Figure 3.4 Crime statistics on a map and risk meter on a mobile phone screen

Although there are many different ways to display crime-related information on a map, for the reasons already mentioned, the Risk-Meter only displays risk in the form of a simple bar graph. This allows users to easily check specific items and see the measurement of risk in a graphical representation. Each row represents the risk attributed to specific items, such as “car” or “laptop”, and the red bar increases in length from left to right as the associated risk increases. The bar’s length is based on the statistical analysis of crime logs. When there is no red bar on the grid, it means no historical information exists in regard to the particular item. Crime information is calculated within 100 meters of the phone’s current location, at the current time of day, plus-or-minus one hour.

Initially, three items are listed: Laptop, Bicycle, and Car. According to the crime logs collected around the MIT campus, property theft is a major crime on campus. Specifically, laptops and bicycles are stolen with the highest frequency throughout campus. Users can also retrieve information about risks to other specific items. In order to achieve this, a user needs to provide information to the system about which items’ risks are to be measured. They can add and subtract items from their Risk-Meter by selecting a list of items or manually entering the item name through the phone’s keypad.

### 3.3 Trust and Communication

Scenario 1 described Ann’s parents accessing their daughter’s potential risk information. However, to protect the retrieval of private information by unwanted third parties, the system employs a simple trust mechanism that requires specific settings on both the master and slave phones. Therefore, to establish a trust between Ann and her parents’ phones, Ann must set her father’s and mother’s contact profiles as masters and her father and mother must set Ann’s contact profile as slave. These demarcations indicate whether we allow each contact to request certain information and activate certain functionalities, such as accessing risk information from remote phones.

An interface for setting the trust between master and slave phones is provided in the “Edit Contact” screen, as shown in the Figure 3.5. By pressing the right soft key to select “Profile”, as shown in Figure 3.1(c) in the previous section, users can access the Edit Contact screen. This screen allows users to edit the selected contact’s information. The screen includes basic information such as a phone number, IP and email addresses, and the trust configuration which sets the contact as master or/and slave. By pressing either the up and down arrow buttons on the keypad, users are able to navigate through the text field entries and check boxes. When a field or check box is highlighted, a user can use the number key to enter characters or to select or deselect the check box.



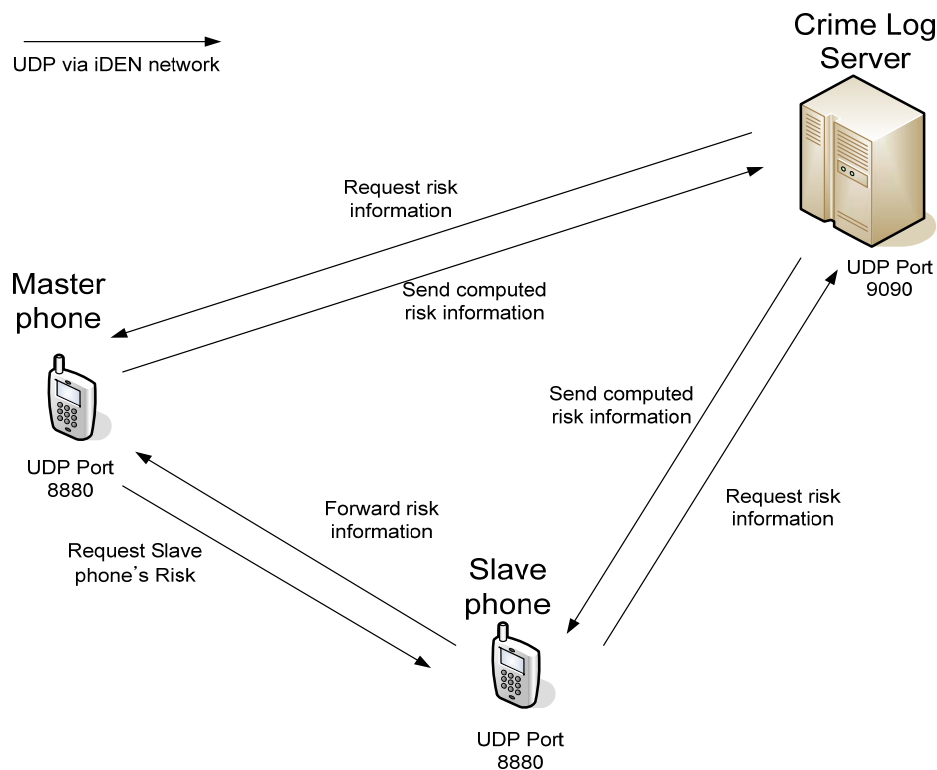
Figure 3.5 Edit Contact screen

Once trust is established and a request is issued, the *Contella* application checks the message header of the request in a UDP packet to confirm the appropriate status

necessary to fulfill the request. When a master phone requests remote access to a slave phone, the master phone includes its IP address, Bluetooth ID, and phone number in the header of the request message. The slave phone that receives the request message first checks if the requester's information matches with an existing master's information. When the information matches, the slave phone accepts the request and performs the task that the master phone requested.

The communication between the phones is accomplished via the iDEN<sup>5</sup> network. In the US, Nextel provides a static IP address for each Nextel-subscribed iDEN phone, and it allows the phone to communicate through this unique IP address. In *Contella*, most of the data communication use UDP over this network in a peer-to-peer manner between mobile phones.

As shown in the Figure 3.6, there are two ways to retrieve risk information: a master phone can request it directly from the crime log server, or it can request it from the slave phone.




<sup>5</sup> Integrated Digital Enhanced Network (iDEN) is a mobile communications technology, developed by Motorola, which provides its users the benefits of a trunked radio and a cellular telephone. iDEN places more users in a given spectral space, compared to analog cellular systems, by using time division multiple access (TDMA). In US, Nextel provides static IP address for each Nextel subscribed iDEN phones, and it allow phone to communicate through this unique IP address.

Figure 3.6 Communication between master phones, slave phones and crime log server

When any user requests risk information about his/her location, the Risk-Meter directly requests the risk information from the Crime Log Server using UDP over iDEN. The server computes the risk for the requesting party and returns the result. However, when a master phone requests a slave phone's potential risk information, as opposed to its own risk information, the master phone must first send this request to the slave phone. When the slave phone accepts the request from the master phone, it then requests its own risk status from the Crime Log Server. The resulting computed risk information is sent back to the slave phone, and the slave phone then relays the information to the master phone.

### 3.4 System architecture of Crime-Log Server and Risk-Meter in Contella.

The Crime Log Server is a GIS database that contains crime information collected by the police. Table 4.3 illustrates an example of a log collected on Oct 14-16 2005, provided by the MIT police.



**MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
POLICE DEPARTMENT**

Press Log

October 14 – 16, 2005

Date & Time Reported	Inc Type	Date & Time Occurred	Address	Comments	Disposition
2005-10-16 18:55:09.0	MISC	2005-10-16 18:50:00.0	M32 / 32 VASSAR ST	PATROL OF STATA CENTER	CLOSED
2005-10-16 05:28:02.0	UNKTROUB	2005-10-16 02:00:00.0	W31 / 120 MASSACHUSETTS AVE	ASSIST OTHER POLICE AGENCY	CLOSED
2005-10-16 03:20:02.0	LARCENY	2005-10-16 01:00:00.0	W7 / 362 MEMORIAL DR	R/P STATES WALLET CELL PHONE AND JACKET STOLEN SOMETIME THIS EVENING.	CLOSED
2005-10-16 02:25:22.0	FIRE ALM	2005-10-16 02:25:00.0	PKS / 530 BEACON ST	ALARM MALFUNCTION ON FOURTH FLOOR, BOSTON FIRE DEPARTMENT INVETSIGATED.	CLOSED
2005-10-15 23:37:51.0	SUSP ACT	2005-10-15 23:37:00.0	E23 / 25 CARLTON ST	R/P STATES SEVERAL INDIVIDUALS PUSHING BOOTH FROM TROLLY TOURS DOWN CARLTON ST, UNITS MAKE CONTACT WITH SUBJECTS.	CLOSED
2005-10-15 15:50:22.0	SUSP ACT	2005-10-15 15:30:00.0	M10 / 122 MEMORIAL DR	PARTY REPORTED A MAN HITTING A FEMALE OUT SIDE OF 77 MASS AVE.	CLOSED
2005-10-15 09:49:01.0	HOMELESS	2005-10-15 09:49:01.0	M10 / 122 MEMORIAL DR	HOMELESS PERSON IN LOBBY 10, TRESPASS WARNING ISSUED.	CLOSED
2005-10-14 18:05:47.0	LARCENY	2005-10-14 01:00:00.0	STUDENT HOUSE / 111 BAY STATE RD	RP REPORTS A DVD PLAYER AND AMPLIPHER WERE STOLEN.	CLOSED
2005-10-14	ATT B&E	2005-10-11	E60 / 30 MEMORIAL DR	RP REPORTS AN ATTEMPTED B&E AT HIS	CLOSED

Table 3.3 Crime log collected and published by MIT Police

As we can see in Table 3.3, the report consists of an incident type, the date and time of the occurrence, the location of the incident and any further comments from police officers. Although the logs were collected systematically by the police, they lacked the GPS coordinates required to compute risk based on location. Thus, we needed to convert each address into GPS coordinates; this allowed us to organize crime information based on location. The locations of reported crimes are based on the building code, so we had to build a table mapping building codes to latitude and longitude.

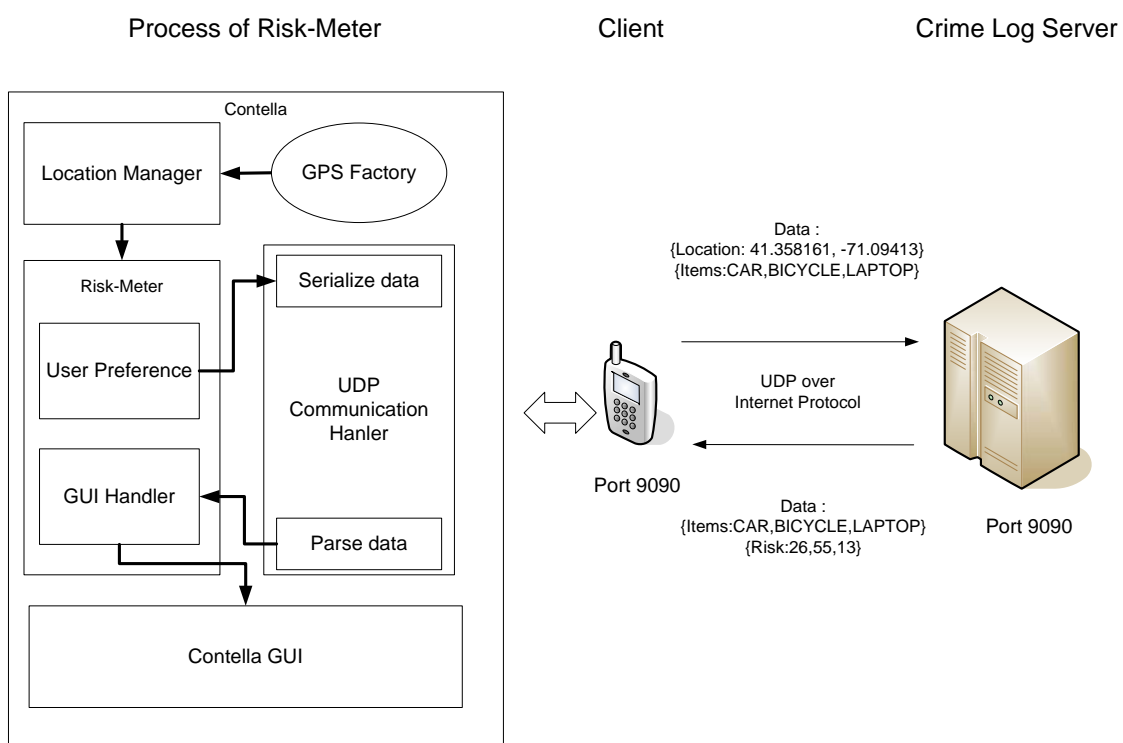


Figure 3.7 System architecture of Risk-Meter in *Contella*

*Contella's* client request data contains location information in a series of strings, and a list of item names. The location is provided from the GPS Factory, which is dedicated to processing incoming GPS data, and passing it on to the Location Manager. The Risk-Meter collects the GPS information from the Location Manager, which handles all location-related services in *Contella*. Location information consists of GPS coordinates in decimal degrees. The Risk-Meter then sends the coordinates, along with a list of items based on a user's preference, to the UDP Communication Handler. The

Handler serializes the data and sends it to the Crime Log Server. The data is sent in UTF strings in UDP packets. The example of data sent from a client is shown as follows:

*{Location:41.358161,-71.09414}{Items:CAR,BICYCLE,LAPTOP}*

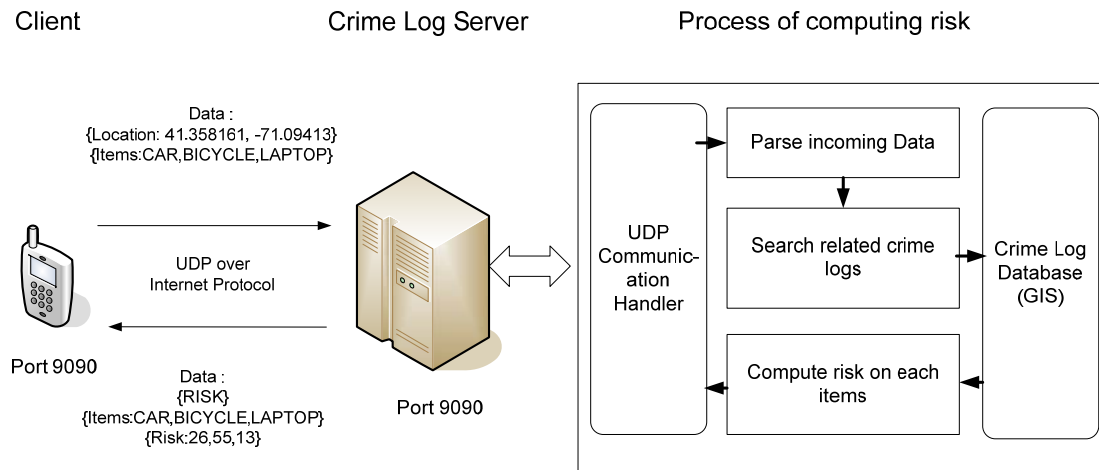


Figure 3.8 System architecture of Crime Log Server

The Crime Log Server is dedicated to serving risk information. It handles and parses incoming information, and then sends the computed results back to the client. The incoming message is handled by the UDP Communication Handler<sup>7</sup>. When the UDP packet arrives, the Communication Handler parses data into lists of strings. The server expects to receive location data and a list of items in order to compute risk.

The algorithm for assessing risk is based on time, location and item names. First, the algorithm searches for the total number of each incident related to a given item from the Crime Log Database. It searches by the name of the item that appears in the database. To increase the hit rate of matching words, we employed a table that expands each word into its synonyms. For example, “car” is expanded to “vehicle” and “automobile”.

We have two search ranges for calculating the risk. For location, we use a 100-meter search range from the given location. For time, we have a window of plus-or-minus one hour from the requested time of day. When the algorithm searches the

<sup>7</sup>In our examples in the thesis uses UDP for communicating with phone client. Although a UDP packet can be lost in the middle of transmission, data communication in iDEN network works more reliably with UDP.

database, it also calculates the maximum number of incidents based on time of day within a 2-km range. The resulting risk is normalized based on the maximum number of incidents on a scale of 100.

