

Ringling In The Rain:

An Agent Based Weather Warning System

by

Chao-Chi Chang

B.S., Computer Science and Information Engineering, National Taiwan University (2000)
M.S., Computer Science, State University of New York at Stony Brook (2005)

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

September 2007

© Massachusetts Institute of Technology 2007. All rights reserved.

Author.....
Program in Media Arts and Sciences,
August 10, 2007

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

Accepted by.....
Deb Roy
Chairperson
Department Committee on Graduate Students
Program in Media Arts and Sciences

Ringin In The Rain:

An Agent Based Weather Warning System

by

Chao-Chi Chang

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on August 10, 2007, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

People's daily lives are impacted by lots of dynamic environmental information, such as weather and traffic. Though most of this information is available on the Internet, there is no easy way for someone to access it while in a mobile state. More importantly, people do not have a constant need for this kind of information unless there is a significant change that may impact their current or future activities.

This thesis creates a distributed multi-agent architecture that uses GPS-enabled cell phones to build a mobile service development framework. The goal of this framework is to build mobile services to deliver timely changes in environmental information that could impact a user's current or future activities. A weather warning system for bicycle riders is built based on this framework to demonstrate its utility. This weather warning system tracks bicycle riders as well as current weather patterns; it warns riders about the risk of getting caught in the rain if it estimates that riders are heading into areas where rain is predicted, or if they are moving too far from shelter to be able to reach it before the rain starts. With this system, bike riders can make more informed decisions about which routes to take to avoid encounters with nasty weather conditions.

The objective of this research is: 1) to examine the feasibility of using agent techniques and GPS-enabled cell phones to create the mobile service development framework, and 2) to investigate the new generation of location-based services in which the movement and status changes of multiple targets are considered.

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: _____

Henry Holtzman
Research Scientist
M.I.T. Media Laboratory

Thesis Reader: _____

Joseph Dvorak
Research Affiliate
Motorola Scientist in Residence

Acknowledgments

First of all, I would like to thank my advisor, Chris Schmandt, who gave me the ticket to enter this wonderland, the MIT Media Lab. I have learned a lot from him on a wide range of topics, including speech interface research activities, patent lawsuits, geoscience, meteorology, hiking tips, as well as the importance of balance between work and family life, which I converted into real action to pursue my girlfriend, Holly, while I was also buckling down on my thesis work. In addition, I greatly appreciate the freedoms he gave me to work at my own pace and to be involved in extracurricular activities during my two years' stay at the lab. These freedoms brought me lots of fruitful results and experience. For example, I went to Paris for the Alcatel workshop to brainstorm their business future in my first year, and I went to Copenhagen for the Nordic Exceptional Trendshop 2006 exhibition to demonstrate my HiTV project on which I collaborated with other Media Lab students. He also encouraged me to apply for my internship at the India IBM Research Lab during the summer of 2006 and shared his experiences in India with me. I learned a lot from him, and I also very much enjoyed my stay in the Media Lab under his guidance.

I would also like to thank my other thesis readers, Henry Holtzman and Joe Dvorak, for their valuable time and input on my work, especially when I needed to finalize a new thesis topic because my original idea, a camera-based joystick interface of cell phones using phones' embedded cameras, was scooped by a similar idea proposed by Samsung. Henry and Joe got together with me and Chris to discuss possible new topics and to finalize the scope of the project, and they did so very quickly. That was why I still had enough time for my thesis work even though I needed to change my thesis topic in the middle of January. In addition, Joe also helped a lot in marketing my project in the cooperation between Motorola and the Media Lab, and he provided the necessary cell phones and data service for me to implement my project. I greatly appreciate Henry and Joe's time and assistance on my thesis work.

Jae-woo Chung, who is one year senior to me and currently pursuing his PhD, is the first person I would like to thank in our group. He shared with me his experience and tips about working in the lab, which helped me quickly get used to the Media Lab life. In

addition, his assistance of integrating his route learning and detection component created in his “Will You Help Me” project with the user movement estimation agent running on cell phones in my Ringing In The Rain project sped up my development time a lot. Finally, we had lots of fun time to brainstorm and work on projects together, such as Going My Way.

Matt Adcock, who was supposed to start at the lab at the same time as me but actually arrived one semester later, is another person in our group I would like to thank. He brought with him a broad range of knowledge in Human Computer Interaction research, and he acts as a live search engine to me. Most of the time, he can point me to the right place for the knowledge and information I need. He also brainstormed with me about the algorithms used in my thesis project. Actually, one of the current running storm movement estimation algorithms is inspired by Matt’s suggestion. It was fun to collaborate with Matt too. Our course project, Puff Button, got very good feedback in the sensor class we attended together.

I also want to say thank you to Rachel Kern, who graduated from our group last year. Though she is busy working at Nokia now, she still tried to help me a lot in editing and proofreading my thesis proposal and thesis in her limited spare time. Her professional English writing skills sharpened my proposal and this thesis quite a bit.

I would like to thank Jeff Goldenson and Paulina Modlibta, the other colleagues in our group, and Isabel Pesce, our UROP student. They helped me a lot in our group meetings to come up with a perfect name for my thesis, Ringing In The Rain.

David Chiou, a Media Lab alumnus and also my former boss when I worked in Groundhog Technologies Inc., was the person who introduced me to the Media Lab. Without his encouragement and advice, I may have not been able to get here. So I would like to thank him for introducing me to this wonderful place, and for the encouragement and advice he gave me for my application to the Media Lab.

Jackie Lee, James Teng, Edward Shen, Wu-Hsi Li, Francis Lam, and Anna Huang are Media Lab students who come from the same country, Taiwan, as I do, or from a place with a similar culture - Hong Kong. I would like to thank them specifically during my stay here. Because of them, I do not feel homesick here since I can always count on them to know what is happening in Taiwan, or easily find somebody with whom to share

Taiwanese dishes.

I am also thankful to all of the other students, faculty members, and staff who helped me during my stay here. Your help and advice during my two years in the lab has made this period a memorable and enjoyable experience in my life.

Thanks to all of my friends in the US and Taiwan for encouraging me in doing my work and accompanying me when I needed somebody to talk to, no matter when. I would not have survived here without your support.

Lastly, I want to thank my great parents and sister for their spiritual and financial support in addition to the stipend from the lab. Their support helped me to focus on my study in the US without worrying about other life issues too much. Also, I would like to thank my lovely girlfriend, Holly, who distracted me from my thesis work in the beginning, but pushed me a lot the rest of time. Her consideration and cheeriness have been a powerful support in helping me to finish my thesis.

Thank you all!!

Table of Contents

1. Introduction.....	15
1.1 Problem.....	15
1.2 Proposed Solution.....	15
1.3 Organization Of This Thesis.....	16
2. Related Work.....	17
2.1 Inspirations.....	17
2.2 Distributed Multi-Agent Architecture.....	18
2.3 Just-In-Time Information Delivery System.....	19
2.4 Weather Information Retrieving.....	20
3. System Details.....	24
3.1 System Architecture and Control Flow.....	24
3.1.1 Data Collection Agent.....	27
3.1.2 Personal Information Management Agent.....	29
3.1.3 User Movement Estimation Agent.....	29
3.1.4 User Goal-Maintaining Agent.....	30
3.2 User Interface and Usage Scenarios.....	34
3.2.1 User Interface.....	34
3.2.2 Usage Scenarios.....	36
3.3 User Movement Estimation Algorithms.....	38
3.3.1 Route Learning and Detection Algorithm.....	38
3.3.2 Linear Approximation.....	39
3.4 Storm Movement Estimation Algorithms.....	39
3.4.1 Storm Tracker Information Extraction Algorithm.....	40
3.4.2 Image Processing Based Algorithm.....	41
3.4.2.1 Radius Scan Method.....	42
3.4.2.2 Center Of Gravity Method.....	44
3.5 Weather Prediction Algorithms.....	47
4. Evaluation.....	53
4.1 Results Analysis.....	53