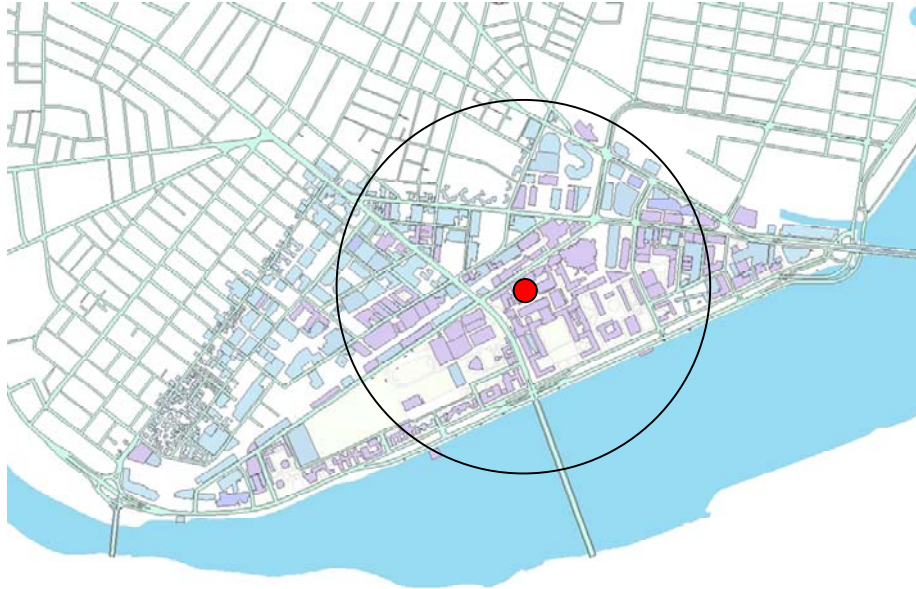


Figure 3.9 Circles of diameter 100m and 2km

After each item's risk is computed, the information is serialized and sent back to the client via the UDP Communication Handler. An example of data sent from a client is shown as follow:

```
{RISK}{Items:CAR,BICYCLE,LAPTOP}{Value:25,36,14}
```

The client receives a UDP packet from the UDP Communication Handler and it forwards the message to the GUI Handler in the Risk-Meter. Users are able to see this information in the form of a bar graph on their phone screens.



## Chapter 4

### Where are you going – Route Detection Algorithm

This chapter discusses a way to learn a user's traveled patterns and detect certain abnormalities such as deviation from average transit time and unusual destination. In order to achieve this, the system must learn about a user's travel routes and be able to discriminate between abnormal journeys and the normal transit patterns learned by the system during a training phase. The algorithm in the system is restricted to detecting recurring routines such as going to school, to the office, and to home. Although deviations from averages do not always imply incidents, it is helpful to provide this information to caregivers. In addition, a trusted user, i.e. a caregiver, should be able to assist the care-recipient through slave phones remotely, to help training routes, setting bookmarked places, and specifying a route to be detected and conveyed to the caregiver through the master phone.

The sections in this chapter will describe the use scenario, interface, and the algorithm for learning and detecting route transition pattern. The following is a simple scenario that describes how the system could be used.

#### 4.1 Scenario

Mary is an elderly woman who lives in a nursing home. Although Mary has a mild Alzheimer's affliction, she is very independent and likes doing grocery shopping, walking a dog, or visiting the doctor by herself. Mary has a grownup son who dearly cares for his mother, but he works in a different town and is not able to visit her that often. However, with the help of *Contella*, he checks his mother's location and makes sure she is safe and on her schedule. Because of her Alzheimer's, she sometimes forgets what she is supposed to do, and her son checks her schedule from time to time to see, for example, if she, for example, has left home early enough to get to an appointment with the doctor. Also, her son gets an alert whenever she deviates from her regular path or she halts on the way. This gives her son more opportunities to help his mother.

Although she has very few activities outside the nursing home, occasionally she discovers a new path while walking her dog or finds new shops to regularly visit. Mary's son remotely sets two interesting locations between which Mary's phone will automatically collect any new route information. Mary walks between the locations as usual and she does not need to be aware that her *Contella* is collecting training information. For the new grocery shops Mary discovers, her son asks her to tell him the location in the terms of streets or addresses, and he remotely enters the location into Mary's phone. So, when Mary walks down to her new grocery store, the route will be learned by her phone.

## 4.2 User Interface

As described in the scenario, *Contella* provides an interface that allows the application users to monitor their loved ones' locations and transit status. It also allows users to bookmark locations and set *Contella* into a training mode to collect route information between bookmarked locations. Users can do this locally as well as remotely. The following sections describe the contextual information display, settings for bookmarking and route learning, sending and receiving alert messages and assisting at a remote care-recipient.



Figure 4.1 Screen shot of *Contella*. Context boxes appear next to each contact entry.

#### 4.2.1 Contextual information

As described in the previous chapter, boxes positioned on the left side of each entry are context icons. They allow users to share context information in order to enhance the awareness of remotely located loved ones. The color of the context box border indicates whether you have established a trust relationship with the contact. Inside the box, an icon represents each contact's location information. The icon represents three states: at a bookmarked location, moving, and status unknown. When a user is at a previously identified (i.e., bookmarked) location, the associated location icon will appear, such as office, supermarket, park, etc. When users are not in a previously specified location, *Contella* considers the user to be in transit and the "walking" icon will appear in the context box (see "Joe Dvorak" on the list in Figure 4.1). In addition, when a user's location is not certain and his/her status is unknown, it shows a globe (as shown next to contact "Paulina Modlitba" in Figure 4.1).

*Contella* provides more information about contacts' transit status in the detailed information screen as seen in the Figure 4.2.



Figure 4.2 Screen shot of a contact's detailed information

The detailed information screen, from top to bottom, consists of an image of the contact, contextual information and two buttons for communication. These buttons allow

the user to make a phone call or send messages to the currently selected contact. The contextual information shows detailed information about the moving contact. For example, Figure 4.3 illustrates the meaning of the possible contextual information.

Through the route detection algorithm, *Contella* predicts its user’s location status. Alternatively, if it can not predict, it will still attempt to provide facts about the user’s location. When a *Contella* user is moving, *Contella* compares previously collected route information and tries to predict the user’s destination and estimate time to arrival. As seen in Figure 4.3, when a remote contact left home and is moving but the algorithm can not predict the contact’s destination, a user sees the icons labeled “B” in detailed information of the remote contact. When the algorithm can predict the destination and compute the estimated time to arrival, the user will see the icons labeled “D” on the screen.

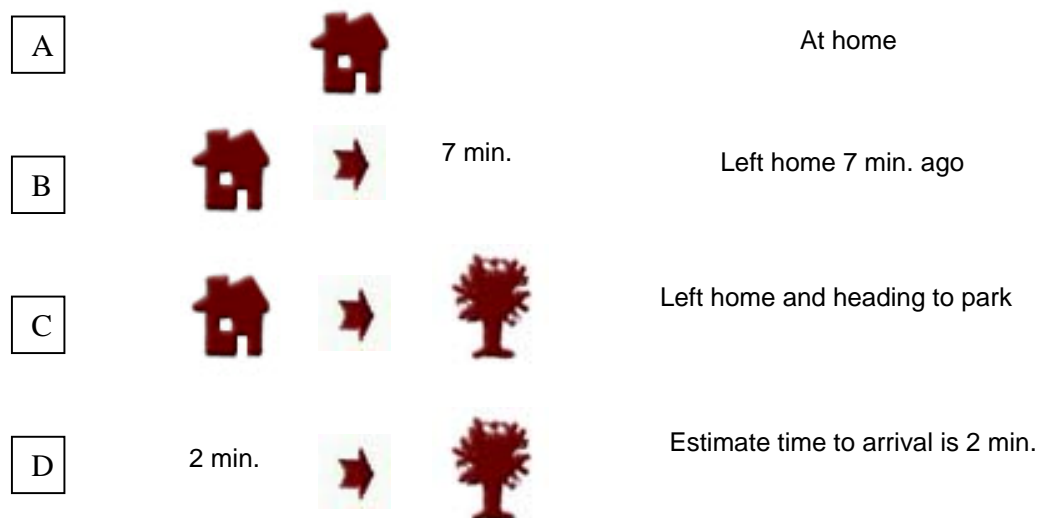


Figure 4.3 Graphical representation of user’s location status.

#### 4.2.2 Settings

*Contella* provides a list of options, through which users can bookmark locations, set the system into training mode (to collect route information), and adjust the threshold for detecting route transition abnormalities.

From the contact list, a software button “Menu” (*see <Figure 4.1>*) will display a list of items that are mostly related to location awareness functionality. Table 1 describes

each item and its function. Some of the functions will be covered in more detail in Section 4.4.

Name of item	Function
<b>GPS Reception Monitor</b>	Shows the status of current GPS reception.
<b>Where am I? (Set Location)</b>	Users may choose to enter their current position explicitly. This option brings up the list of labeled locations which user previously defined.
<b>New Contact</b>	Allows the user to create a new contact
<b>Mark Location</b>	Allows the user to label current position.
<b>Location List</b>	Lists the names of previously marked locations.
<b>Training: {FALSE TRUE}</b>	Selecting this item toggles the training mode to false or true. This menu sets <i>Contella</i> 's route training mode. When it is set to 'TRUE', <i>Contella</i> begins to collect route information while the user is moving.
<b>Routes</b>	Lists routes which were previously trained by user.
<b>Detection Threshold</b>	It allows the user to change the route abnormality threshold.
<b>Preference</b>	This item contains a menu that is used for debugging purposes such as dumping collected data to server.
<b>Exit</b>	Quit from the <i>Contella</i> application.

Table 4.1 Function description of sub menu.

### 4.2.3 Communication

As described in the previous chapter, *Contella* uses UDP over the iDEN network to communicate between master and slave phones in a peer-to-peer manner. In this section the communication used between the master and slave phones is divided into two parts: sending alert message to the master phone and providing remote assistant to a the slave phone.

#### 4.2.3.1 Alert Messaging

When an abnormal route is detected and it exceeds a threshold set by the user, the slave phone sends alert messages to its master phones. Initially, the slave phone searches the phone book in *Contella* to find contacts that are setup as masters. Once the search for

finding emergency contact finished, the slave phone starts to distribute alert messages to its master phones (the phones of caregivers). When caregivers get the alert message, their phone vibrates and makes an alert sound. On the *Contella* phone screen, information about the abnormal activity of the contact whose phone sent the alert message will appear. This is shown in Figure 4.4.

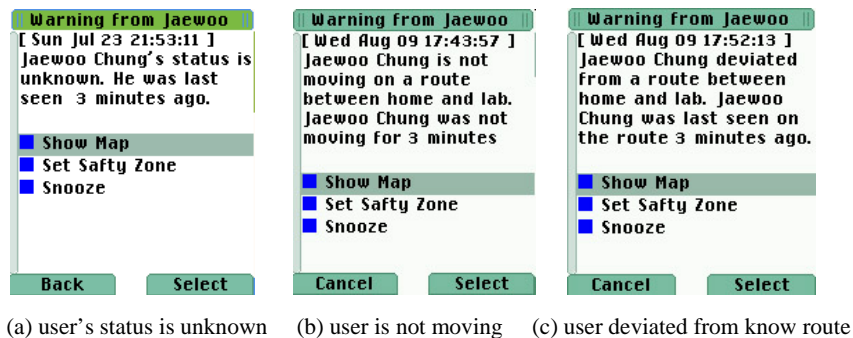


Figure 4.4 Screenshot of status report.

The status information shown on the screen is generated by the care-recipient's *Contella*, and it contains information about when the care-recipient deviates from a route, how long the care-recipient has stopped moving, or the last moment the care-recipient was observed at known routes or locations. There are three options that the caregiver has in reacting to the message; a button for accessing map to view care-recipient's location, a button to snooze (to temporally discard) the alert, and a button to temporally set the care-recipient's current location (and the surrounding 100 meters) as a safety zone.



Figure 4.5 Screenshot of a map displaying location of remote care-recipient

The 'Show Map' option of the status report screen allows care-givers to locate their loved ones. When the caregiver assesses the situation to be safe, they can set the

care-recipient's current location to be in safety zone temporarily. However, when the location is bookmarked, the location becomes permanently set as a safety zone.

#### 4.2.3.2 Remote Assistance

The Scenario described earlier in this chapter explained how Mary's son remotely accessed Marry's phone and added locations. In addition, he set up two locations so that route information is automatically collected while Marry travels between them.

Three main functions are provided in order for care-givers to assist loved ones; specifying a route for detection, assigning semi-supervised route training between two labeled locations, and transferring labeled locations from master phones to slave phones. From the hidden menu (*see <Figure 4.6(b)>*), caregivers are can access and execute these functions remotely from master phones. This allows a care-giver to provide those functions to care-recipients who are not keen at using technology, such as young children or elderly people (like Mary, the character mentioned in the scenario above).

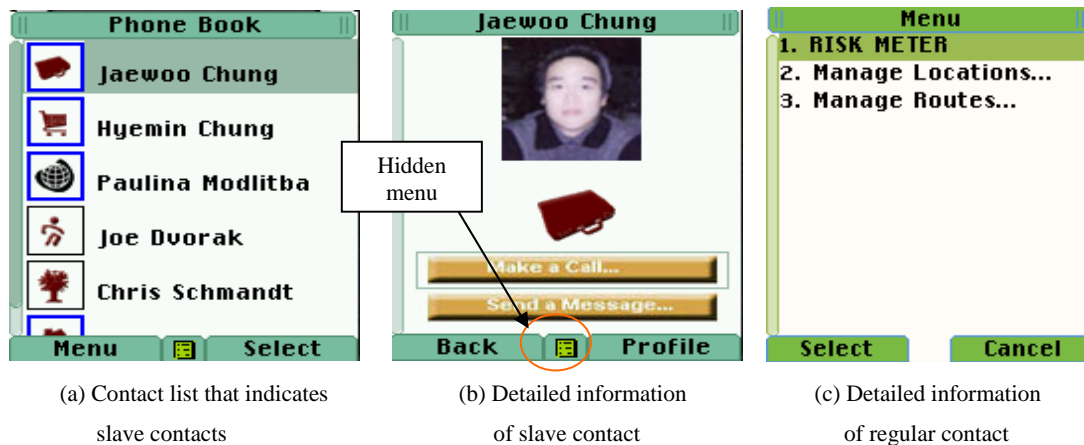


Figure 4.6 Screenshot of trusted user and remote assistant menu.

The “Manage Location...” function allows a user to remotely manage a slave phone's labeled locations. For example, they can remotely transfer a location (name and GPS coordinates) stored in their master phone to the slave phone, or add a new labeled (bookmarked) location into the remote slave phone by providing GPS coordinates.

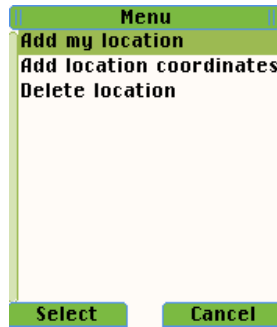
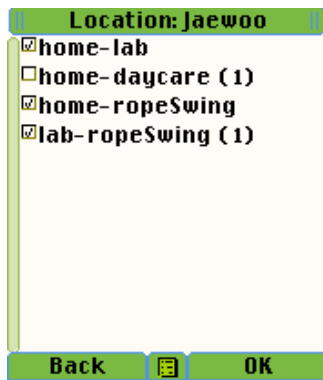
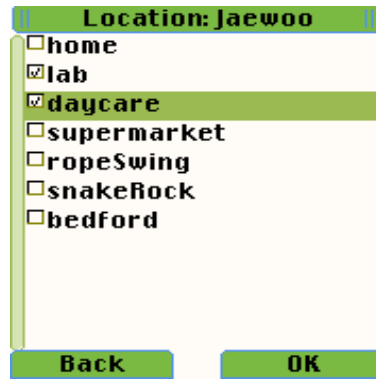


Figure 4.7 Screenshot of Manage Location menu.

The “Manage Routes...” function allows a user to remotely assign a route to be detected in the slave phone as shown in the Figure 4.8. The items on the list in Figure 4.8 (a) represent route information that was previously collected by remote care-recipient. The small check box indicates that a route, for example “home-lab”, is marked as one to use when detecting abnormalities in the care-recipient’s travels. Once the caregiver remotely sets a route to detection mode through their master phone, the slave phone will start to detect the slave phone user’s transit pattern and report their status to master phone.



(a) Route information of slave phone



(b) Assigning two locations in slave phone

Figure 4.8 Screenshots of slave phone’s route list and bookmarked list.

A number in brackets on the right side of the route (*see <Figure 4.8 (a)>*) indicates a route between the two locations is scheduled to be collected. The number indicates the numbers of trials to collect along that route. A master phone can also remotely set two labeled locations for automatic collection of route information. The slave phone will then collect this information whenever the slave phone user travels



between the two locations, as shown in figure 4.8(b). This allows the slave phone user to train routes without any knowledge of how to operate the route training mode. The slave phone user can even be unaware that their route is being collected by their own phone.

### 4.3 Design Consideration on Route Learning and Detection Algorithm

After Natalia Marmas's work on *WatchMe*, as mentioned earlier, in the related work in Chapter 2, we decided to experiment with a phone-only version, using the same GUI running on an ordinary phone, and using only GPS for context.

There were a number of difficulties in accomplishing this. The start up process was cumbersome; users had to collect several days of data which could then be analyzed for possible significant locations, and once they were named it might be necessary to collect another week's worth of data to classify the routes. In the mean time, batteries died due to the volume of data transmitted and the software sometimes hung while trying to re-establish broken connections (this was eventually fixed by switching from TCP to UDP). Also, having a server component added a layer of complexity. We therefore decided to build a modified version of *WatchMe* that substituted the elegance of the watch for normal phone menus and input, with route classification and detection running entirely on the phone.