4.1.1 Collected Data Analysis.....	53
4.1.1.1 Storm Movement Direction Estimation Analysis	53
4.1.1.2 Storm Movement Estimation Performance Analysis.....	56
4.1.2 User Experience	57
4.2 Problems Found For Current System.....	58
4.2.1 General Problems.....	58
4.2.2 Problems For Storm Tracker Information Method	60
4.2.3 Problems in the Radius Scan Method	63
4.2.4 Problems in the Center of Gravity Method.....	63
4.3 Possible Enhancements	64
5. Conclusions.....	66
5.1 Conclusions.....	66
5.2 Future Work	67
Appendix A. Ringing In The Rain Project File Structure.....	69
Appendix B. Ringing In The Rain Communication Message	70
References.....	71

List of Figures and Tables

Figures:

Figure 1. Ambient Device’s Information Orb, which can be used to display weather information.....	21
Figure 2. Ambient Device’s Weather Umbrella, on which the handle will blink when it might rain.	21
Figure 3. Commercial lightning/storm detector products. Left: ThunderBolt, Right: SkyScan.....	22
Figure 4. Ideal architecture and components diagram of Ringing In The Rain.....	25
Figure 5. Real architecture and components diagram of Ringing In The Rain	26
Figure 6. Control flow of data collection agent	27
Figure 7. Radar image example	28
Figure 8. Storm tracker information example.....	29
Figure 9. The main control flow of user movement estimation agent	32
Figure 10. The main control flow of user goal maintaining agent.....	33
Figure 11. Default information window of Ringing In The Rain client application	34
Figure 12. Main menu of Ringing In The Rain client application.....	34
Figure 13. Display current radar image.	35
Figure 14. Ringing In The Rain Configuration.....	35
Figure 15. User remains in a known location with rain approaching	36
Figure 16. User is in transit along a known route with rain approaching.....	36
Figure 17. User stays in an unknown location with rain approaching.....	37
Figure 18. User is in transit along an unknown route with rain approaching.....	37
Figure 19. Currently raining in user’s location.....	37
Figure 20. User will not encounter rain	37
Figure 21. Typical radar image from weather data source	41
Figure 22. Original radar image.....	42
Figure 23. Ray traced radar image.....	42
Figure 24. It’s not raining yet at the user’s current location.	47
Figure 25. It’s raining at the user’s current location.	47

Figure 26. Illustration of the weather prediction based on route learning and detection algorithm.....	50
Figure 27. Illustration of the weather prediction based on linear approximation method	51
Figure 28. Reliability of the estimation result	55
Figure 29. Radar image with clutter option off.....	59
Figure 30. Same radar image as Figure 29 with clutter option on.....	59
Figure 31. Radar image with clutter option off.....	59
Figure 32. Figure 31 with clutter option on	59
Figure 33. Partially scanned radar image.....	60
Figure 34. Radar station is down	60
Figure 35. Radar image of a special condition that the storm tracker information method may not handle well.....	61
Figure 36. Storm tracker table of a special condition that the storm tracker information method may not handle well.....	62
Figure 37. Radar image after processed by radius scan method.....	63
Figure 38. Radar image one frame later than Figure 37 after processed by radius scan method.....	63

Tables:

Table 1. Pseudo code of Radius Scan Method.....	43
Table 2. Pseudo code of Center of Gravity computation method for the largest storm ...	46
Table 3. Pseudo code of Center of Gravity computation method for the whole storm system	47
Table 4. Pseudo code weather prediction process when route learning and detection result is used.....	49
Table 5. Pseudo code weather prediction process when linear approximation result is used.	49

Chapter 1

Introduction

The main problem this thesis tries to solve will be described in the first section of this chapter. The following section will introduce the proposed solution for that problem. The last section will explain how this thesis is organized.

1.1 Problem

In many countries, the population of bicycle riders has surged during the past few years. This surge was due to a variety of reasons, including the cost of gas, traffic control regulation, health issues, and leisure [8, 15]. Regardless of the reasons that people ride bicycles, one of the most frustrating things for them is being caught in rain that they could have avoided by taking a detour, if they had known they were heading toward a storm beforehand. This frustration leads to a bigger and more general question: is there any prompt and precise way to offer people information which may impact their current or future activities while they are in a state of mobility with limited information accessibility? More specifically regarding the scenario at hand, how can we best inform bicycle riders that they are riding toward an approaching storm, so that they are equipped with enough information to make an informed decision about their route?

1.2 Proposed Solution

Lots of dynamic environmental information is available on the Internet, such as weather data that can be used to solve the aforementioned problem. Offering people this information promptly and precisely may impact their current and future activities while they are in a mobile state with limited information accessibility. To use the dynamic environmental information on the Internet for solving the problem, I propose combining a distributed multi-agent architecture with GPS-enabled cell phones to form an application framework that enables the creation of various mobile services which can deliver such information to users' cell phones at the right time and place.

This framework defines several kinds of software agents, including: 1) A data collection agent, which is responsible for collecting and monitoring the dynamic

environmental information on the Internet, 2) A user-movement estimation agent, which tracks and estimates users' movements based on current moving patterns and movement estimation algorithms, 3) A personal information management agent, which manages users' information such as calendars, address books, and emails, and offers auxiliary personal information to other agents, and 4) A user goal-maintaining agent, which is in charge of achieving a user's goals, using the results generated by the previous agents.

As a result of this work, frustrating situations - such as getting caught in the rain while biking, or missing an important appointment due to an accidental transportation service shutdown or traffic jam – can be avoided by delivering related information to users' GPS-enabled cell phones in advance. This thesis will explain how to use this framework to implement a weather warning system for bicycle riders to solve the problem in the aforementioned scenario as the demonstration for the utility of this framework.

1.3 Organization Of This Thesis

The remainder of this thesis is organized as follows: In Chapter 2, I will discuss the inspirations for the Ringing In The Rain system, as well as some of the related research projects that have been done along this direction. Chapter 3 gives a more in-depth system description, along with the technical details of the architecture of the system, the integration of all of the system components, and how each algorithm used in the system works. Chapter 4 presents the evaluations of the algorithms used in the Ringing In The Rain system, and a discussion of special conditions that the current algorithms cannot handle well. Finally, in Chapter 5, I draw conclusions about the system I have built, and also highlight areas for future work to be done to further build upon and improve the Ringing In The Rain system.

Chapter 2

Related Work

This chapter will describe in the first section how this thesis topic originated by introducing some previous research that inspired the creation of this project. The second section will compare some existing distributed multi-agent architectures and the reason why I selected one in particular for the Ringing In The Rain system. In the third section, I will introduce other just-in-time information delivery systems with similar generic goals to that of Ringing In The Rain - delivering personalized information at the right time and right place. The last section will compare Ringing In The Rain to existing popular methods of obtaining weather information.

2.1 Inspirations

Ringing In The Rain was inspired primarily by two previous research projects, which were also conducted in the Speech and Mobility Group (also known as Speech Interface Group) at the MIT Media Lab, namely comMotion [9, 10] and Will You Help Me [3]. I will describe these projects and how they inspired the Ringing In The Rain project briefly in the following section.

comMotion is a computing environment that exploits mobile users' locations to provide context-aware messages to them. comMotion has a behavior-learning agent to learn about users' mobility patterns and extract salient locations in their lives. Therefore, messages or reminders can be sent to users when they appear in the relevant contexts. For example, a shopping list can be delivered to users when they are near the supermarket. comMotion was a landmark early project exploring how we can utilize spatial and temporal context to deliver useful information to users. However, in the comMotion system, most information is preset to be delivered in specific contexts rather than generated dynamically by computation.

Risk Meter, a sub-system in the Will You Help Me project, starts offering statistical information, crime logs in this case, according to the current time and users' locations. Furthermore, based on users' mobility patterns and the current time, Risk Meter can also provide the computed risk of each route that users usually take. Therefore,

users can determine which route to take for a statistically safer journey. The Risk Meter in Will You Help Me furthers the comMotion concept with 1) computed information, i.e. the statistical risk information in Risk Meter system, and 2) by broadening the context from points to lines, i.e. from individual specific locations to routes between these locations. However, in the Will You Help Me system, the information delivered to users is still relatively static since it is computed from statistical crime log data.

Inheriting the concept from comMotion and Will You Help Me of delivering information to users according to their spatial and temporal context, the Ringing In The Rain project enhances the completeness of this concept by introducing the idea of a prompt and precise way to offer people dynamic environmental information. Such information may impact their current or future activities while they are in a state of mobility with limited information accessibility. In addition, the route learning and route detection component used in Will You Help Me is embedded in the user-movement estimation agent in Ringing In The Rain as one of its user movement estimation algorithm providers. Therefore, these two projects not only inspired the birth of the main idea in Ringing In The Rain, but they also contributed to its implementation.

2.2 Distributed Multi-Agent Architecture

Caglayan et al. [1] compared two popular access frameworks for an intelligent agent - a blackboard model and a software bus model - for communications between agents. The blackboard model uses a shared database, called a blackboard, which all agents can access. Thus agents can communicate with each other via the blackboard. The software bus model is similar to the term bus used in hardware. Each agent can communicate with another specific agent via the software bus. Caglayan et al. [1] suggested using the “software bus” model because of a scalability problem with the blackboard model, even though the blackboard model offers better sharing among agents.

Several general-purpose multi-agent architectures are built based on these two models. Cohen et al. [4] and Helsinger et al.[5] extended the blackboard model and solved its scalability problem by introducing a hierarchical blackboard server architecture and a multi-layer application interaction architecture, respectively. Iglesias et al. [6] adopted the software bus model solution to develop their network model in which their

network agent acts very similarly to the communication middleware in the software bus model.

I decided to follow Cohen and Helsinger's [4, 5] architecture to take advantage of its capabilities to share information among agents; for example, the agents that maintain the user's goals can easily share related information collected by the various data collection agents.

2.3 Just-In-Time Information Delivery System

The goal of a Just-In-Time information delivery system is to deliver personalized information to users at exactly the right time when the information will be most useful and relevant. The agent-based system Soltysiak et al. [16] proposed is one typical just-in-time information delivery system example in the Internet era. The system collects and manages a user's personalized information, and delivers it to the user when he or she is most likely to need it. However, a user's spatial contexts are not considered in the system design.

While cell phones become more and more popular, spatial contexts play increasingly important roles in the data collection and management process. Patents filed by Chan and Chang [2] and Johnson [7] show the evolution of bringing in spatial contexts into just-in-time information delivery systems. Though these patents do not describe how a system will predict a user's future travel path, they both claim they will deliver search results to a user's mobile device at the right time and place with regards to areas that the user may travel to and taking into account interests stored in the user's profile. Although the Ringing In The Rain system adopts a similar implementation to the aforementioned projects, i.e. an agent based on a client/server architecture between mobile devices and server computers, due to the computation power and communication bandwidth on mobile devices, the design of the proposed multi-agent framework used by Ringing In The Rain is very flexible. This means all system components can be hosted and run on mobile devices easily, once capable mobile devices emerge on the market. In addition, Ringing In The Rain focuses not only on queried information, such as Internet queries according to users' profiles and locations, or on computed information from statistical data, like the statistical risk in the Risk Meter, but also on information

estimated from dynamic data on the Internet. For example, the storm movement information used in Ringing In The Rain is estimated from radar images and other types of sources found on various weather web sites.

2.4 Weather Information Retrieving

The most popular way to access weather information is on the Internet via computers. Many weather websites, such as The Weather Channel [30] or Weather Underground [32] allow users to retrieve relevant weather information by entering a local or a destination zip code. Additionally, various newspapers provide weather forecast delivery services via fax and/or pager to travelers, based on the area code of a contact number in his/her destination city. With the popularity of network-connected handheld devices, such as cell phones and Blackberries, it is becoming easier and more common for people to access weather information via these devices.. The Phoneshell project done by Schmandt [14] also allows mobile phone users to call a number to get frequently updated weather information via speech synthesis. Meyerson [11] points out this trend of accessing weather information via mobile-networked devices. The problems with these solutions is that users retrieve information in a passive way since the frequency with which they visit a weather website may not match the frequency or time of a weather data update.

This passive access problem is solved in various ways later on. Morris [12] tried to solve it by parsing timely weather information from websites and emailing subscribers whenever there was an update. AccuWeather's [17], DON'T GET WET .NET [20], National Weather Service [25], and Google [22] use similar approaches but inform users of weather updates via cell phone. Though these solutions offer a more active way of pushing relevant weather information to users, they still have some common drawbacks: 1) The resolution of the weather information is often not precise enough because the weather websites use a zip code-based area. Rainfall during a thunderstorm season is much more localized and dynamic, and 2) They do not consider a user's current location or activity, which may result in the delivery of useless information to users.

XM Radio's Personal Weather Tracking System [33] is currently the most mature solution since this system obtains a user's travel plans via route planning done by a

navigation system. Therefore, this system can provide weather information along the route on which a user is driving. Bushnell's ONIX 400 [27] is the first portable device utilizing XM Radio's service, which solves the portability problem. However, the cost of this kind of solution is high, since it includes both hardware plus service, and it only runs on proprietary devices with limited two-way communication ability, which is not a general and extensible solution.

Ambient Devices [18] delivers weather information in quite a different way. Their products, as shown in Figure 1 and 2, utilize the pager network and each pager base station mainly broadcasts local information. Users glance at their devices to know the weather situation, which is indicated by changes in color and light. However, their products cannot solve the location resolution and portability problems.



Figure 1. Ambient Device's Information Orb, which can be used to display weather information



Figure 2. Ambient Device's Weather Umbrella, on which the handle will blink when it might rain.

The Berkeley Building, also known as the Old John Hancock Building, located in Boston, MA, broadcasts a weather beacon via red and blue lights [19], which use a code to indicate the local weather forecast. The code uses a popular rhyme as a mnemonic:

*Steady blue, clear view.
Flashing blue, clouds due.
Steady red, rain ahead.
Flashing red, snow instead.*

Users can know the local weather forecast by glancing at the weather beacon on the top of the building, which is very similar to what Ambient Devices offers. Though this method is not very portable or cheap, it offers an easy way for many people to know the local weather forecast simultaneously.

Commercial lightning/storm detector products, such as ThunderBolt Storm Detector [31] and SkyScan Lightning Detection Systems [29] as shown in Figure 3, offer another dimension of solutions for this problem. The mechanism of how they work is mainly based on their ability to pick up electromagnetic noise generated by huge flows of electric current, i.e. lightning in the thunderstorm, according to knowledge databases on the Internet [21, 24]. These products are designed for outdoor workers who are at an increased risk of getting struck by lightning. They mostly provide information about the distance between the user and the center of the thunderstorm center where the lightning usually occurs. In addition, it is possible to calculate the storm movement velocity information if consecutive lightning events are sensed. However, the information offered by these kinds of devices, namely distance and velocity of the storm center, is not enough to determine the storm's boundary or its direction of movement, not to mention that these systems are built proprietarily with one single goal and without extensibility.



Figure 3. Commercial lightning/storm detector products.

Left: ThunderBolt, Right: SkyScan

Ringling In The Rain proposes a generic solution, which uses GPS-enabled cell phones as the user interface to build the weather warning system. GPS-enabled cell phones can be used not only to display information but also to track users' locations and their estimated moving directions/speeds to offer more precise weather information. The

network connectedness, portability, and penetration rate of cell phones makes this solution both general and portable.

Chapter 3

System Details

This chapter will explain the system details, focusing specifically in the first section on how the architecture of Ringing In The Rain is designed, and on the user interface in the second section. The remaining sections will focus on explaining different algorithms used in estimating users' movements, storms' movements, and weather prediction respectively.

3.1 System Architecture and Control Flow

The Ringing In The Rain system is designed based on the distributed multi-agent architecture, which follows the philosophy of the blackboard model with the advantage of easily sharing information and scaling in multi-layer architecture [4, 5].