We experimented with different methods of route detection in order for the detection algorithm to work with the limited computation power available in mobile phones. The challenge of developing a route detection algorithm was making the algorithm simple enough so that it could fit into a mobile phone – we used J2ME on Motorola's i860 and i870 phones as our main development platform. But at a deeper level, the challenge was to obtain accurate route descriptions more or less on the fly. *WatchMe's* use of a server allowed a week's worth of data to be stored for later analysis, which allowed for better detection of the true location from the frequently noisy GPS data.

The following sections detail the learning and prediction techniques (endpoint detection, route identification, and route prediction) used in *Contella*.

### 4.3.1 Endpoint Detection

Whenever a good AGPS<sup>10</sup> fix is available, *Contella* allows users to label locations and save them as landmarked places. From the phone menu's "Mark location" item, a user can choose a label from a predefined list. The label can be changed later to any name the user desires. The area within a 70-meter radius of the recorded position is regarded as the labeled place. When a user is within this radius, *Contella* recognizes that the user is currently in a labeled place.

Although a 70-meter radius is quite a large area, we find that the detection range is a reasonable size. When the labeled place is large, such as a school or an office in a big building, for example, there may be a significant distance between the marked location and the user's actual location. Thus, if the detecting radius is not large enough, the user's perception of the space would not match with *Contella's* location detection. Although the user might think they are still in the same area, *Contella* will not recognize the user as being in the labeled location. This problem is exacerbated by the fact that the GPS precision error can easily exceed 40 meters.

Another problem related to detecting location is determining whether a user departed from a labeled location. For example, when a subway station is close to an office, and a user who walks out from the office to take the subway hasn't been exposed to GPS satellites long enough to get coordinates, *Contella* will fail to detect the user's location, when the detection algorithm solely depends on GPS data. We solved this problem by checking cell tower IDs to see if the current cell tower ID matches the cell tower IDs associated with labeled location. Cell tower IDs are initially collected when a user labels a location. When a new cell tower ID appears while the GPS coordinates are currently available and the user is in a labeled location, *Contella* automatically collects the cell tower ID and associates it with the stored location. If the current cell tower ID is not found in the collected cell tower IDs associated with the assumed current location, then *Contella* changes the user's status as having departed from the previous labeled location.

---

<sup>10</sup> Assisted GPS (AGPS or A-GPS) is a technology that uses an assistance server to cut down the time needed to determine a location using GPS. The GPS receiver communicates with the assistance server that has high processing power and access to a reference network in order to get a faster result.

### 4.3.2 Route Identification

In order to design a route learning and detection algorithm that is simpler and requires a smaller amount of computation than WatchMe, we came up with an algorithm based on intermediate point detection. Intermediate points (*iPoints*) are locations that are interpolated between two labeled locations. If we compare the route to a subway train's route, the stations where the user enters and exits the train are the labeled places, and stations between these endpoints are *iPoints*. Because each set of *iPoints* are associated with a route, detecting an *iPoint* and the direction of travel will infer which route the user/passenger is taking and where he/she is heading. As in a subway train's routes, multiple routes and destinations can be associated with each *iPoint*.

Training mode is selected from a menu. While training, when a user leaves from a labeled place, *Contella* generates a new route template and begins to collect GPS data until the user arrives at another labeled place. A new location can also be labeled while in training mode, and this labeled location is used as a destination, or endpoint, of the newly collected route. Every 50 meters, as shown in Figure 4.9, several pieces of information are collected, including GPS coordinates and their accuracy to be used as *iPoints* and cell tower IDs associated with the *iPoint*. In addition, each template contains the elapsed time interval between *iPoints*, which has been collected during the training. This time interval is used to calculate the estimated time to arrival.



(a) Plot of raw GPS route data received  
Every five seconds



(b) *iPoints* collected every 50 meters from GPS  
coordinates received by *Contella*

Figure 4.9 Collected GPS coordinates by *Contella*

As the user progresses on his/her path, existing *iPoints* may be encountered, and these *iPoints* are registered on the newly generated route template. Thus, *Contella* avoids having redundant *iPoints* when multiple routes share the same points as shown in the Figure 4.11(a) in the later section. The size of the data of a 90-minute, 9 km. trip route template and its *iPoints* is about 2.5 KB. The data structure in the current system is not yet optimized, so we expect the data to be even smaller when optimized.

A route template is used to detect travel in either direction along the route. Although a route is trained in one direction, the *iPoints* are identical when the user takes the same path in the opposite direction. In the route templates, the order of the *iPoints* collected during training is preserved. When two consecutive *iPoints* are detected, *Contella* will find a route that contains these two *iPoints*; the order of the *iPoints* will determine the destination.

**Improving routes by re-sampling *iPoints*:** The route template is mainly used to determine the destination and the ETAs (Elapsed Time to Arrival). In a good route template, *iPoints* are dispersed by about 50 meters equally throughout the template. However, when the quality of GPS reception is poor, we observe noise and gaps in route templates as shown in Figure 4.10

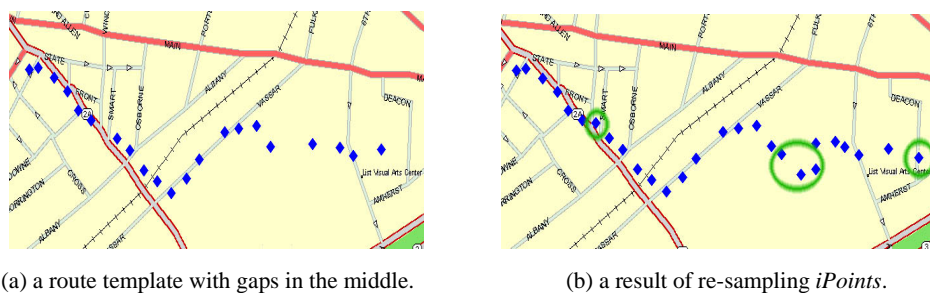


Figure 4.10 Improved routes by re-sampling *iPoints*

In order to fix this problem, either the route should be re-trained or additional information should be applied to enhance the reliability of the route template. In *Contella*, we solved this problem by automatically re-sampling *iPoints* where gaps exist in the templates. As shown in Figure 4.10(a), an initially trained route template could

have gaps in it because of various conditions such as speed, trees, and buildings. While a user is on a known route, *Contella* examines the distance between *iPoints* on this route. When a distance between consecutive *iPoints* is greater than 100 meters, *Contella* starts to collect new *iPoints* and fills in the hole in the route template on the fly.

Additionally, as GPS data provides an accuracy range in each GPS fix, *Contella* compares the accuracy range of currently received GPS data with the accuracy of *iPoints*. Some loss of accuracy is due to static obstacles, such as tall buildings, but other times it is due to more transitory factors such as the ever-changing visible satellite constellation, seasonal foliage, placement of the phone/GPS receiver on the body, and speed of travel. When the currently received GPS accuracy range is smaller than the *iPoint* where the user is located, *Contella* replaces the original coordinates of the *iPoint* with the most recently received coordinates. This improvement helps route templates to be reliable and eliminates the need to re-collect entire route templates when the initial one is poor. This re-sampling capability could also be extended to create detour routes within an existing route.

### **4.3.3 Route Prediction**

When a user leaves from a labeled location, the route template candidates that are associated with the location will be selected and used to search for *iPoints*. *Contella* receives GPS coordinates every five seconds, and tries to match these coordinates with *iPoints* from the route candidates. The search number of adjacent *iPoints* is limited to 15 for each route. When there is an *iPoint* near the currently received GPS coordinates within a radius of 35 meters, *Contella* is able to detect it. Because *iPoints* are equally dispersed 50 meters apart, it is less likely to detect two different *iPoints* within a 35-meter range. The average width of tested streets is within 35 meters and the detection range is large enough to include both pedestrian sidewalks.

As described previously, when two consecutive *iPoints* are detected, the algorithm will find a route that contains these two *iPoints*. From this route template, it is possible to determine the destination, and calculate the estimated time of arrival.

When multiple routes are found with different destinations, *Contella* will fail to choose a route by solely depending on *iPoints* to determine the destination. Similar to the subway station route example, a station could be part of multiple routes, but a timetable is used to discriminate routes and destination. Similarly, *Contella* keeps track of users' transit time patterns in templates to decide which are the most likely route and destination that the user is taking. For example, one of our *Contella* users leaves from *home* to head to the *office* usually between 9 am to 12 pm, and returns *home* between 7 pm to 9 pm. He sometimes goes to the *supermarket* around 8 pm. There is about a 50% overlap of the routes from *home* to *office* and *home* to *supermarket* as shown in Figure 4.11(a). When the user takes these routes, *Contella* checks the departure time and increments a number into a corresponding slot in a timetable that is kept in each route template; the timetable has eight slots, each representing a three-hour time interval (e.g. 9AM – 12PM, 12PM – 3PM, etc, covering the 24 hours of the day). By looking up the timetable kept in the route templates, and comparing the numbers between the two routes based on the time of day, *Contella* predicts the user's destination. If the user left *home* in the morning, *Contella* predicts that the user is going to the *office*. On the other hand, if the user left *home* in the afternoon, it will predict that the user is going to the *supermarket*.

```
(home ==> supermarket around 8 pm.)
Time      Location      Prediction
20:43:39  42.362282, -71.084266, home,0 min., , ,
20:43:47  42.362453, -71.084336, home,0 min., , ,
20:44:10  42.362818, -71.084693, home,0 min., supermarket,6 min.
20:44:44  42.362690, -71.085104, home,0 min., supermarket,6 min.
20:45:13  42.362725, -71.085637, home,1 min., supermarket,5 min.
20:45:50  42.362812, -71.086722, home,2 min., supermarket,4 min.
(home ==> lab around 10 am.)
Time      Location      Prediction
10:14:18  42.362274, -71.084229, home,0 min., , ,
10:14:23  42.362346, -71.084288, home,0 min., , ,
10:14:35  42.362648, -71.08476, home,0 min., lab,6 min.
10:14:58  42.36282, -71.08527, home,0 min., lab,5 min.
10:15:26  42.362797, -71.08579, home,1 min., lab,4 min.
10:15:54  42.362843, -71.08633, home,1 min., lab,4 min.
```



(a) Route path from *home* to *office* and *supermarket*.  
Black diamond dots are *iPoints* where two routes were merged



(b) Indexing *iPoints* based on cell tower IDs.

**Figure 4.11** *Contella's* route templates

**Indexing *iPoints* in route template:** When a matching *iPoint* is not found on the first pass, as just described, *Contella* tries to match every existing *iPoint*. If the number of *iPoints* is small, it is reasonable to search all the *iPoints*. However, if the number is large, significant computation is required to compare each *iPoint* every five seconds, which violates a design constraint. In order to stay within constraints, we used cell tower IDs as reference points to index *iPoints*. In other words, *iPoints* are partitioned into groups based on cell-towers, as shown in Figure 4.11(b).

*Contella* will only search for *iPoints* within a group defined by cell tower range when it fails to find *iPoints* from previously known routes. By searching this small space, *Contella* can find nearby routes that weren't part of previously detected routes. For example, suppose a user left location A, and took a route connected to location B, but then the user changed his/her mind and cut into a route connecting C to D. Because route A-B is not connected to route C-D, the initial search of finding *iPoints* will fail. Instead of attempting to search through the entire set of *iPoints*, the algorithm will only look for *iPoints* that are associated with current cell tower ID. This will allow *Contella* to efficiently reduce any unnecessary searching. If a part of route C-D is in the same cell-tower range area, *Contella* will detect nearby *iPoints* and predict the destination and estimated time to arrival. If *Contella* does not find any *iPoints* for several minutes, it will simply return the elapsed time from the location of departure.

**Estimating time to arrival:** The simplest way to estimate the time to arrival is to calculate the current speed and apply it to the distance that remains to the destination.

Distance can also easily be calculated by summing the distance intervals between *iPoints* to the destination. Although, a user's average speed varies slightly, perhaps within 10 percent, the estimated time of arrival fluctuates a great deal whenever the distance to the destination is large. Particularly when walking, the GPS precision error can easily range from 20-30 meters, causing the speed of walking to vary more than 30 percent. This results in a bad estimation of arrival time.

Another way to estimate time of arrival is to collect a user's interval of elapsed time while the user is taking a particular route. This time interval can be collected every time the user takes a particular route, and be applied to the previously stored average. When user's transit speed pattern is regular, we can use this average to estimate the time, and the result will be quite reasonable. If the ETA depends on the sum of remaining interval time between *iPoints*, the ETA will shrink quickly as the user's speed increases and he/she encounters *iPoints* at a more rapid rate. However, if the user changes his/her means of transportation, such as switching from walking to bicycling or driving, the estimated arrival time will not be correctly calculated.

Thus, in *Contella*, two factors are considered in calculating the estimated time of arrival: a user's average transit time intervals (TTI) and his/her current speed. TTIs are collected initially when a new route is constructed. This initial interval is used to construct blocks of *iPoints* where the speed of transition is similar. For instance, one of our *Contella* users takes a route with multiple kinds of transportation, including driving a car, walking and riding the subway. The route length between the subject's *home* and *office* is about 11.2 km. The subject drives his car for about 1.2 km from *home* to an intermediate location. Then, he parks the car and depending on weather, he either walks or bikes to the subway station 4 km away. From the station, he rides on the subway to his office. According to the speed and circumstantial evidence from GPS reception, we can divide his route into three blocks of different speed zones. If the average of the user's current transit speed and the average speed within one block are within 30 percent, TTI will be applied to estimate the time of arrival. If the user's current speed does not match the average in the current speed zone, the average current speed will be used to calculate the time until the end of the current zone, and the remaining blocks will estimate the time of arrival using TTI. For example, when our subject decides to walk instead of taking his



bike, the speed of the walk will be applied to calculate the current block's ETA (Speed zone 2), which ranges from where the subject's car is parked to where he takes the subway. If this segmenting mechanism algorithm is not applied, all the remaining distance is applied to walking speed, and more than 30 minutes are added to the actual estimated time to the office; it takes about 25 minutes to travel from the station to the office by subway, and more than an hour by walking.



**Figure 4.12.** Dividing a route into speed zone blocks.

#### 4.3.4 Abnormal Transit Detection

Based on the algorithm for destination prediction and estimation of time to arrival, we can use the system to detect abnormalities such as i) deviation from average transition time and ii) unusual destinations. As described in the previous section, the route prediction algorithm simply returns the elapsed time from the departed location when it fails to predict a destination. In other words, if the user's transit pattern does not match the previously collected route templates, it fails to predict the destination and provides facts about the user's current motion. This feature is used to detect whether the user is still following the template or deviates from the template to infer abnormalities in transit on the route.

However, if the system detects every route's transit abnormalities and alert the user and their caregivers, it will begin to annoy them very quickly. Therefore limiting and restricting the detection area is required.

In every route template, the user can specify whether the route is used for detection. When user's traveling route matches the route template, the algorithm starts to

check if user is moving along the route at the expected pace and direction. Although the tolerance delay of TTI (transit time intervals) is set to three times of normal transition time interval by default, but users are allowed to change this through the *Contella* user interface.

*Contella* provides three parameters for adjusting the detection sensitivity: Minimum tolerable delay time (0 – 15 minutes), Maximum tolerable distance from route (50 – 150 meters), and Minimum tolerable speed proportion (0 – 100 %).

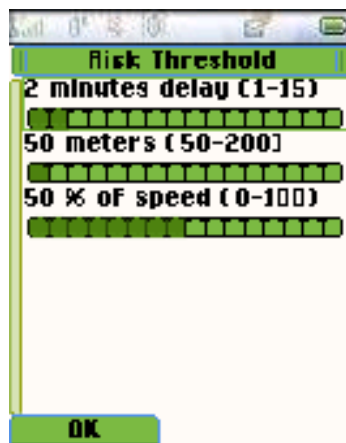


Figure 4.13 Screenshot of Risk Threshold

When the user specifies a route to be detected, the detection starts when it confirms the user is on that route. For instance, we can consider two different routes with two different destinations that have an overlapping section (see <Figure 4.13>).