Following are the components needed for the Ringing In The Rain weather warning system; four kinds of agents have been created for this system: data collection agents, user movement estimation agents, personal information management agents, and user goal-maintaining agents. Data collection agents monitor and collect dynamic environmental information on the Internet whenever there is an information update. User movement estimation agents track a user's movements and generate estimated future movements based on a user's historical moving patterns with the assistance of estimation algorithms. Personal information management agents offer helpful information to other agents by retrieving information from users' calendars, address books, emails, and other personal data sources. User goal-maintaining agents use collected information, user movement estimation, and personal information to ensure that users' goals are met.

Figure 4 shows the ideal system architecture and component diagram of the Ringing In The Rain weather warning system. Agents can run independently within the same machine, such as on a single cell phone. They can also run on separate machines, for example, a cell phone with a server computer, and communicate with each other via shared blackboard.

Each data collection agent is in charge of one specific weather data source. The more weather data sources, the more data collection agents are needed in the system. The

personal information management agent talks to the user's calendar software, from which it extracts time and location information, and then it writes to the shared blackboard.

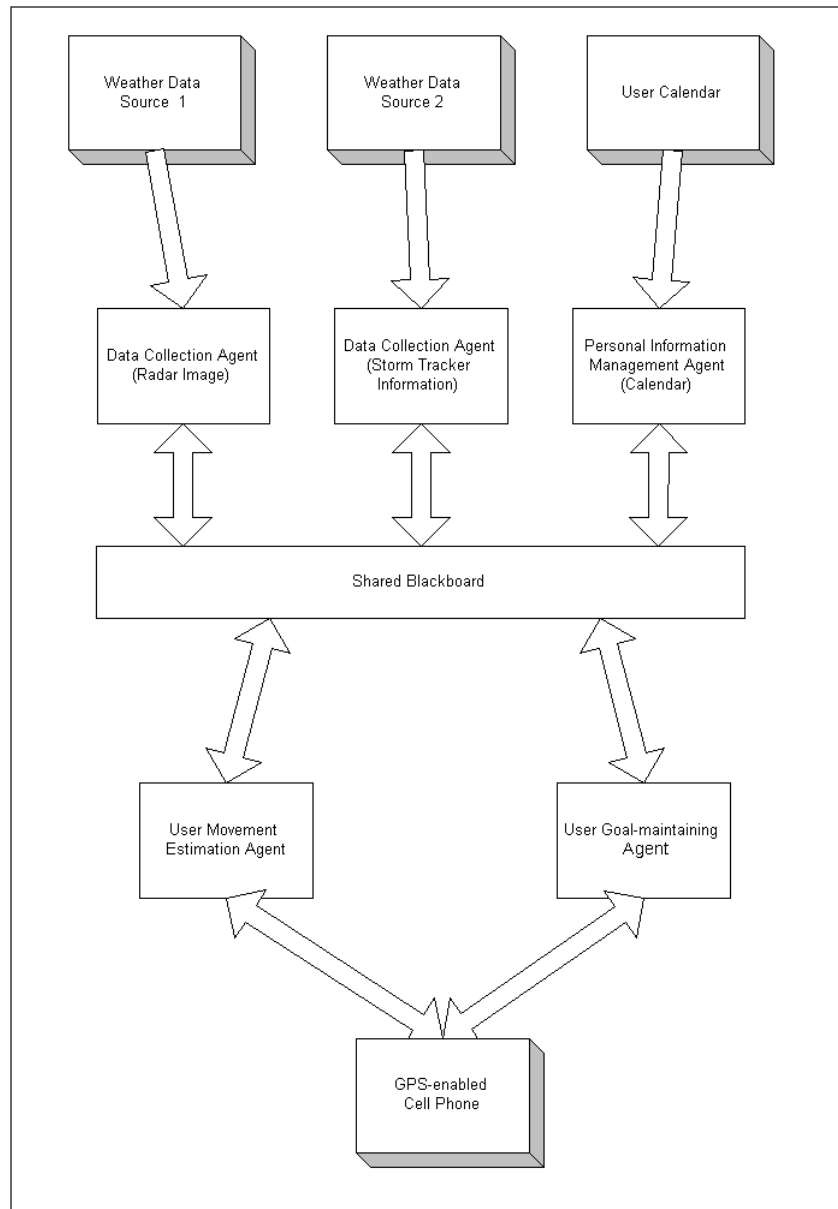


Figure 4. Ideal architecture and components diagram of Ringing In The Rain

In the current system implementation, due to the limitations of hardware computation power and cell phone communication bandwidth, the architecture and component diagram is slightly modified as shown in Figure 5. A client-server architecture design is used in this implementation, which requires both a cell phone and a

server computer. The cell phone runs the client application and hosts the user movement estimation agent. The server computer runs the server application and hosts the data collection agents, the personal information management agent, and the user goal-maintaining agent to operate the whole system. The details of how each agent works will be explained in later subsections.

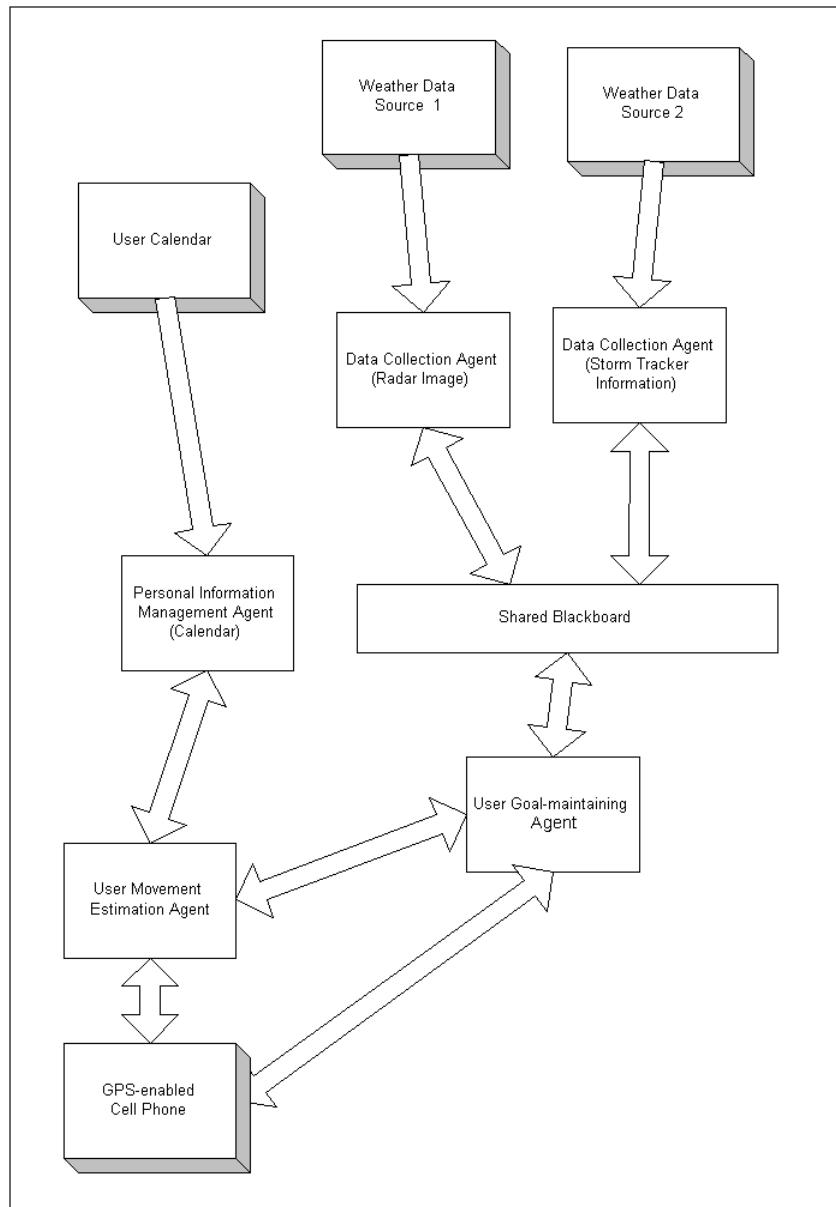


Figure 5. Real architecture and components diagram of Ringing In The Rain

3.1.1 Data Collection Agent

In the current system, two data collection agents are created and run on the server computer for each user to check Weather Underground [32] for periodic updates of weather information in different formats. Figure 6 shows the control flow of these agents.

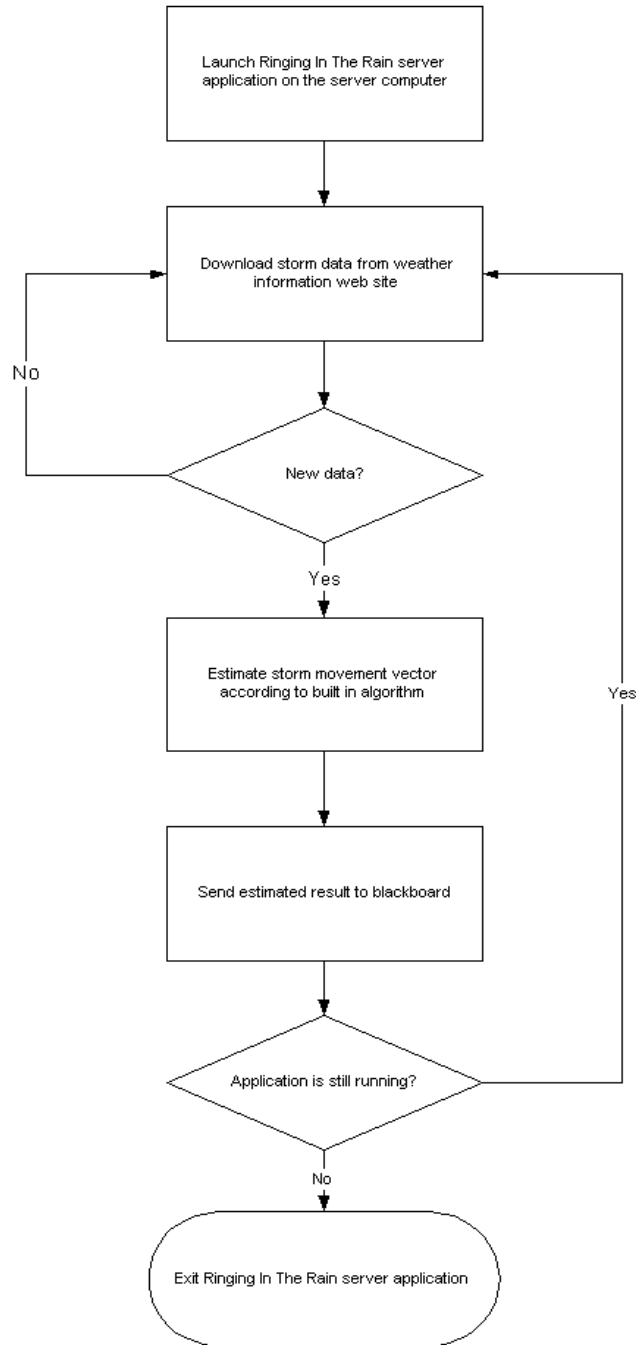


Figure 6. Control flow of data collection agent

One of these two data collection agents monitors the radar images of areas around the user's current location, checking for any appearance or movement of rain. This is shown in Figure 7. The other agent monitors the storm tracker information provided by weather sites if available; this is shown in Figure 8. Both of these agents download data from Weather Underground [32], and they send the updates they observe to the shared blackboard in this framework. The details of the algorithms used to transform the collected data to a storm movement vector will be explained in a later section in this chapter.

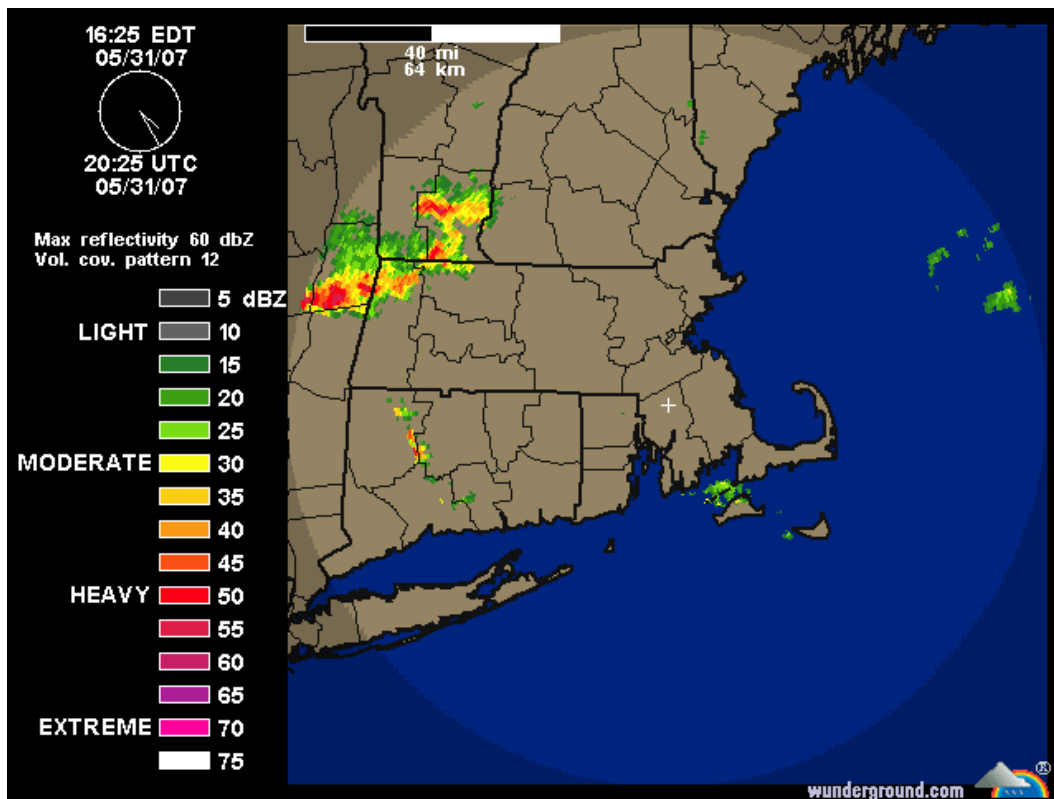


Figure 7. Radar image example

NEXRAD Storm Table ▼ = Tornado Vortex Signature ◆ = Mesocyclone ■ = Hail Storm

ID	Max	Top	VIL	Chance of Severe Hail	Chance of Hail	Max Hail Size	Speed	Direction (from)
X0	56 dBZ	19,000 ft.	35 kg/m ²	20%	50%	0.50 in.	20 knots	NNW (327)
Y0	53 dBZ	20,000 ft.	17 kg/m ²	0%	0%	0.00 in.	18 knots	N (354)
W0	52 dBZ	14,000 ft.	8 kg/m ²	0%	0%	0.00 in.	32 knots	NW (313)
K0	51 dBZ	13,000 ft.	9 kg/m ²	0%	0%	0.00 in.	32 knots	NW (315)
- More Information About Storm Data -								

Figure 8. Storm tracker information example

3.1.2 Personal Information Management Agent

The personal information management agent, currently running on the server computer, acts in an assistant role in this system. This agent's main function is to retrieve location and time information from a user's calendar entries and provide it to the user movement estimation agent. This data may help improve the route estimation results. It was originally designed to write the data from users' calendar entries to the shared blackboard. Instead, it has been implemented to communicate with the user movement estimation agent directly via a socket at the end, because the user movement estimation agent is the only agent that needs such information in the current system. I will explain in the next section how the user movement estimation agent utilizes the data provided by the personal information management agent.

3.1.3 User Movement Estimation Agent

The user movement agent runs on a cell phone and tracks users' movements with the GPS-enabled cell phones that they carry; the tracked movement data is used for movement estimation. There are two algorithms built in this agent to handle user movement estimation. One is the route detection algorithm proposed in Chung's master's thesis [3]; this algorithm is used if users' movements are within their normal moving patterns. In this case, the user movement estimation agent will communicate with the user goal-maintaining agent in the format: (Route ID, iPoint, Direction). If users are taking new routes that do not exist in their historical moving patterns, a linear approximation

result, based on the tracking data, will be used to estimate their movements. This information is in the format: (start latitude, start longitude, end latitude, end longitude, duration). The details of these algorithms will be described in later sections; the main control flow of the user movement estimation agent is shown in Figure 9.