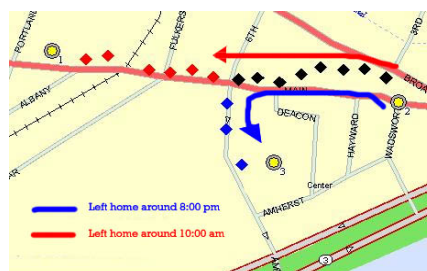


Figure 4.14 Overlapping routes

When the user walks along the overlapping section, the *Contella* prediction system starts to calculate a probability value using the route transit count table and the

current time of day. Thus, although the user is on a route where the algorithm should initiate a positive detection, the detection will only be activated when the most probable of the route and its destination matches the route to be detected. Because the algorithm for detection is restricted to detect daily routines such as going to school, the office, and home, this feature disambiguates the other routes from target routes to be detected.

## 4.4 System Architecture

The architecture for “Where are you going” is designed to accommodate the structuring of collected route data and the detection of route transition. In addition, there is a communication component that supports the exchange of context information and the remote access and manipulation of the slave phone’s location-related properties.

### 4.4.1 Data Structures

The system is based on five databases, each respectively containing;

- Contact Information
- User Profiles
- Route intermediate points (*iPoints*)
- Route Data
- Cell-Tower Data

The user profile and contact information databases have similar data structures. Both contain basic information about a person’s name, phone number, the Bluetooth ID of their phone device, its IP address and some other ‘trust’ information. The phone owner’s profile is stored in a single Java object called *User Profile*. *Contella* organizes contact information in the contact database by way of a Java hash table. Both the profile information and the contact information are used to make calls, or to send data over IP (depending on the trust settings).

There are two classes of location information in the system: *IntermediateData* (*iPoint*) and *LocationData*. *IntermediateData* is a parent class of *LocationData*, and an *iPoint* class stores information about its ID, GPS coordinates, GPS reception accuracy, *RouteData* IDs, and *CellTowerData* IDs. The IDs of *iPoints* and other data objects in the system are automatically obtained from the system based on the time of data creation.

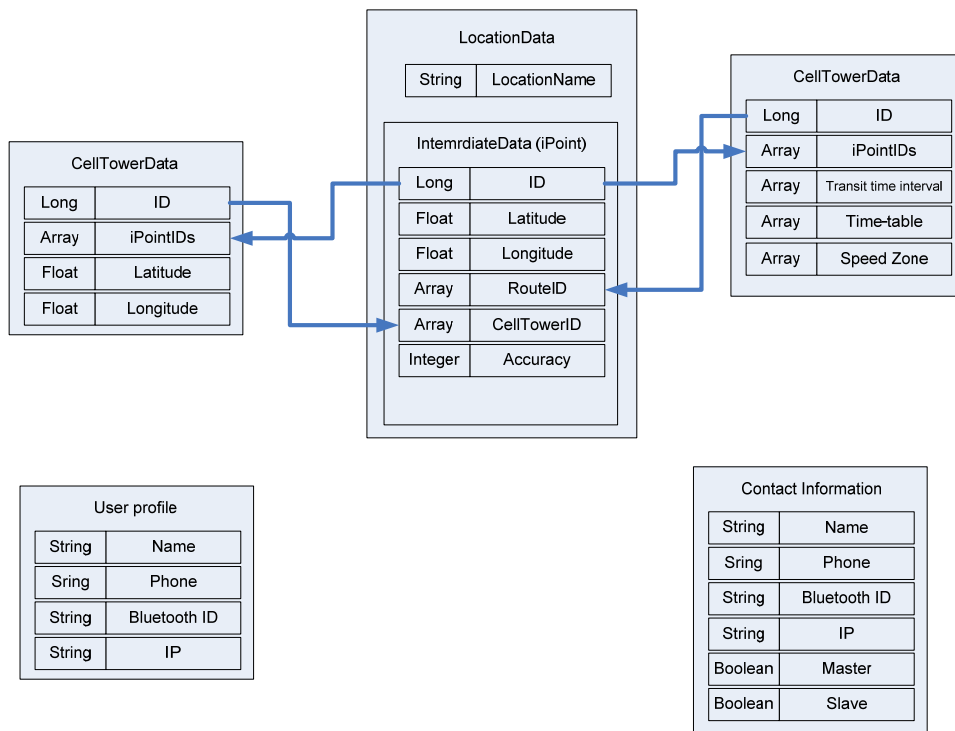


Figure 4.15 Data structure of Location Manager and contact list

Each *iPoint* stores all the information about a specific location. The *iPoint* contains information of latitude/longitude, associated routes, and cell-tower IDs. Multiple *iPoints* are stored in the *iPoint* database. The database handles searching and storing *iPoints*. *LocationData* inherits the data structure of *iPoint* but it also additionally stores the name of the *iPoint*. *LocationData* stores the user's bookmarked locations, and multiple locations are stored in the Location Database in the form of a Java Vector. *LocationData* also contains its associated *CellTowerData* IDs and *RouteData* IDs.

When a new cell tower is detected, a *CellTowerData* object is created. A *CellTowerData* object contains an array of *iPoint* IDs whose location are associated with this cell tower. The GPS API provides the location of the currently assisting cell tower, and this information is used to index *iPoints* as described in the previous chapter. *CellTowerData* objects are stored in the *CellTower* Database.

*RouteData* is composed of a unique ID, an array of *iPoint* IDs, start and end points of *LocationData* IDs. In addition, it also contains a number of index arrays that contain information about transit-time intervals (TTI) between *iPoints*, a timetable index for recording route transit time of day, the distance between *iPoints* and speed zone information. The *iPoint* array, timetable and TTI index array are persistent information that are stored in memory but the distance and the speed zone arrays are dynamically generated from the persistent information arrays. The *iPoint* array in the *RouteData* object is used to search for adjacent *iPoints* as described in the previous section. The system stores multiple *RouteData* objects in the Route Database.

#### 4.4.2 System Component

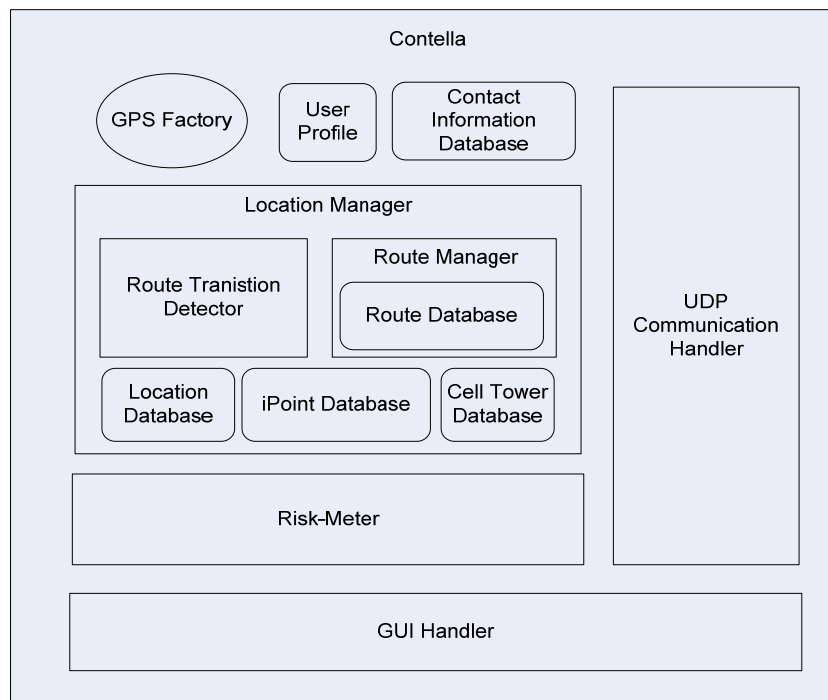


Figure 4.16 Detailed System Diagram of Location Manager in *Contella*

The *GPSFactory* is a component that listens for and handles the GPS events that come from the GPS receiver build into the Motorola i870 phone. The Motorola proprietary Java API in the iDEN SDK provides the functionality for handling GPS. *GPSFactory* checks for GPS events every five seconds and converts GPS information into an *IntermediateData* (*iPoint*) object as required.

The *Location Manger* is a core component of the system that handles incoming *iPoints*. The *Location Manager* handles *iPoints* based on three different cases; bookmarking, route training and route detecting.

In the case of bookmarking, the *Location Manager* converts the latest *iPoint*, together with the current location's name (as labeled by the system user) into *LocationData*. The *LocationData* is then stored in Location Database and its ID is registered in the associated *CellTowerData*. If the currently detected cell tower is not know to the system, the *Location Manager* creates a new *CellTowerData* object, then registers the location ID in the new *CellTowerData* object.

When the system collects route information, the *Location Manager* checks if the user has left from a labeled location. When the user leaves from the location, the system first stores the *LocationData* ID in the route as one of the end points in the *RouteData*, and then it starts to collect *iPoint* data. While collecting *iPoints*, the system also collects cell tower information and the interval transit time from the previous *iPoint*. When the user arrives at a labeled location that is different from the departed location, the *Location Manager* registers this location's *LocationData* ID as the other end point of *RouteData*. However, currently collected *iPoints* are stored not in the *iPoint* Database but in a temporary space until it finishes collecting routes. Because a route is collected only between two different locations, the system postpones permanently storing the information until it reaches a different location. Upon arrival at the different location, *iPoints* are stored in *iPoint* Database. Then, the preserved order of collected *iPoint* IDs are registered in *RouteData* with associated information such as the ITT and the timetable.

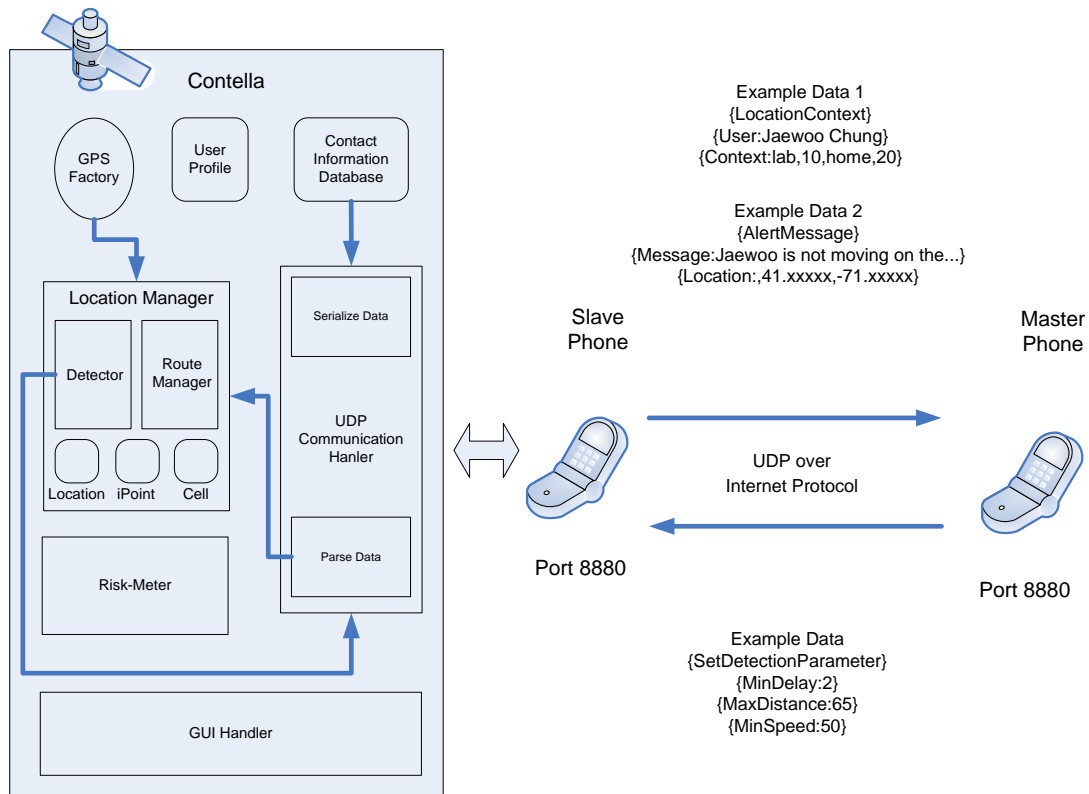


Figure 4.17 Slave phone's perspective of processing alert and context messages

While moving, the Route Transition Detector is activated. The Route Transition Detector handles the detection of nearby *iPoints* and the user direction of motion. The *Location Manager* passes an *iPoint* to the Route Transition Detector which then searches for any nearby *iPoints* in the *iPoint* Database. As we described in detail in the previous section, the Route Transition Detector searches *iPoints* based on the previously detected *iPoint's RouteData* IDs, and if they do not exist, it searches for *iPoints* associated with the current cell tower information. When it detects an *iPoint*, it determines the relationship with the previously found *iPoint* and computes the destination and ETA. Subsequently, it generates a context message and sends it to all the *Contella* phones whose information is stored in the Contact DB.

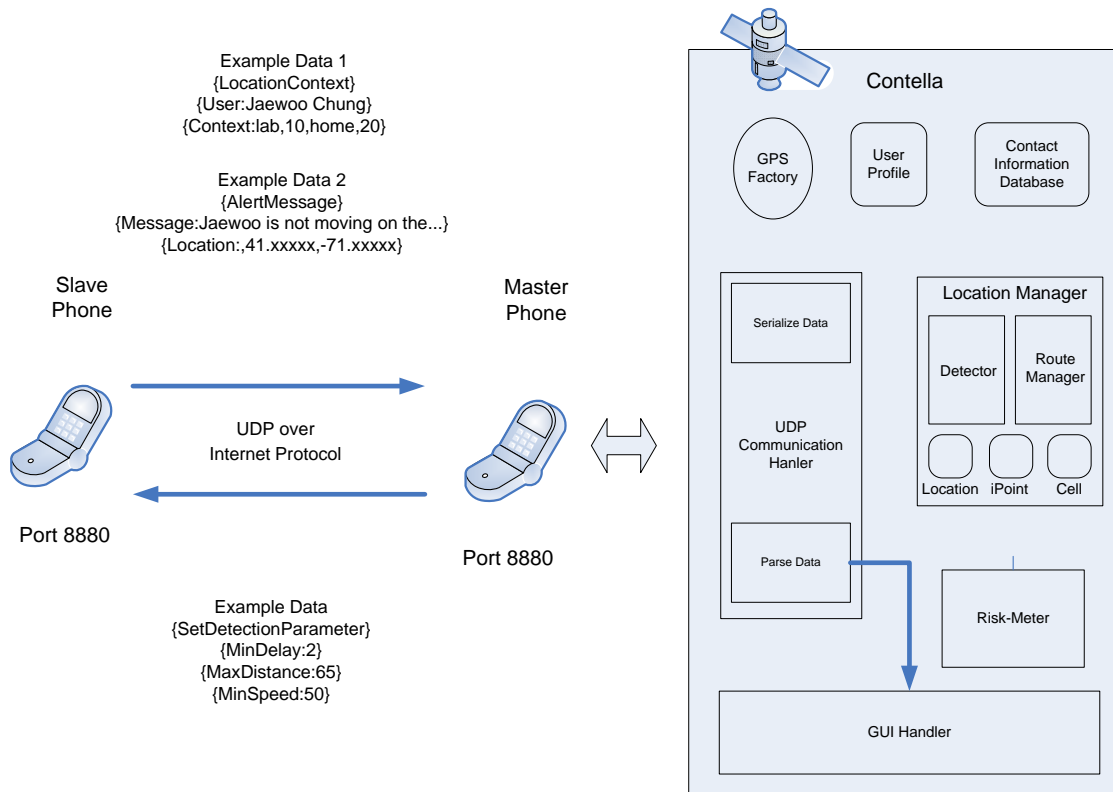


Figure 4.18 Master phone's perspective of processing an alert message

If, the Route Transition Detector can't predict the destination and ETA time, it means that either the user deviated from the route or the user stopped moving between *iPoints*. In that situation, the system creates an alert message and sends it to any contact who has been designated as a 'master'.



## Chapter 5

### Who is around you - Finding Contacts Around You.

This chapter explores how we can find someone who might be able to help our loved ones when they are in need. It also looks at how we can successfully communicate with these intermediate parties in a way that does not interfere with their privacy or burden them with social pressure.

The goal of this part of the system is to provide a mechanism to cultivate social activity by logging a user's communications, locations (via GPS) and nearby phones (via Bluetooth) scanning. In addition, it provides a means to communicate with a third party who is physically close to our distant family members, while minimizing the exposure of the third party's privacy. The system also enables us to receive confirmation of whether or not someone is available to help our distant loved ones at their physical location. If the first person who is contacted is not available, we will then need to try to contact each person suggested by the system (in turn) else until we eventually find someone who is available and willing to help.

The following is a simple scenario that describes how the system could be used.