The user movement estimation agent sometimes uses time and location information from users' calendar entries for better route estimation; the personal information management agent provides this calendar information. For example, suppose the personal information management agent returns a message, "11:00-lab", indicating the user's next appointment on his calendar takes place at 11:00 at the lab. If the user movement estimation agent receives this information at 8:30 AM, and there are no other appointments listed before 11:00 AM, the user movement estimation agent can probably assume the user is on his way to the lab or will be leaving for the lab relatively soon. Therefore, this information may help speed up the route selection decision in the algorithms used in the user movement estimation agent.

3.1.4 User Goal-Maintaining Agent

The user goal-maintaining agent in Ringing In The Rain has several tasks: 1) It handles the weather prediction request from the Ringing In The Rain client application running on the cell phone, 2) It predicts whether a user's movements and an area of predicted rainfall will intersect, and 3) It is responsible for delivering the weather notification. The user goal-maintaining agent will use a combination of the aforementioned rain area movement data stored on the shared blackboard and the user movements sent back from the cell phone with the weather prediction requests to calculate any possible intersections. If there is any intersection predicted in the near future, it will notify users via cell phone about the estimation it made. Figure 10 shows the control flow of the user goal-maintaining agent. Details of the weather prediction algorithms used in the user goal-maintaining agent will be described in a later section.

One rare interaction between the user movement estimation agent and the user goal-maintaining agent, which does not appear in the control flow chart, is the route synchronization, which means the route information in the user movement estimation agent and user goal-maintaining agent is out of synchronization. This is most likely to

happen in the following situations: 1) It is the user's first time using this system, so the user goal-maintaining agent does not have any stored route information from this user, and 2) The user movement estimation agent has learned some new routes. In either case, once the user goal-maintaining agent encounters a weather prediction request with unknown route information, it will ask the user movement estimation agent to synchronize the route information with it automatically.