#### 5.1 Scenario

James has an elderly father who lives alone in Philadelphia. Like most people with concerns about their elderly parents, James worries about his father's health problems. However, James lives in Chicago, far away from his father, and the only way to check on his father's well being is through a phone call. James regularly calls his father to see if he is OK.

One morning, James calls his father's home phone but his father doesn't answer. Immediately, he dials his father's mobile phone, but his father still does not answer the phone. James checks the GPS information on *Contella* to see where his father is. Surprisingly, *Contella* tells James that his father is at home. James thinks it might be an emergency situation and he uses *Contella's* "Emergency Contact" button to deliver a message to someone who is living close by to his father, who can check in to make sure his father is all right. *Contella* brings up a textbox that James can use to compose a

message. *Contella* initially provides a predefined message for James, which says: “Hello, could you please check if my father Jeffrey is OK? My number is 617-253-xxxx. Thank you. James Lee.” James modifies the message, and presses ‘send’ in the hope that someone who has a close relationship with his father will receive the message.

First, the message is sent to his father’s retired co-worker, Phillip, who has visited James’ father’s house before. *Contella* has found someone who has a recent history of visiting James’ father. However, Phillip is very busy at the moment and isn’t able to act on the request. He answers “No” by pressing the ‘no’ button. Immediately James receives a message that the request was denied, but he doesn’t know who denied the request as this information is kept anonymous. He presses the retry button to send out the same message that he composed to another contact.

The second message is sent to his father’s new friend Fred, whom he met a few months ago. Fred has spent time with James’ father and has an idea of what James’ father might be doing at the moment. He is willing to help his friend’s son out and he replies to the emergency message by pressing “Yes” to the message. Immediately James gets a “Yes” message, and he knows that someone will help him out. Soon, James receives a phone call from Fred. Fred tells James that his father went out fishing early this morning with some of his other new friends, and he probably forgot to take his mobile phone with him. Thus *Contella* showed that he was still at home.

## **5.2 User Interface**

The user interface described in this chapter mainly focuses on asynchronous text messaging between caregivers and the social peers of their care-recipients through *Contella*. The user of the master phone initiates the messaging by selecting “Emergency Contact” in the contact’s detailed screen as shown in Figure 5.1(b).



Figure 5.1(a) Detailed contact information, (b) Emergency Contact button, (c) Automatically composed message

When a caregiver presses the “Emergency Contact” button (*as shown in <Figure 5.1(b)>*), a predefined message is automatically created using the caregiver’s identification information, as well as that of the currently selected contact, as shown in Figure 5.1(c). Users are allowed to edit the message by using the keypad on their mobile phones. In this screen, the user may press the “OK” button to send the message to the care-recipient’s peers. Alternatively, the user may press “Cancel” to abort this task. Some trust must be established between the phones of the caregiver and the care-recipient in order to access this functionality. As mentioned in previous chapters, in order for the “Emergency Contact” option to be used (as a master phone application), the caregiver must be set as ‘master’ in the care-recipient’s slave phone, and the care-recipient must be set as ‘slave’ in the caregiver’s master phone.

In this service, there is no direct interaction between the caregivers and care-recipients. The service is not trying to interact directly with the contact but rather with the care-recipient’s peer. This creates an indirect way, or even an alternative way, to communicate with the target contact when his or her communication channel is temporally unavailable.

The care-recipient’s peers who have installed *Contella* on their phones will receive a help request message. When the message arrives, it appears on the recipient’s phone along with an alert sound, as shown in Figure 5.2. Recipients have an option either to accept the request by pressing “OK” or to deny it by pressing “NO”.



Figure 5.2 Received help request message

In response, the caregiver will receive one of two messages, depending on how the care-recipient's peer chose to respond to the request. When the response from a peer is positive, the caregiver will receive a message saying that the request has been accepted, accompanied by a vibration and an alert sound from the phone, as shown in Figure 5.3(a). However, if the peer's response is negative, the caregiver receives a message saying, "Your request for help is denied", which is also coupled with a vibration and an alert sound. However, when a caregiver receives this denial message, he/she also has an option to retry the search for a new peer to contact.



(a) Screenshot of acceptance message , (b) Screenshot of denial message

Figure 5.3 Response messages of emergency contact message

### 5.3 Design consideration of the system.

#### 5.3.1 Data logging and implicit information exchanges between phones.

This section describes issues concerning the collection of information about who is physically proximate to users of the system. First, we searched for a way to collect information about mobile phones that people carry all the time. Nathan Eagle, in his Reality Mining project [16], used Bluetooth to detect other personal mobile devices close by that were also equipped with Bluetooth. Eagle used this information to derive social networks among groups of people. Short-range Bluetooth scans distances up to approximately 10 meters and is able to identify other nearby Bluetooth devices. It can also access the Bluetooth IDs of the devices it finds. Although this method is very useful for logging information about nearby social peers, Bluetooth scanning only provides information about proximate Bluetooth IDs. Given only this information, we cannot necessarily identify the owner of each device. Thus, we are still left with the problem of collecting additional information about each device in order to identify its owner.

To solve this problem, we used a similar approach, but took things one step further. In our method, we not only scan for proximate Bluetooth devices, but we also identify what kinds of services are available in each detected device in order to find phones with *Contella* installed. Bluetooth uses UUID to identify services and applications installed on phones. UUID is a Java class that represents an immutable universally unique identifier that is composed of a 128-bit value. The Bluetooth API in JSR-82 allows us to assign a service or an application with a UUID. This assigned UUID identifies existing services running on mobile phones.

*Contella* employs two phase-detection, device level and service level, for scanning Bluetooth devices. After *Contella* detects proximate devices, it executes an additional search for the UUID to see if *Contella* is running on each detected device. When a matching UUID is found, *Contella* attempts to exchange contact information with the other phones also running *Contella*. Contact information for a phone includes its IP address and phone number. This implicitly exchanged information is not accessible to

the owner of a *Contella* phone but it is used to send emergency contact messages when necessary.

In addition to Bluetooth IDs, location information is also collected by *Contella*. As described in the scenario in Chapters 5 and 6, location awareness helps a caregiver to assess the safety of his or her remote dependents. Consequently, the system not only keeps a log of proximate phones, but it also stores the location at which two *Contella* phones come within range of each other. In a situation in which we know where our loved ones are, such as at home, we may also want to find out who else is nearby. Additionally, we may want to find out if anyone else has recently visited this location. We can look at the saved time information in the system to infer who is likely to be near our loved ones at different times during the day.

The collected logs are divided into two groups: logs of those who are already in *Contella*'s contact list, and logs of those whose information is unknown to *Contella*. The system keeps this information in two lists: a contact database for known people, and an internal contact database for unknown people. This internal contact list is hidden from users and the information on it is available to the phone's owner only when the owner explicitly adds someone from this list onto his contact list; then the internal information will be added into *Contella*'s contact list as well. In addition, this internal contact list is additionally used in finding contact process.

### 5.3.2 Search Process

Based on *Contella*'s phone book, the system indexes each contact's log information in the form of a table. This helps *Contella* have a more efficient search process. Bookmarked locations form one axis of the table and time windows (each an hour in length) constitute the other axis. Therefore each cell of the table will contain the number of instances of when the phone was in a particular location at a particular time. On the other hand, for each of the logs that were not collected at any of the bookmarked locations, only the total number of Bluetooth scan detections is considered. This number conveys how often a contact has been within range of a phone's owner.

In addition to this location-time table, when a master phone makes a request for an emergency contact, the system considers phone logs, as well as the current distance between a contact and the system's user, in order to find an appropriate contact. The process of searching for a contact is described as follows: First, *Contella* identifies the user's current location through GPS. If the user is at any bookmarked location, the system looks up this location in the location-time table for each contact to determine the likelihood of any of the user's contacts being in the same location at the same time. If the user's current location is not bookmarked, then this procedure is skipped.

Second, the system calculates which contacts have spent the most time with the user by looking at which contacts have the greatest number of Bluetooth detection scans. Third, the number of Bluetooth detection scans is added to sort the contacts in a list. Finally, *Contella* chooses ten people who have the highest sum, and sends out GPS coordinates to the contacts. After the distance from the user to each of the contacts is returned, *Contella* chooses the nearest person to send the emergency message to.

Although the algorithm to search for an emergency contact is carefully designed to maximize the probability of getting help, it is inconclusive whether this would actually work in a real-world situation. It is highly difficult to apply this algorithm to all kinds of different situations, without large-scale and long-term user studies with the system. However, the purpose of this algorithm is not to optimize the search process for finding a person for every situation, but rather to find *any* contact that would at least have "relevance" to the system user. The major emphasis of this system is to enable communication to confirm one's willingness to offer help in a potential emergency situation.

#### **5.4 Communication process between Master phone and Slave phone.**

In order for the system to be successful, three conditions should be satisfied: i) The system should be able to find people physically proximate to our dependents who are in need, ii) The system should hide from the requester the contact information of the peers of our dependents so that they will not feel any social pressure affecting their

decision to accept the request to help. iii) The system should establish communication with our dependent's peers in order to get confirmation of whether the peers will accept or deny the caregiver's request for help. We have already described an approach to solving part of the first constraint. In this section, we discuss design issues related to the second and third constraints.

There are two possibilities for communication with the contacts who are identified as possibly being able to provide help. The first is that when a master phone commands a slave phone to search for possible contacts, the slave phone would send all the search results back to the master phone. (To take this approach one step further, a master phone could even request a slave phone to return all of its logs so that a caregiver could identify a possible contact him or herself). One advantage of this approach is that caregivers would have access to the contact information of their dependents' peers, and could choose to call a peer directly to obtain confirmation about whether they could help. However, this violates one of our design constraints; the privacy of our dependents' peers is exposed without permission.

The second possibility is to allow the slave phone itself to send the caregiver's composed help request message directly to the dependents' peers. This approach protects the privacy of the dependents and their peers, but still successfully forwards the help request message to the dependents' peers. In addition, this method provides a negotiation process in the message by giving the message recipient the option to accept or deny a request from caregivers. This negotiation is necessary so that caregivers can find other contacts when they get a denial of their request.

In this second case, the message is not sent directly from the caregiver. Additionally, the information about the message's recipient is made anonymous to the caregiver by allowing the slave phone to choose who the recipient will be. This results in the response to the request being a voluntary effort, such that the message's recipient can be impartial in his/her response to the request. This creates an environment in which people can make requests without putting social pressure on the recipients, and thereby, not affects their decision on how to respond to the request. Also, the requester is not a total stranger but rather, a peer's caregiver, who is requesting help for someone that the



recipient presumably knows and cares about. As mentioned earlier, we intended to design a service that would create a communication channel between two people to allow mutual voluntarily help.

## **5.5 System Architecture**

The system “Who is around you” in *Contella* consists of data structures for collected logs, a process for data mining, and Bluetooth communication. The log manager is the main component of the system; it handles log collection and data analysis to enable searching contacts from collected logs. The log manager consists of two main data structures: *Bluetooth Log Data* and *Call Log Data*.

### **5.5.1 Data Structure**

*Call Log Data* is a simple data collection that records every incident of a phone call in a mobile phone. Motorola provides a J2ME SDK API for accessing call logs stored in Motorola iDEN phones. In addition, the API provides a listener that listens to updates in recent call data. This call event listener allows us to keep track of call events in real-time and to store our own log data beyond the allowed limit of phone call logs. *Call Log Data* contains a phone number, time and duration of each phone call, and a Boolean variable for identifying calls as either incoming or outgoing. Each record of *Call Log Data* is stored in the Call Log Database.

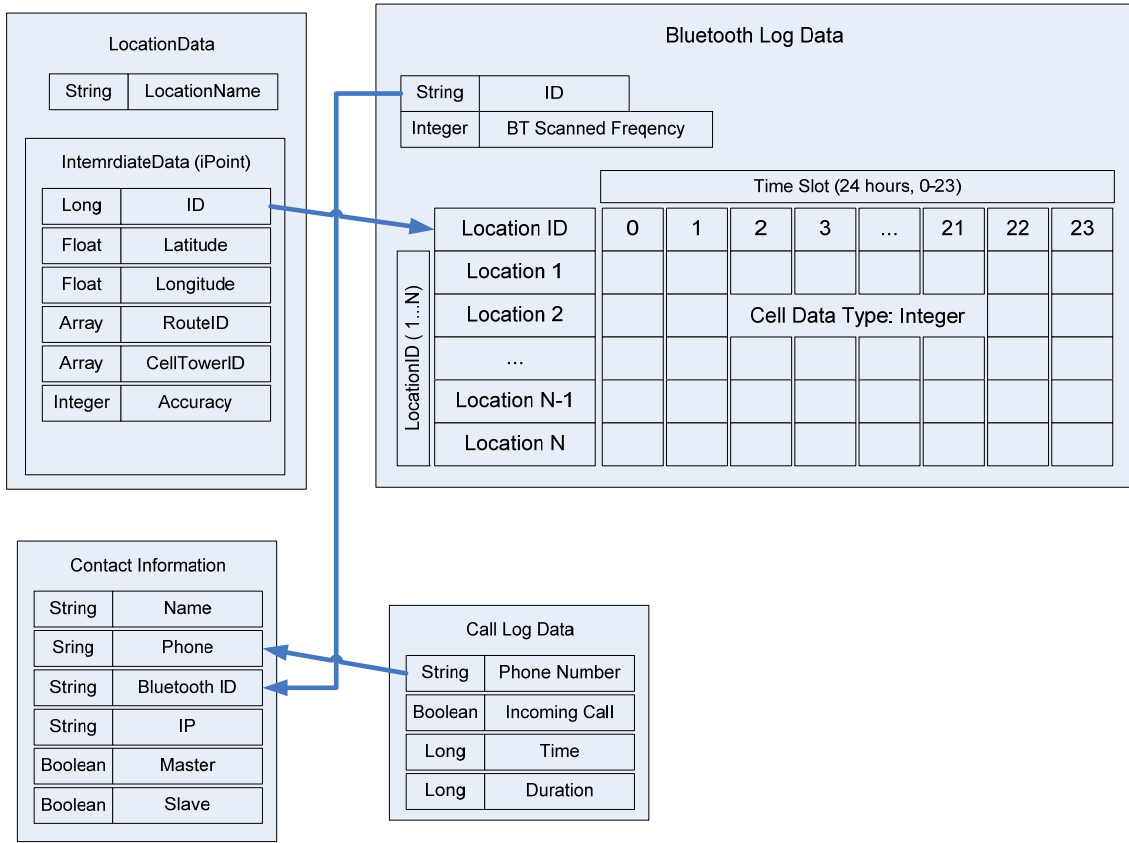


Figure 5.4 Data Structure of Log Manager

*Bluetooth Log Data* consists of a table object, ID, and Bluetooth Scanned Frequency. For each contact in *Contella's* phone book, a *Bluetooth Log Data* entry is created. When a *Bluetooth Log Data* entry is created, its ID corresponds to the contact's Bluetooth ID. Whenever the system performs a Bluetooth scan and finds a match with any contact's Bluetooth ID, the Log Manager increments the Bluetooth Scanned Frequency.

A table object in the *Bluetooth Log Data* is a two-dimensional table that consists of a location axis and a time axis. The location axis is comprised of *Contella's* labeled places that are stored in the *LocationData* object. When a table is created, the Log Manager refers to all the Location IDs stored in the Location Database and creates a table row for each location. Whenever there is a change in the Location Database, such as adding or deleting a location, the table is updated.

The table 24 fixed columns, each corresponding to one-hour intervals of time in a day. Whenever a detected Bluetooth ID matches with a contact’s Bluetooth ID while both phones are in a labeled place, it increments a number in the cell that matches both the current location ID and time of day. This table is used to track the Bluetooth detection frequencies of people who are co-located with *Contella*’s owner in any labeled place.

### 5.5.2 Log Handler

The *Log Handler* consists of two process components: the Log Scanner and the Log Collector. The Log Collector organizes logs created from events such as recent calls, GPS positioning, and Bluetooth scanning. The Log Scanner searches for contacts using the Call log Database and the Bluetooth log Database.