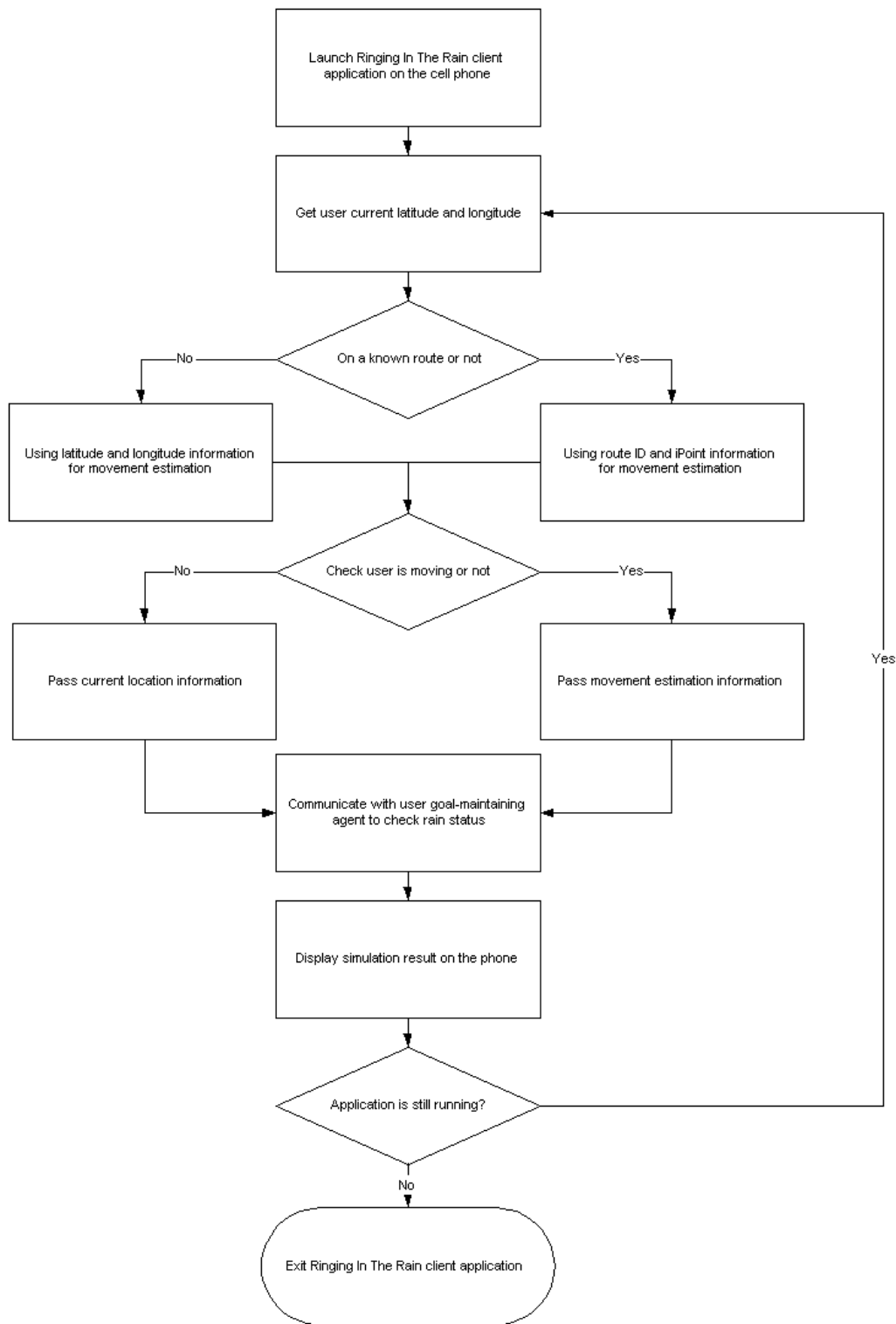


Figure 9. The main control flow of user movement estimation agent

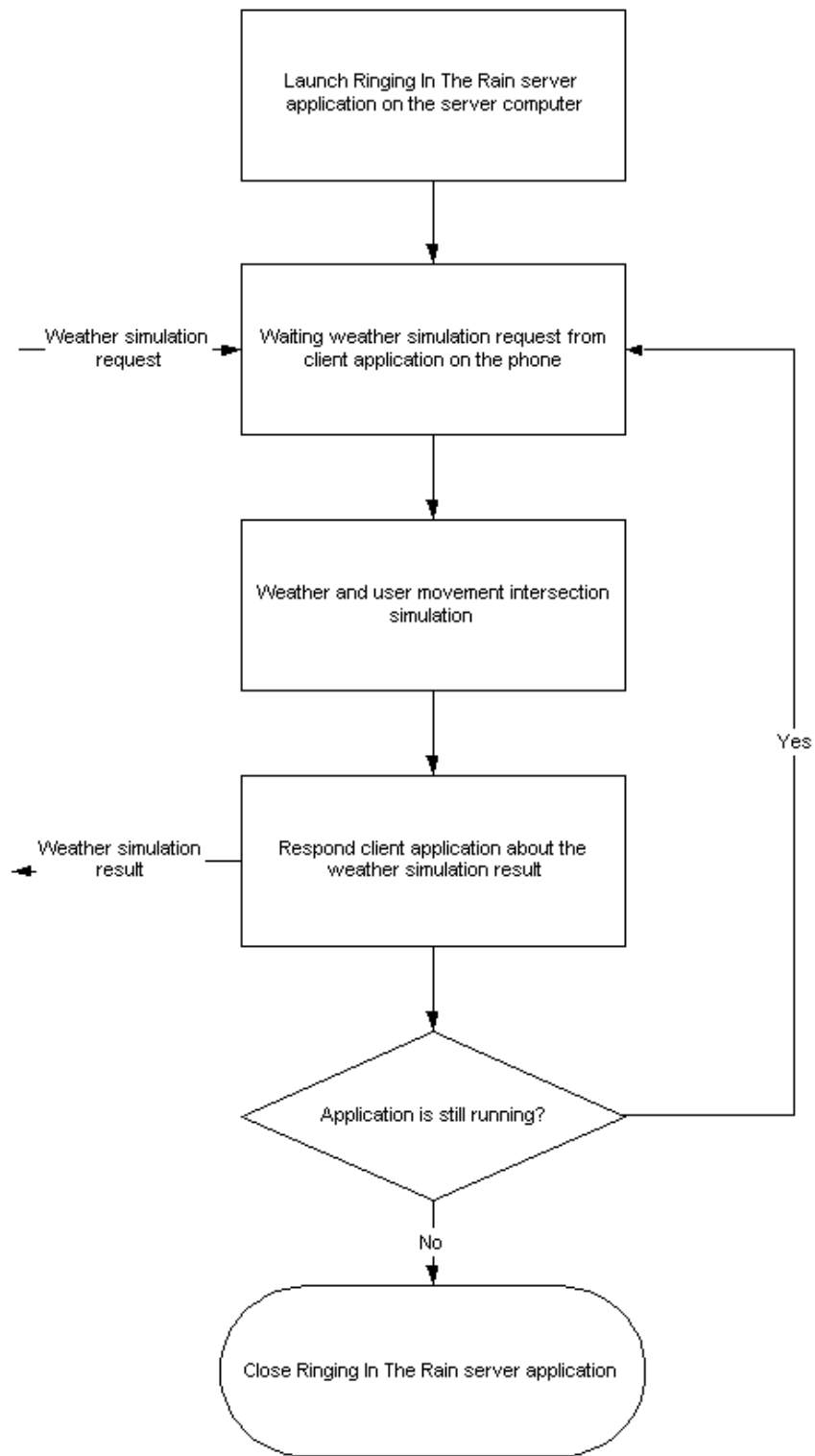


Figure 10. The main control flow of user goal maintaining agent

3.2 User Interface and Usage Scenarios

In this section, I will describe the user interface of the Ringing In The Rain system, used in the client application running on the cell phone. I will also provide some usage scenarios.

3.2.1 User Interface

The design goal of the Ringing In The Rain client application interface is to minimize the interaction between users and applications. Therefore, the default window that appears when users launch the application on the cell phone looks like Figure 11, which simply displays a user's location and the current weather information. This main window can tell users all that they need to know about the weather information. A menu key in the default window will lead users to the main menu of the Ringing In The Rain client application, as shown in Figure 12.



Figure 11. Default information window of Ringing In The Rain client application



Figure 12. Main menu of Ringing In The Rain client application

The first three options in the main menu - Request Calendar, Init All Data, and Dump Route - are assisting functions for the client application. The Request Calendar is used to request the next calendar entry from the personal information management agent to assist in route prediction. Init All Data is for loading pre-determined locations and route information into the application. Dump Route is for synchronizing a user's route

information collected from the cell phone with the server computer. Users will not receive any feedback when they select these options because all the actions triggered by these options happen in the background. In addition, Request Calendar and Dump Route will be triggered automatically if needed, during the communication between the user movement estimation agent and the user goal-maintaining agent.

If users want to know the current storm situation, they can select the Display Radar Image option, which shows the current radar image downloaded from the web, as shown in Figure 13.



Figure 13. Display current radar image.

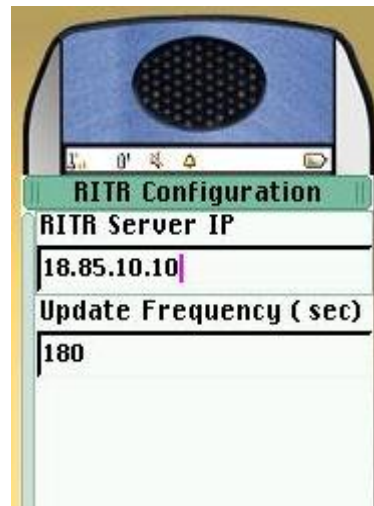


Figure 14. Ringing In The Rain Configuration

Figure 14 shows the configuration window that appears when users select the Configuration option. Users can specify the Ringing In The Rain server IP address here, and can also configure how frequently the client application requests an updated weather prediction.

The Check Service Status option offers users the option to check the current status of the Ringing In The Rain server. If the server is not running, it will not return any status to the client. Otherwise, the server will return one of the following: 1) “System is running”, which means the service is running fine, 2) “Can’t get radar image from internet”, which means that perhaps there is something wrong with the connection between the data collection agents and the weather data sources, and 3) “Storm

movement vector isn't ready yet", which means the data collection agents are still processing the recently downloaded weather data to extract the storm movement information.

3.2.2 Usage Scenarios

The main purpose of Ringing In The Rain is to inform users about the possibility of upcoming rain in order for them to make a better decision about future travel plans. Therefore, the Ringing In The Rain client application displays all necessary information in the default window so users can determine at a glance the risk of encountering rain, as shown in Figure 11. Depending on whether users are in a known location or whether they are currently moving along a known route, slightly different information may display on the main default window. Figures 15 to 20 show information that may be displayed in various different circumstances.



Figure 15. User remains in a known location with rain approaching

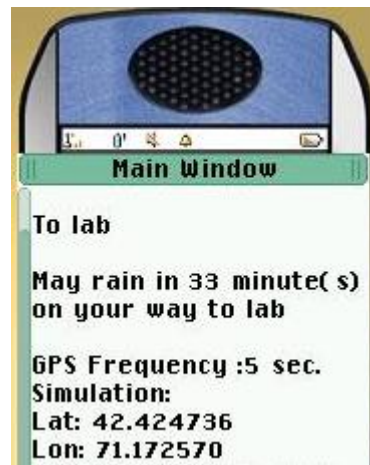


Figure 16. User is in transit along a known route with rain approaching



Figure 17. User stays in an unknown location with rain approaching

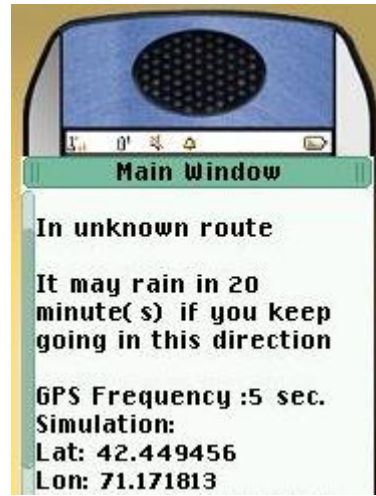


Figure 18. User is in transit along an unknown route with rain approaching

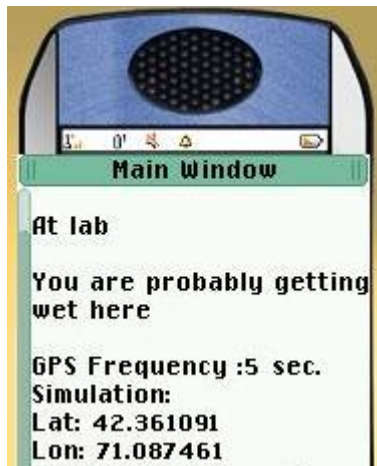


Figure 19. Currently raining in user's location



Figure 20. User will not encounter rain

The information shown on the main window contains two parts, a location and the estimated weather information as mentioned in the previous section. If a user is in a known location or along a known route, the user's current location or the destination of the route s/he is taking will be shown in the location part, as in the cases in Figures 15, 16, 19 and 20. Otherwise, the location message will show that users are along an unknown route, as in Figures 17 and 18. Depending on whether users are moving or not and whether rain is approaching or not, the estimated weather information message changes accordingly. Figures 19 and 20 show two extreme cases, namely, that it is currently

raining in the user's location in Figure 19, and that there will be no rain in the user's location for the next 60 minutes in Figure 20. The other Figures show different cases, including both moving and not moving along known and unknown routes.

3.3 User Movement Estimation Algorithms

This section explains two user movement estimation algorithms, the Route Learning and Detection algorithm and the Linear Approximation algorithm. These algorithms are used in the user movement estimation agent running on the cell phone. The details of these two algorithms will be described in the following two subsections.