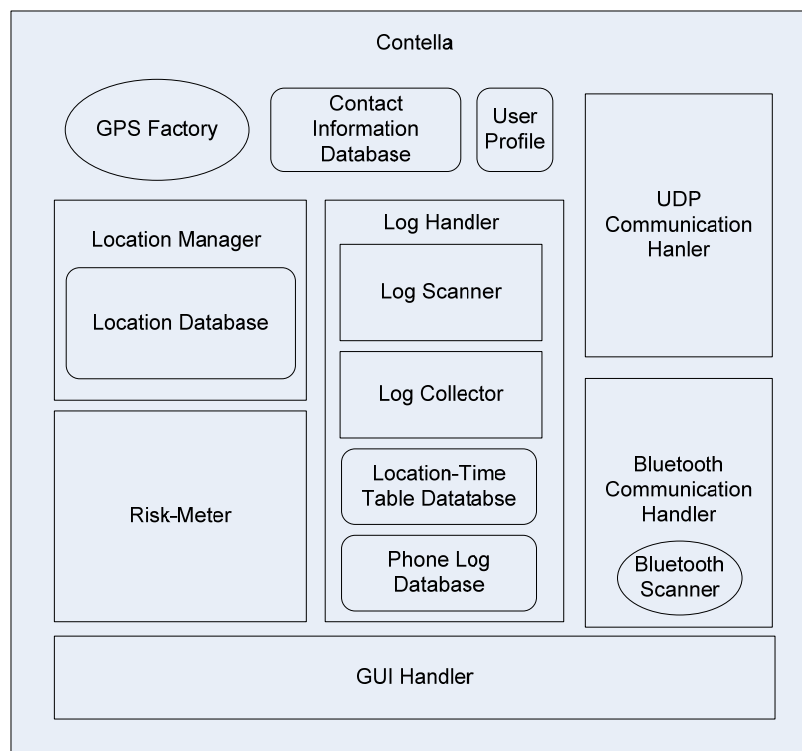


Figure 5.5 System components for service “Who you are around”

The Log collector requests the *Bluetooth Communication Handler* to scan for proximate devices every five minutes. Whenever the Bluetooth Scanner returns scanned IDs to the *Log Handler*, the Log Collector then searches for any matching IDs between those it has just detected and those saved in the contact list. If it finds any matches, it updates the corresponding Bluetooth Log Data. In addition, the Log Collector tracks every incoming and outgoing call event and records this information in the Call Log Data.

Whenever *Contella* requests emergency contacts, which is initiated by care-givers through the master phones, the Log Scanner first checks the most recently detected GPS coordinates from the Location Manager to see if the coordinates are within a range of 100 meters from any bookmarked place. Depending on what it finds, it accesses the Bluetooth Log Database and the Call Log Database to gather information for conducting a search. After finding candidates, the Log Scanner accesses the candidates' IP addresses from the Contact Database, and sends out GPS coordinates to the candidates' *Contella* phones through the UDP Communication Handler

When *Contella* receives this request from a fellow *Contella* phone, the message is forwarded to the *Log Handler*. The *Log Handler* checks if the requester is in the contact list. If it finds the requester in the database, it computes the distance to the other *Contella* phone's location and sends the distance information back to the requester. If it does not find the requester in Contact Database, it discards the request.

### 5.5.3 Bluetooth Communication Handler

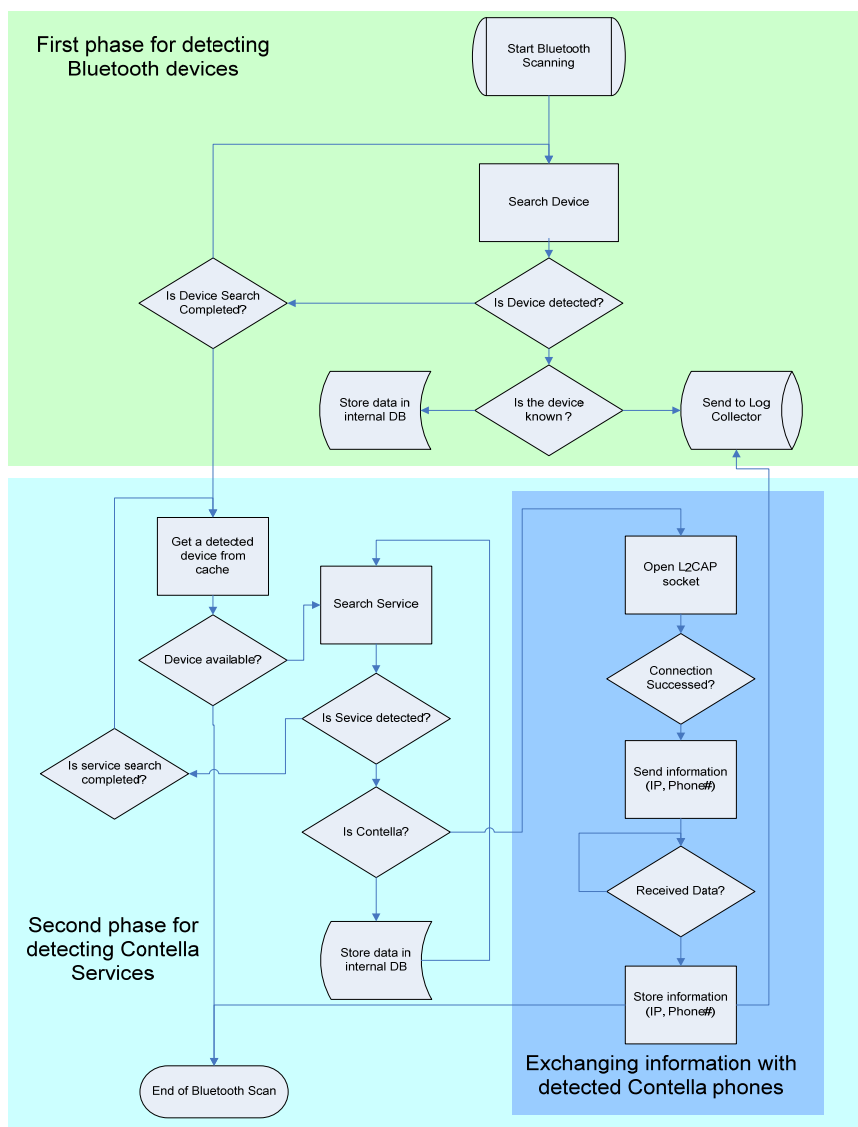
The *Bluetooth Communication Handler* handles the scanning of proximate Bluetooth devices as well as Bluetooth services which reside on the detected devices. In addition, using Bluetooth's L2CAP<sup>11</sup> protocol, we were able to establish a communication connection between other Bluetooth devices without pairing them. This allows *Contella* phones to spontaneously exchange information without any involvement of the devices'

---

<sup>11</sup> L2CAP, Logical Link Control and Adaptation Protocol is used within the Bluetooth protocol stack. L2CAP is layered over the Baseband Protocol and resides in the data link layer. L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions.

users. We used a fixed 512 byte size packet in L2CAP for sending and receiving encoded text messages.

The following diagram depicts the control flow of scanning and exchanging messages between Bluetooth devices using the L2CAP protocol. When the Log Collector requests for the *Bluetooth Communication Handler* to scan for proximate devices, the handler conducts a search in two phases. First, it begins to search for proximate devices and determines whether each device is known to the system. If the device is known, it sends the scanned log information to the Log Collector. If the device is not known, it stores the detected information in a temporary cache.



**Figure 5.6** Bluetooth Handler’s communication and Bluetooth scan flow

When the device search is finished, the *Bluetooth Communication Handler* begins the second phase, in which it searches for *Contella* services on the detected devices stored in the temporary cache. The procedure of searching for these services is shown in the blue block of Figure 5.6. For each detected device in the cache, the *Bluetooth Communication Handler* searches for a *Contella* UUID, which is assigned in every *Contella* phone. When it finds a UUID, it means that another *Contella* phone was found, and it tries to open a socket via L2CAP to exchange data with this phone. After the completion of the data exchange, the received data is sent to the Log Collector.

## Chapter 6

### Evaluation

This chapter describes some evaluation on the three major components of the system: “Where are you”, “Where are you going”, “Who is around you”.

#### 6.1 Where are you – Risk-Meter

To properly evaluate the system Risk-Meter, it requires the measuring of users’ perception of risk and a comparison with the Risk-Meter’s risk measurements. Such an experiment would require a large number of subjects in different locations for long periods of time. Also, in order to get truly valuable feedback from the users, it is desirable to consider subjects with a lot of experience and knowledge about crime incidents and tendencies in their neighborhood. Instead, we decided to interview a few people who have good knowledge about security and safety around the MIT campus area.

For the evaluation of the Risk-Meter, we interviewed three people who work in the area of safety and security (either professionally or voluntarily). For each subject, we first explained the purpose of the risk-meter and explained how the system works. This took about 10 minutes. Then, we let them use the application around the MIT campus for a few hours. After the use of the application, we interviewed each subject to get their feedback and comments.

The profiles of the interviewees were as follows:

- Subject 1 – The organizer and student member of the MIT Crime club (male).
- Subject 2 –President of Student-Alumni Committee on Institutional Security Policy (male).
- Subject 3 – An MIT Police sergeant from the crime prevention department (female).

Generally, the purpose of the Risk-Meter was well understood by all three subjects, and they were enthusiastic to experience the application. *Subject 1* expressed his

initial response to the application; “it is very exciting to see such an application.” One of the subjects explained that the first step of crime prevention is awareness of crimes, and he thought that this “handy device that shows statistics of crime is very useful for students in the campus”. *Subject 1 and 2* pointed out that the broad use of the application will expedite the process of modernizing the police report system by coupling GPS information as the location where the crime occurred. Also, they all liked the red bar graphs which allowed them to see the result by just glancing at the screen for a brief moment. They also thought this bar graph was very intuitive. When we asked what the bar graph might mean, they were each able to correctly explain what it depicted.

After they walked around the campus, they expressed their understanding that the Risk Meter was showing information relevant to their proximate surroundings. For example, when *Subject 1* reached student center, a place where a lots of bikes and laptops are stolen, the bar graph jumped up a bit. They liked the fact that the phone knew where they were so that they did not need to input their location into the phone. Around 6 pm, the measurement for “car’s” risk reached at its peak, and slowly decreased over a 2 hour period. *Subject 2* closely observed this, and appreciated the fact that the measurement of the risk reflects the time of day. He explained that he feels this information to be more vivid than what we can get from crime-log applications on desktop computers. He said he will remember this fact much better than what he learned on his desktop computer.

They also pointed out that there are flaws in the system. For example, when *Subject 3* was around the student center at 9:30 pm on Wednesday, the Risk Meter showed low risk on the item “laptop”. She said “this is not true.” She pointed out that although the total number of stolen laptops is small, the human traffic in the student center is quite low at that hour. The risk has to consider the fact that sparsely populated areas can be less safe. *Subject 2* pointed out that although risk is low, this does not mean there will be no crime, so it is important to let users know this does not mean people and their property are “Safe.” All the subjects were concerned about the possibility that users may interpret this information to measure safety instead of risk. They each asked for an explanation to be inserted into the system to avoid users having a sense of false safety.

Another potential problem of Risk-Meter is that it measures risk based on police-logs. However, if there is some amount of crime log data missing in from a certain



location, the subjects pointed out that they felt people would lose confidence in the Risk-Meter and would stop using it. In fact, *Subject 1* discovered a hole while he walked along Vassar Street, around the football field. The Risk-Meter showed no information around this area. This problem is due to the fact that the police log information is based on building addresses. When an address is converted into GPS coordinates, the crime incidents associated with any given building will cluster on a single point. Although police keep detailed crime incident information internally, the only available locations for us to use are the building addresses. However, when we presented the problem of crime-logs that only keep addresses of buildings to *Subject 3*, she understood the issue and promised future collaboration with us. She said she would try to provide more detailed crime-logs in the future.

Finally, they commented about the additional features that they wanted to see added to the Risk-Meter. One of subjects wants to implement a function that allows users to bookmark where their cars or bikes are parked. Then, the Risk-Meter will alert user when the risk of their property at the parked location reaches certain threshold. *Subject 2* wanted to see the most recent crimes for each type of item, for instance, “three robberies occurred here in the last seven days”. *Subject 3* wanted to build some sort of authorization into the system so that criminals cannot use this information against people.

Overall, although the Risk-Meter needs some improvements, the idea of using crime-logs in combination with location based information retrieval was well received by the subjects. All the subjects want to see the system used by students very soon. In fact, the system will be introduced to incoming students in the Fall 2006 semester.

## 6.2 Where are you going – Route Detection

In this section, we evaluated the core algorithm for route detection in the system. The evaluation experiment had five people participants, one female and four males. The purpose of the experiment was to test the route detection algorithm and measure its performance in terms of i) how well the system learns about routes, ii) how well the system detects and predicts users' route transit behaviors, and iii) how well the system detects abnormalities in the transition.

First, each participant set the *Contella* to be in training mode in order to collect route information, then they each traveled on his/her own route at his/her own pace. They each made several round trips on their assigned routes. As an observer, we followed their trace of travel and checked their behaviors. Before they were traveling on their routes, we asked them to intentionally stop, or reduce their speed of travel. Also we asked them to deviate from their routes to see if the system detected these aberrations. Their transportation methods varied from walking to bicycling to driving. The length of the routes also varied from 759m to 3.6 km, and the elapsed time of the travel on a one-way trip varied from five to 11 minutes. At the end of the experiment, the participants generated a total of six routes and traveled on these routes a total of 30 times. The condition and transportation method of each route is described as follows:

Table 6.1 Route information in the experiments

Route ID	Length	Duration	Transportation	Road condition
Route 1	1.0 km	11 minutes	Walking	One end of the route is surrounded by tall buildings, and this trend continues along the route for 70 meters. The rest of the route passes through a residential area, which has one- or two- story buildings and very few trees along the path. The width of the road is about 35 meters.
Route 2	759 m	5 minutes	Walking	The most of the path is surrounded with medium sized buildings (about 5 stories high) and dense trees along the side walks. The width of the road is about 30 meters
Route 3	769 m	7 minutes	Walking	No major obstacles are found along the path. The width of the road is about 50 meters.
Route 4	1.4 Km	5 minutes	Bicycling	Half of the route is covered with dense buildings. The diameter of the first rout is about 20 meters. The second half of the route has no major obstacles and the width of the rout is 40 meters.
Route 5	1.9 Km	6 minutes	Bicycling	No major obstacles are found around the path. One traffic light is found on the route. The width of the road is about 55 meters.
Route 6	3.6 Km	7 minutes	Driving	Buildings along the road are about 5 stories high, and, in average, 3 lanes for each side of the road. Seven traffic signal lights are found on the road.

Six different aspects of the measurements are considered to evaluate the algorithm. For the most part, the performance of the route detection/prediction and the detection of abnormalities which occur in route transit are measured. The following listings are six measurements that were taken from all the routes in this experiment.

- Distance between *iPoints* (intermediate points).
- Elapsed time between *iPoints*.
- Percentage of *iPoint* detections made on a route travel.
- Difference of each predicted time and actual time it took to travel on a route.
- The average time of detection of a delay on a route.
- The average time and distance to detect a deviation from a route.

#### **Distance between *iPoints*:**

The algorithm is set to collect an *iPoints* every 50 meters. However, the data from the experiment showed that there was an increase in the distance between *iPoints* as a user's travel speed increased.

Route ID	Length	Duration	Transportation	Average Distance
Route 1	1.0 km	11 minutes	Walking	64 m
Route 2	759 m	5 minutes	Walking	69 m
Route 3	769 m	7 minutes	Walking	69 m
Route 4	1.4 km	5 minutes	Bicycling	66 m
Route 5	1.9 km	6 minutes	Bicycling	86 m
Route 6	3.6 km	7 minutes	Driving	278 m

Table 6.2 Distance between *iPoints*

As shown in the Table 6.2, as the speed increased, the length of the distances between *iPoints* also increased as well. However, not only the speed of travel but also static obstacles, such as tall buildings and trees, prevent the system from collecting *iPoints*.

Although the average distance between *iPoints* was about 20% greater than 50 meters, if each distance between *iPoints* is considered, as shown in the Figure 6.1, most of the distances between *iPoints* were, in fact, close to 50 meters.

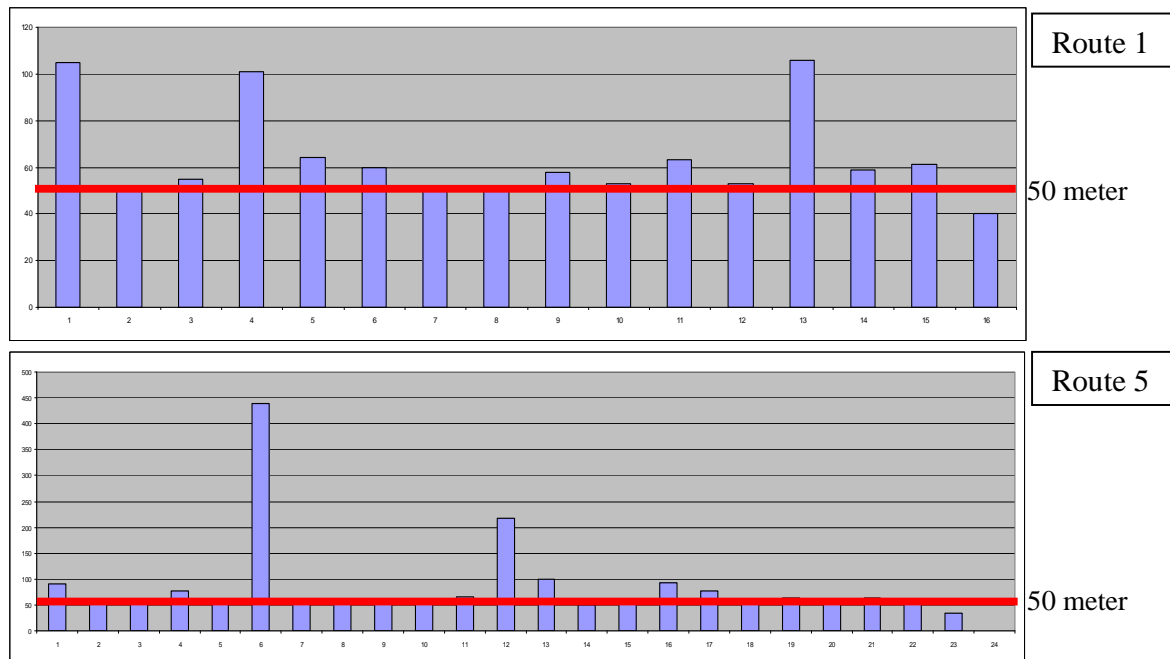


Figure 6.1 Distances between *iPoints* on Route 1 and Route 5. The red lines across the bars indicate 50 meter threshold.

The Figure 6.1 compares the distances between *iPoints* on Route 1 and Route 5. The blue bars in the figure indicate the distances between adjacent *iPoints* along the route. Route 1 has an average *iPoint* distance of 64 meters and Route 5 has an average distance of 86 meters. Although the average distance difference is more than 30 meters for Route 5 and 16 meters for Route 1, for 90% of the times, the system correctly collected both routes at every 50 meters. From this we can conclude that the most of the time, the system is correctly collecting *iPoints* at every 50 meters.

#### **Elapsed time between iPoints:**

The traveled-time interval (or TTI) between *iPoints* varies based on the speed of travel. Intuitively, as speed increased, the TTI decreased between *iPoints*. In order to accurately predict the estimated time of arrival at the destination, the system is required to detect one *iPoints* per minute. If the system fails to detect an *iPoint* within a minute, the algorithm stops predicting and provides facts about the departed location and elapsed time since the departure. Thus, ideally, the TTI should be less than one minute.

Route ID	Length	Duration	Transportation	TTI
Route 1	1.0 km	11 minutes	Walking	41 second
Route 2	759 m	5 minutes	Walking	29 second
Route 3	769 m	7 minutes	Walking	37 second
Route 4	1.4 Km	5 minutes	Bicycling	14 second
Route 5	1.9 Km	6 minutes	Bicycling	17 second
Route 6	3.6 Km	7 minutes	Driving	33 second

Table 6.3 Traveled-Time Interval between *iPoints*

As shown in Table 6.3, the data from the experiment showed that all the routes had less than one minute of TTI (mostly around 30 second) between the detection of *iPoints*. In fact, the distance between *iPoints* is designed to get at least two predictions per minute at normal walking speed. Figure 6.2 compares the TTI of Route 4 and Route 6.

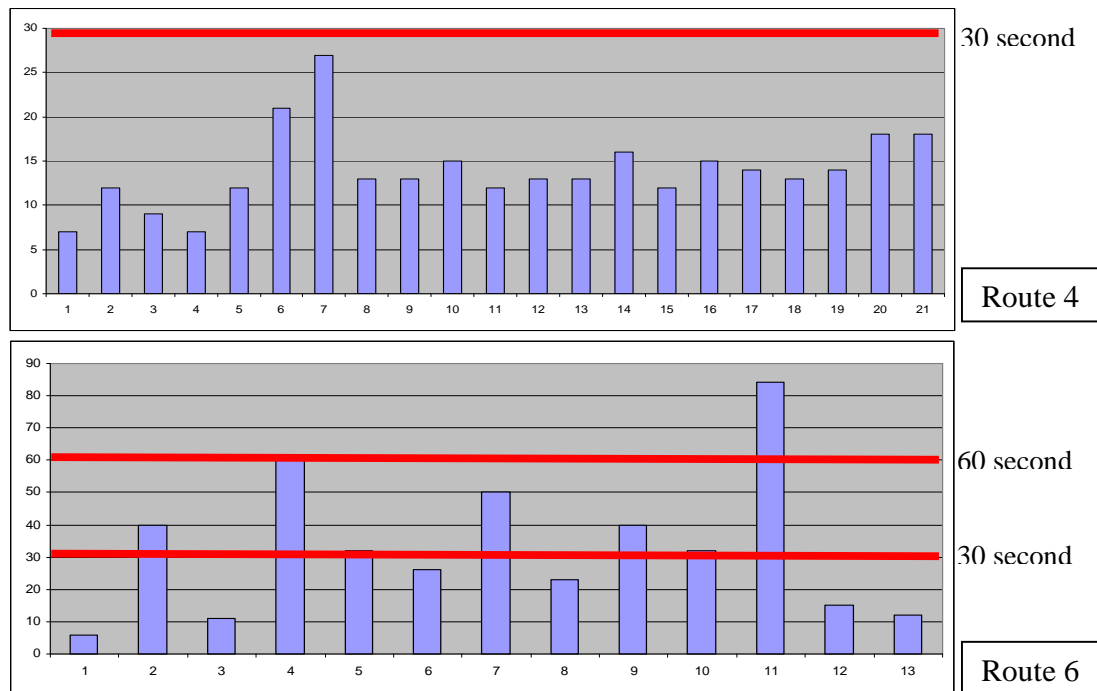


Figure 6.2 TTI between *iPoints* on Route 4 and Route 5.

As shown in the Figure 6.2, the TTI of the Route 4 is well below 30 seconds, and at the very least, the algorithm can make two predictions in a minute. On the other hand, the participant traveled on the Route 6 by driving. In the case of that route, the TTI at the 11<sup>th</sup> *iPoint* is over 60 seconds. Consequently, when the user travels between the this

*iPoint*, and the next one, the system will not be able to make a prediction fast enough before the user reaches the next *iPoints*.

Despite the fact that the average distance between *iPoints* is 278 meters in the Route 6, the system is expected to adequately predict the destination and estimate time more than 92 % of the time for the drive along the route because most of the TTIs were less than a minute. This result also confirms that the system effectively collects and generates the route templates for the prediction and detection of the route along which the users are traveling.

**Percentage of *iPoint* detections made while traveling on a route:**

After collecting route information and generating routes templates by the system, each participant made a return trip on the same route to see if the system successfully detected the route. The following Table 6.4, which shows the percentage of successful *iPoint* detection, is calculated based on all the trips made along the route where the *iPoints* were collected by the system.

Route ID	Length	Duration	Transportation	Number of <i>iPoints</i>	Detection	Road Diameter
Route 1	1.0 km	11 minutes	Walking	17	96%	35 m
Route 2	759 m	5 minutes	Walking	12	97%	30 m
Route 3	769 m	7 minutes	Walking	12	89%	45 m
Route 4	1.4 km	5 minutes	Bicycling	22	86%	40 m
Route 5	1.9 km	6 minutes	Bicycling	24	93%	45 m
Route 6	3.6 km	7 minutes	Driving	14	98%	40 m

Table 6.4 Percentages of *iPoint* detection on each route on the same route.

As shown in Table 6.4, the total *iPoint* detection average is over 93 %. The routes that have detection average below 90% are the Route 3 and Route 4. However if we consider the fact that the participants took the opposite side of street or sidewalk while they are traveling on the return trip at least once, the detection rate is higher than the result showed in the table. The width of the street on Route 3 and Route 4 are 40 and 45 meters respectively such that the other side of the street is out of the detection range (the system considers *iPoints* detected within 35 meters). Consequently, the system failed to detect any *iPoints* on Route 3 when the participant made a trip on the opposite side of the street. The major portion of detection failed on Route 3 was due to this problem and if the failure is ignored, the percentage of *iPoint* detection is increased to 97 %. Route 4 has a

similar, and if only the walking on one side of the street is considered, the detection percentage is also increased to be greater than 90%.

The Route 6 has the highest percentage of detecting *iPoints* in the Table 6.4. We presume that this is due to the fact that cars are driving relatively in the middle of the road, while pedestrians are walking along the sides of the road where trees and building walls may be an obstacle for getting GPS fixes from satellites. However, we do not have enough experiments that involve driving to confirm this hypothesis. All in all, the overall detection rate of the *iPoints* is over 93 %, and is adequately detected *iPoints* on both sides of the streets when the width is less than 40 meters.

**Difference between each predicted time and actual time it took:**

The difference between the actual time of arrival and the predicted time of arrival can be used to measure the performance of the route detection system. The prediction of arrival time is computed based on the route templates. We compared the predicted time of arrival with the actual arrival time on each routes.

Route ID	Length	Duration	Transportation	Number of <i>iPoints</i>	Average deviation in minutes
Route 1	1.0 km	11 minutes	Walking	17	0.53 minute
Route 2	759 m	5 minutes	Walking	12	0.45 minute
Route 3	769 m	7 minutes	Walking	12	0.22 minute
Route 4	1.4 km	5 minutes	Bicycling	22	0.19 minute
Route 5	1.9 km	6 minutes	Bicycling	24	0.73 minute
Route 6	3.6 km	7 minutes	Driving	14	0.69 minute

Table 6.5 Average deviation between predicted time and actual time to arrival.

As seen in Table 6.5, the overall average deviation of the actual arrival time from the prediction time was less than a minute. Thus, the average prediction error is within a minute and the results show that the system has a very accurate prediction performance. The experiment showed errors at a maximum of 2 minutes off from the actual arrival time in a few trips, but this maximum error was only seen 6 times out of 91 prediction incidents.

### The average time of detecting a delay on a route:

While participants were traveling on routes, they were asked to stop or reduce their transit speed deliberately to see if the system could detect this behavior. By default, the system is set to detect a two-minute delay between one *iPoint* and another. If a user travels between two *iPoints* and does not reach the second *iPoint* within two minutes, the system begins to alert the user with sound and vibration. In addition, the system detects the average speed of the user; by default, the system tolerates 50% of a speed reduction. For example, if the user is traveling at less than the half of the normal speed which is based on the previously collected route template, (or in other words, the user did not reach to another *iPoints* within the twice time of the normal TTI,) the system alerts the user. Users are allowed to adjust these parameters on the application on their phones.

In order to increase the detection of delays but not to be affected by detections of route deviation, the tolerable route deviation parameter it increased to 500 meters. This ensures that system will only detect any delays between *iPoints*.

Route ID	Length	Duration	Transportation	Number of attempt	Success of detection	Average time of detection
Route 1	1.0 km	11 minutes	Walking	3	3	2.3 min
Route 2	759 m	5 minutes	Walking	4	4	2.1 min
Route 3	769 m	7 minutes	Walking	3	3	2.4 min
Route 4	1.4 km	5 minutes	Bicycling	2	2	2.3 min
Route 5	1.9 km	6 minutes	Bicycling	2	2	2.1 min
Route 6	3.6 km	7 minutes	Driving	1	1	2.1 min

Table 6.6 The average time of detecting a delay on a route

As shown in Table 6.6 the system successfully detected any delays in the transit along the routes. All delays were detected by the system and the average time of detection was within 2.5 minutes. However, although all the delays were detected, a couple of detections did not match with a user's transit behavior. For instance, one participant stopped in the middle of traveling on the Route 3. The expectation that the system would detect this delay is within 2 minutes. However, the system did not detect this behavior, and instead, the system changed the participant's predicted direction back to the location they departed from. This problem is occurred because of GPS precision errors. The system collects GPS coordinates every 50 meters and the *iPoint* detection area



within a 35 meters radius. When the GPS precision accuracy is poor while initially collecting the route data, and the current GPS accuracy is also poor, there is a chance that the system will see an *iPoint* that the user already passed. This will result in changing of the destination prediction, and consequently the system will think that the user is still moving even when he has stopped. Although this error was found only once in this experiment, this error can be prevented in the future by either increasing the interval (currently every 50 meters) at which GPS route data is collected, or decreasing the *iPoint* detection radius to be smaller than 35 meters. In our observed instance of this, the system later detected the fact that the participant had delayed moving.

**The average time and distance of detecting a deviation from a route:**

In order to conduct the experiment to detect only the deviation only from the known routes, the delay parameters were set to allow maximum tolerance in the algorithm (tolerance delay: 10 min, tolerance speed: 10%). The tolerance deviation distance was also restricted to 100 meters. If a user deviates more than 100 meters from the last detected *iPoint*, the system makes an alert. The participants were asked to deviate from their routes during the experiment. After the system detected their deviations and made an alert, the participants returned to the route where they had left it to see if the alert was terminated.

Route ID	Length	Duration	Transportation	Number of attempt	Success of detection	Average time of detection
Route 1	1.0 km	11 minutes	Walking	2	2	1.5 min
Route 2	759 m	5 minutes	Walking	2	2	1.1 min
Route 3	769 m	7 minutes	Walking	2	2	58 sec
Route 4	1.4 km	5 minutes	Bicycling	4	4	32 sec
Route 5	1.9 km	6 minutes	Bicycling	3	3	28 sec.
Route 6	3.6 km	7 minutes	Driving	2	2	35 sec.

Table 6.7 The average time of detecting a deviation from the known routes

As expected, all the deviations from known routes were successfully detected. As presented in the previous result such that the system robustly of detected *iPoints*, the detection algorithm also showed the satisfying performance in this experiment. The time it took to detect deviations varied because of the travel speed of the participants. The

average of detection time was about one minute when the participants were walking, while the average detection time for moving by car or bicycle was about 30 seconds.

One potential problem was found, which could be easily fixed by modifying the detection algorithm. If the distance between the currently detected *iPoint* and the next adjacent one exceed a certain distance, about 250 meters for example, the system will not make an alert until the user deviates from the most recently detected *iPoint* for more than 250 meters. In order to fix this problem, the system needs to calculate the both distances from the most recently detected *iPoint* and next anticipated adjacent *iPoint*. If the user is on a know path and progresses toward the destination, the distance from the previously detected *iPoint* gradually gets larger as the distance from the next expected *iPoint* decreases. Therefore, checking the both distances enhances the detection of whether the user is actually on the track or off the track.

All in all, the performance of the route detection algorithm in the experiment showed that the system collects route information effectively, and also generates route template that detects routes with more than 92 % precision. In addition, its predictions of estimated arrival times are made within an error range of 0.6 minutes on average. The system detected abnormal transit patterns with 100% accuracy with a few minor errors.

### 6.3 Who is around you – Finding contacts.

In order to evaluate the system for its ability to find contacts properly, it is desirable to let users experience using the system for long periods of time in their daily lives to see how they actually interact with other people and how the system gathers information to make inferences about their social relationships. However, given our limited resources and time, we decided to conduct a small scripted experiment that would help to evaluate the system to see if i) the sensors adequately reflected the actual activities of people, and if ii) the algorithm for finding contacts worked in the way that we predicted. We created a scripted scenario involving three different personas: an office mate, a co-worker, and a close friend who also works at the same office. The experiment was designed to evaluate the core algorithm for finding social peers. Each participant in the experiment was assigned to one of the three personas we created. The personas are described as follows:

- *Contella user*: the key user in the experiment. The experiment is conducted from the point of view of this *Contella user*. The user's phone will try to find social peers based on the activities between this user and his peers detected through Bluetooth, phone, and GPS logs.
- *Peer 1*: a friend and officemate of the *Contella user* who has a social relationship with him both in and out of the office. They often call each other on the phone.
- *Peer 2*: an officemate who spends as much time in the office as Peer 1. However, the *Contella user* and Peer 2 do not have any social relationship other than their interaction in the office. They don't call each other on the phone.
- *Peer 3*: a work buddy who works in the same building as the *Contella user* and who exchanges phone calls with him for work-related purposes. At times, they also talk to each other in person.

Each user was provided with a phone installed with *Contella*, and told to carry the phone with them wherever they went. During the experiment, they also made efforts to co-locate at certain hours. In addition, they called each other on the phone in order to create social activities between the peers in accordance with their specified personas.

The following table describes the location and phone activities of each participant during the experiment:

Time	Location	Co-located peers	Phone calls
9:30am	Home	<i>Contella</i> user, Peer 1	Peer 3 – twice
10:00am	Office	<i>Contella</i> user, Peer 1, Peer 2	
10:30pm	Office	<i>Contella</i> user, Peer 1, Peer 2	Peer 3 – once
11:00pm	Office	<i>Contella</i> user, Peer 2	Peer 1 – once
11:30pm	Office	<i>Contella</i> user, Peer 1, Peer 2	
12:00pm	Cafeteria	<i>Contella</i> user, Peer 3	Peer 1 – once

Table 6.8 Activities of each participant

To summarize the table, *Peer 1* and *Peer 2* were co-located for 2.5 hours, and *Peer 3* was co-located with *Contella user* for half an hour. While the participants were co-located, *Contella* logged the scanned BTIDs, GPS coordinates, phone calls, and associated timestamps into the system. In addition, the *Contella* user made phone calls to his peers, except for *Peer 2*. He exchanged a total of three calls with *Peer 3*, and with *Peer 1*, he exchanged two phone calls during the experiment.

Our hypothesis in this experiment is that the algorithm would select *Peer 1* first when looking for a contact because *Peer 1* was most frequently co-located with the *Contella user*, and *Peer 1* was also involved in 40% of the *Contella user's* total number of phone calls. To support this hypothesis, the sensors for detecting proximate Bluetooth IDs and location would also have to adequately reflect the actual activities of each of the participants. The following table is the Bluetooth-scanned Location-Time table containing an actual data set collected from this experiment. During the experiment, a

Bluetooth scan was activated every 3 minutes (which therefore scanned 20 times per hour).

<User 1> Percentage that User 1's phone was detected by Bluetooth = 45 % (37)

Location	9:00 – 10:00 am	10:00 – 11:00 am	11:00 – 12:00 am	12:00 am– 1:00 pm
Office	0% (0)	70% (14)	55% (11)	15% (3)
Park	5% (1)	0% (0)	0% (0)	0% (0)
Supermarket	0% (0)	0% (0)	0% (0)	0% (0)
Cafeteria	0% (0)	0% (0)	0% (0)	0% (0)
Home	10% (2)	0% (0)	0% (0)	0% (0)

<User 2> Percentage that User 2's phone was detected by Bluetooth = 44 % (36)

Location	9:00 – 10:00 am	10:00 – 11:00 am	11:00 – 12:00 am	12:00 am– 1:00 pm
Office	5% (1)	65% (13)	100% (20)	10% (2)
Park	0% (0)	0% (0)	0% (0)	0% (0)
Supermarket	0% (0)	0% (0)	0% (0)	0% (0)
Cafeteria	0% (0)	0% (0)	0% (0)	0% (0)
Home	0% (0)	0% (0)	0% (0)	0% (0)

<User 3 Percentage that User 3's phone was detected by Bluetooth = 11 % (9)

Location	9:00 – 10:00 am	10:00 – 11:00 am	11:00 – 12:00 am	12:00 am– 1:00 pm
Office	0% (0)	0% (0)	0% (0)	0% (0)
Park	0% (0)	0% (0)	0% (0)	0% (0)
Supermarket	0% (0)	0% (0)	0% (0)	0% (0)
Cafeteria	0% (0)	0% (0)	0% (0)	45% (9)
Home	0% (0)	0% (0)	0% (0)	0% (0)

Table 6.9 Bluetooth Location-Time table

The current location of the *Contella* user was correctly detected via GPS in the experiment, but, as seen in the table, Bluetooth did not always detect other proximate Bluetooth devices. For example, the phones of *Peer 1* and *Peer 2* were detected only 70% and 65% of the time, respectively, during the time period of 10:00 – 11:00 am, when in fact, they were co-located during this whole period. However, in other cases, the Bluetooth scanning adequately reflected situations in which users were co-located.

On the other hand, from 9:00 – 10:00 am as seen in Table 6.9, the Bluetooth device detected *Peer 1*'s phone only 15% of the time. Although this result might seem incorrect, this was actually a location transition period during which *Peer 1* and the *Contella* user were both traveling between their respective "Office" and "Home"

locations. Also, the system recorded this detection that occurred at the outside the known location and updated the total count of the Bluetooth scan for *Peer 1*.

The algorithm for finding contacts computed its results based on the data set collected by the *Contella* user’s phone. Overall, as we expected, the algorithm picked *Peer 1* as its first contact. The results produced by the algorithm, which we described in Chapter 5, are shown in Table 6.10 below.

Table 6.10 Results of the contact-finding algorithm

Location	9:00 – 10:00 am	10:00 – 11:00 am	11:00 – 12:00 am	12:00 am– 1:00 pm
Office	<b>Peer 2 (P = 1.44)</b> Peer 1 (P = 0.85) Peer 3 (P = 0.71)	<b>Peer 1 (P = 1.37)</b> Peer 2 (P = 0.92) Peer 3 (P = 0.71)	<b>Peer 1 (P = 1.18)</b> Peer 2 (P = 1.11) Peer 3 (P = 0.71)	<b>Peer 1 (P = 1.06)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.58)
Park	<b>Peer 1 (P = 1.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)
Cafeteria	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 3 (P = 1.35)</b> Peer 1 (P = 0.85) Peer 2 (P = 0.44)
Home	<b>Peer 1 (P = 1.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)	<b>Peer 1 (P = 0.85)</b> Peer 3 (P = 0.71) Peer 2 (P = 0.44)

The P values in the table are calculated based on the activity logs collected by the *Contella* user’s phone. The following formula is a mathematical representation of the computation in the algorithm that weighs each peer in order to sort contacts.

$$P\text{-total} = \text{Sum}( p(\text{Contact} | \text{Location}, \text{Time}) + \\ p(\text{Contact's total detection by Bluetooth}) + \\ p(\text{Contact's total phone call}) )$$

As shown in the results in Table 6.10, the algorithm selects *Peer 1* 85% of the time, and this result supports my hypothesis. In the case where the *Contella* user is in the “office” location from 9:00 – 10:00 am, the algorithm chooses *Peer 2* (officemate). This result makes sense because the *Contella* user spent the most of this time period with *Peer 2* in the “office”. However, if *Peer 1* (friend & officemate) is near the “office” during this time, the algorithm selects *Peer 1*. In addition, when the *Contella* user is in the “Cafeteria” location from 12:00pm to 1:00 pm, the algorithm picks *Peer 3*. This also makes sense because the *Contella* user is only co-located with *Peer 3* in the “Cafeteria.”

The results generally made sense to all the participants of the experiment. However, there was a potential experimental problem that could lead to questionable results.

The system selects *Peer 3* as the second best contact for the *Contella* user. In some cases, this result makes sense because frequent phone calls indicate a strong social relationship between these two peers. Consequently, we considered the phone calls and co-location as having the same weight when calculating the P value in the algorithm. However, if the total interaction time between peers is considered, then the algorithm should give more weight to co-location than to phone calls. Thus, the algorithm requires further improvements; performing more real-life experiments will give us additional insight about how to weigh the different social activities in the algorithm.

To sum up, during the experiment, the system effectively collected log information from Bluetooth, GPS, and phone calls. We also confirmed that the experiment supported our hypothesis that the algorithm would pick *Peer 1* most of the time. However, we found that a potential problem exists in figuring out how to weigh different activities to infer social relationship. Although the algorithm works in general, it is hard to conclude whether it will work reliably in real life situations outside of this simulated experiment. However, the purpose of the algorithm is not to find optimal contacts, as mentioned in Chapter 5, but to find any peers who might have some connection with our loved ones. In that respect, the system served its purpose well in this experiment.





## Chapter 7

### Conclusion and Future Work

#### 7.1 Contribution

The system described in this thesis touches on three different areas, each of which is aimed at enhancing our security and safety in daily life through the use of mobile phones. We have covered the retrieval of historical crime information from Crime-Log Servers (“Where You Are”), development of a route detection algorithm (“Where You Are Going”), and a method for increasing connectivity between people for the purposes of requesting and providing help remotely (“Who Is Around You”).

**Where are you:** The Risk-Meter showed how we can enhance our awareness of crimes around us for the purpose of enhancing our security and safety. The thesis showed that police logs have a great potential to be used by the public. With an easy access to this information, as well as good summarization techniques, it is possible that they can increase our situational awareness within our surroundings. We successfully showed a ‘proof of concept’ by implementing the system in GPS equipped mobile phones and delivering a summary of crime information where it is relevant to user’s spatial and temporal location. In addition, the system allows users to choose which crime information is to be summarized and displayed on their screen. The simple interface of the Risk-Meter also contributes to its potential to be adopted as a service that provides easy access to summarized risk information. In fact, there are already plans to introduce the Risk-Meter to incoming students during the Fall 2006 orientation at MIT, Boston University and Harvard<sup>12</sup>.

**Where are you going:** Utilizing the capability of location awareness through GPS, we developed a route detection algorithm in a mobile phone. Previously, our route detection algorithm ran on a desktop server machine and the mobile phones were required to stream GPS information to this external server to analyze data for the route

---

<sup>12</sup> by the Student-Alumni Committee on Institutional Security Policy

detection. This provided two potential problems; i) scalability issues, and ii) private information is constantly tracked by third party.

Scalability is a major issue with centralized servers because, in order to share information between peers, peers are constrained to subscribe to same server or service. For example, it is impossible to communicate between MSN messenger, Yahoo messenger, and AOL Messengers. The phone version of the route detection algorithm solves this potential problem by keeping the users' private information in the users' hands. In addition, because the phones exchange information directly between phones there is no need to build or maintain a server to relay this information between phones.

The algorithm in this thesis is also optimized for using only the phone to detect routes. The previous algorithm that ran on the desktop was based on a standard pattern matching algorithm and used Matlab. It was simply inefficient to use the same algorithm and transplant it into the phone because it requires lot much memory as well as significant computation power to run the algorithm. Our approach was to build a new algorithm, optimized for route detection, which uses less memory and computation power so that it will sufficiently run on modern mobile phones. The algorithm is very successful, as can be seen in the evaluations described in the thesis. Although there are many parts to be improved in the algorithm, it is currently robust enough to be used as a platform for other projects.

**Who is around you:** This aspect of the project broke new ground in the sense that phones can now find other, familiar phones as a communication channel resource. Then, when necessary, the phone of an intended recipient can relay messages to other 'peer' phones with the intention that important messages might be more successfully delivered in person (by the owner of the 'peer' phone). The system is originally designed for finding our loved one's peers and requesting their help. In the thesis, we introduced this novel system to show how we can design a system that learns about surrounding phones, and can relay a message to those surrounding phones without revealing the private information of the help message recipient.

## 7.2 Future Work

In the system we describe, there are a number of interactions between the three major components, yet there are many parts that have a potential to be integrated more tightly. Joe Dvorak has proposed that the route detection algorithm can be improved by involving not only information provided by Risk-Meter but also the locations of caregivers and peers of *Contella* phone users. Because multiple routes can exist between two locations, the system can recommend an optimal route that takes into consideration the predicted duration to reach at destination, the corresponding risk and the distances from caregivers when users arrive at each of the intermediate points.

In addition, we can improve the system for finding contacts. In the thesis we described one way caregivers can find peers of care-recipients for help, but we can also consider that care-recipients may wish to ask for help from their peer's care-givers should their own care-givers not be available. This creates an expansion of connectivity between pools of caregivers and recipients. People could contact each other and help each other in the pools, with each pool containing people who are not complete strangers. This allows the system to be used as a communication network for communities who may or may not have other direct communication channels available. Because the system collects information that can be used to infer social relevance between people, the network of community will dynamically reform to reflect current social dynamics in order to provide connectivity and communication to people who are socially related to each other.



# Appendix A

## Route Templates Collected during the Experiment

### Index:

#### Phone # 1's Route Template

- Location Database
- Route Database
  - Route 1
  - Route 6
  - Route 3

#### Phone #2's Route Template

- Location Database
- Route Database
  - Route 2
  - Route 4
  - Route 5

#### Data Format:

- Location Database
  - Location: [LocationName], [Lat],[Lon],[Accuracy],[LocationID]
- Route Database
  - Route: [RouteID],[LocationID],[LocationID], [number of *iPoints*]
  - *iPointID*: [*iPointID*] :[Distance from previous *iPoint*], [Lat], [Lon]
  - *iPoint* : [index], [*iPointID*], [accumulated distance], [Accumulated Elapsed Time] :[Latitude], [Longitude]

## Bibliography

- [1] ARJIS, The Automated Regional Justice Information System from <http://www.arjis.org>
- [2] Blythe, A., Wright, C., Monk, F. (2004) Little bother. Personal and Ubiquitous Computing, Volume 8 , Issue 6, 402 – 415, Springer-Verlag, London, UK
- [3] C.Y. Lee, “An Algorithm for Path Connectivity and its Applications,” *IRE Trans. on Electronic Computers*, Vol. 10, No. 3, Sept. 1961, pp. 346-365
- [4] Eagle, N., Pentland, A., (March 2006) Reality mining. *In Proceeding of Personal and Ubiquitous Computing. Volume 10, Issue 4.*
- [5] Editors of Business week, (OCTOBER 31, 2005). "Working Late" Won't Work Anymore, *Business Week online*, retrieved on November 14, 2005 from [http://www.businessweek.com/magazine/content/05\\_44/b3957069.htm?chan=tc](http://www.businessweek.com/magazine/content/05_44/b3957069.htm?chan=tc)
- [6] Fawcett, J., Robinson, P., (May 2000) Adaptive routing for road traffic., *Computer Graphics and Applications, IEEE Volume 20, Issue 3*
- [7] I. Getting, “The Global Positioning System,” *IEEE Spectrum*, Vol. 30, No. 12., Dec. 1993, pp. 36-38, 43-47
- [8] Kingsley, T., Kathryn P., (OCTOBER 2000). Getting to Know Neighborhoods, National Institute of Justice from <http://www.ncjrs.gov/pdffiles1/jr000245d.pdf>
- [9] LG's security phone. (2005 March 9) *Engadget home page*. Retrieved November 14, 2005 from <http://www.engadget.com/entry/4338125245845446/>
- [10] Live Personal Assistance Anytime, Anywhere At The Touch Of The Big Red Button, (2005). *SafeGuardian*. Retrieved November 14, 2005 from <http://www.safeguardian.com>
- [11] McGinity, M., (2004). Wearing wireless safety net. *Communications of the ACM*, Volume 47 Issue 9, 15 – 18, ACM Press.
- [12] Marmasse, N. (2004) Providing Lightweight Telepresence in Mobile Communication to Enhance Collaborative Living. *Ph.D. dissertation, MIT Media Laboratory, 2004.*
- [13] Mermasse, N., Schmandt, C (2003). Safe and Sound. *ACM CHI'2003*, 726 – 727, ACM Press.
- [14] Mehta, V., Zarki, E., (2004). A Bluetooth Based Sensor Network for Civil Infrastructure Health Monitoring. *Wireless Networks*, Volume 10 Issue 4, 401 – 412, Kluwer Academic Publishers
- [15] Kim Tae-gyu, (February 23, 2004). Mobile Carriers Provide Handsets for Security, *The Korean Times*, retrieved on November 14, 2005 from <http://times.hankooki.com/lpage/biz/200402/kt2004022318280311860.htm>

- [16]N. Eagle, (2005). "Machine Perception and Learning of Complex Social Systems", Ph.D. Thesis, Program in Media Arts and Sciences, MIT.
- [17] Palen, L., (2002). Mobile technology in connected life. *Communications of the ACM*, Volume 45 Issue 3, 78 – 82, ACM Press.
- [18] Plomp, J., Tealdi, P. (2004). Ambient Intelligent Technologies for Wellbeing at Home. Proceedings of the 2nd European Union symposium on Ambient intelligence EUSAI '04, Vol. 84, 81 - 82, ACM Press.
- [19] SK Telecom's i-Kids Service, (May 24, 2005). *SK-Telecom. Press release*, retrieved on November 14, 2005 from [http://www.sktelecom.com/eng/cyberpr/press/1196855\\_3735.html](http://www.sktelecom.com/eng/cyberpr/press/1196855_3735.html)
- [20] SK Telecom to provide real-time mobile home security services, (April 07, 2004). *Mobile Tech News*, retrieved on November 14, 2005 from <http://www.mobiletechnews.com/info/2004/04/07/105914.html>
- [21] Teen Arrive Alive, (2005). *Teen Arrive Alive home page*, retrieved on November 14, 2005 from <http://www.teenarrivealive.com>
- [22] Zaworski., M., (January 2006). Automated Information Sharing, National Institute of Justice from <http://www.ncjrs.gov/>