3.3.1 Route Learning and Detection Algorithm

The Route Learning and Detection algorithm is the main algorithm used in the user movement estimation agent. This algorithm is inherited from the Will You Help Me project [3]. It introduces a data structure named Intermediate points (also called iPoints). iPoints are interpolated positions between two locations labeled with latitude and longitude data. A route is formed with two labeled locations, one for the start and one for the end, and a set of iPoints in between them. A new route template is created when the Route Learning and Detection algorithm learns a new route. iPoints are collected every 50 meters and include latitude and longitude data, as well as the time spent in transit between them. The iPoints fill in the route template to build up the route while users move between the start and end locations. Route detection is achieved by matching already learned routes with a user's current latitudinal and longitudinal values from GPS on the cell phone. Once any iPoint is found near the current location, the route that contains that iPoint will be used for route detection later on. When users travel along the detected routes, their moving directions can be determined too. The time spent in transition from one iPoint to the next can be used to estimate a user's total travel time. Further details about route learning and detection can be found in Chung's master thesis [3]. In Ringing In The Rain, if users are traveling along a known route, meaning that their locations are close to existing iPoints, the learned route information can be used to calculate their destinations, future locations along the route, and travel time. This information can also be used for weather prediction later on. For example, suppose the

user movement estimation agent finds that the user is currently near iPoint 53 in route 1139237258602 and he is heading toward the direction of the starting location of this route. From this data, the user movement estimation agent can estimate that the user is heading home (using route information and direction) and the time it will take for the user to arrive home is approximately 49 minutes (using route and iPoint information). This is how the Route Learning and Detection algorithm helps to estimate a user's movements.

3.3.2 Linear Approximation

If users are not traveling along any known route, a linear approximation approach can be used to estimate their movement. A set of information, included starting latitude and longitude, ending latitude and longitude, and travel time between the start and end locations, is recorded to estimate movement. When new GPS data arrives, the previous ending point latitude and longitude becomes the new starting point latitude and longitude and the newly obtained latitude and longitude become the new ending point latitude and longitude. The time between these two pieces of GPS data is considered as the time spent in transit between these two locations. Therefore, we can use the following formula to estimate users' movement in this case.

$$(Lat_{est}, Long_{est}) = (Lat_{end} + (\frac{Lat_{end} - Lat_{start}}{Time_{spent}}) \times Time_{est}, Long_{end} + (\frac{Long_{end} - Long_{start}}{Time_{spent}}) \times Time_{est})$$

Formula 1. User movement estimation formula in traveling along an unknown route situation

3.4 Storm Movement Estimation Algorithms

The Ringing In The Rain weather prediction process, to be explained in detail in a later section of this chapter, determines whether storms and users will intersect. It does so by mapping a user's latitude/longitude onto a bitmap image that contains a storm's territories., and then it checks whether they overlap. Therefore, the goal of the storm movement estimation algorithms is to convert downloaded weather data to a movement vector format used in the system, $(X^{pixel/minute}, Y^{pixel/minute})$, so the simulation process can shift the storms boundaries accordingly.

Although only two data collection agents are used in the Ringing In The Rain system, several different storm movement estimation algorithms are used. Details of these algorithms will be explained in the following subsections.

3.4.1 Storm Tracker Information Extraction Algorithm

As shown in Figure 8, storm tracker information is sometimes available, provided by weather data websites as text data in a table of html files. The storm tracker data usually contains several fields. The most useful data for Ringing In The Rain are the last two fields in the table shown in Figure 8, namely speed and direction (from). Therefore, once the data collection agent downloads the data from the storm tracker information, that agent will extract the storm speed and direction information from the downloaded data.

Different storms' directions of movement can vary as shown in Figure 8 - one is from NNW, one is from N, and two are from NW. If there is a direction of movement that most storms have in common, it will be selected as the whole storm system's direction of movement. If there are multiple directions with the same number of storms, the first one in the stored data structure will be selected in the current system implementation, even though the Max value in the table, which represents the storm strength, may be used for better determination. Once the direction is determined, the movement angle of each storm within the selected direction group is averaged to get a single representative movement direction. The fastest storm speed will be used to calculate the storm movement vector. The reason that the fastest speed is used is because the system tries to do the estimation aggressively to ensure that users will have the least chance of getting wet. Therefore, after extracting the representative angle and the speed, it is possible to compute the storm movement vector. The storm movement vector is calculated with this formula:

$$(X, Y) = ((\sin(360 - \text{angle}) \times \text{speed})/60, (\cos(360 - \text{angle}) \times \text{speed})/60)$$

Taking the data in Figure 8 as an example, the selected storm direction is NW because the most storms, two in this example, come from this direction. The

representative angle is $(313+315)/2 = 314$. The fastest storm speed in this group is 32 knots. A knot is equal to approximately one pixel in the current system after the conversion, according to the scale on the radar image. So the storm movement vector in this example is $((32 \times \sin(360 - 314))/60, (32 \times \cos(360 - 314))/60) = (0.31, 0.43)$.

3.4.2 Image Processing Based Algorithm

Two main kinds of image processing-based storm movement estimation algorithms are used in the Ringing In The Rain system. Before digging into the details of these algorithms, first I will explain what kinds of information can be extracted from a typical radar image such as Figure 21.

The left hand side of the image in Figure 21 shows the time that this radar image was generated and the storm strength legend. The territory (area of pixels with color in the color legend), location, and strength of each storm can be found easily on the right hand side of the image. A user's location, in latitude/longitude format, can be converted into a pixel coordinate on the image. The pixel color of that coordinate can be used to determine whether it is raining or not in the user's location. In addition, by comparing a sequence of these radar images, the movement of storms may be observed. This is why several image processing-based storm movement estimation algorithms are created, and also the basis of the weather prediction process, which will be described later.

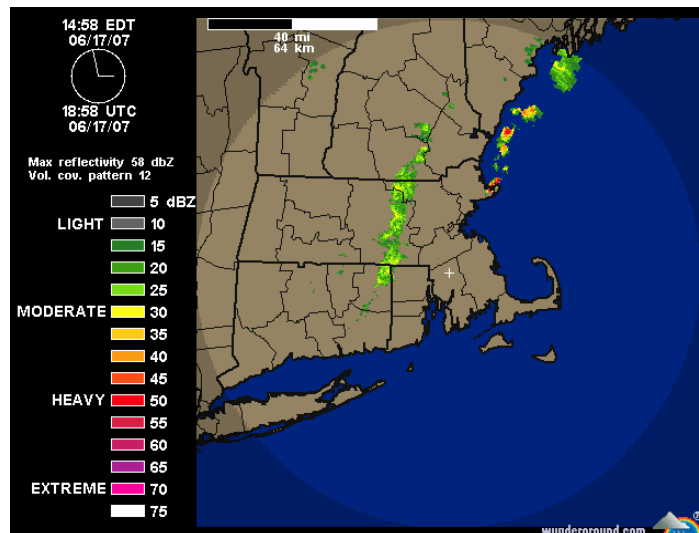


Figure 21. Typical radar image from weather data source

3.4.2.1 Radius Scan Method

The radius scan algorithm is used in the data collection agent that monitors available radar images. The main idea of the radius scan algorithm is to check all directions, i.e. 360 degrees, in a specific location to see whether any storm boundary is approaching. Thus, a sequence of radar images is needed for this algorithm in order to examine whether a storm's boundaries are coming or leaving at different angles. An observation location is needed for this algorithm. The user's current location is the best choice for the observation, but instead, a frequently visited location is used in the real implementation because this eliminates the need for communication between the client and server for obtaining the user's current location while this algorithm is running. Therefore, the latitude/longitude of a user's frequently visited location is used as the reference point from which to conduct a radius scan.

In the radius scan algorithm, a ray tracing [28] like method is used. From the observation location, whose latitude/longitude is mapped onto a pixel in the radar image, the algorithm produces a ray at each angle. Each ray stops when it meets any storm boundary, as shown in Figure 22 and 23. From where the ray stopped, the distance between the storm's boundaries and the observation location can be computed for every angle. Through the iteration of the same process, the movement vector can be calculated accordingly. More details about that iterative process can be found in the pseudo code in Table 1.

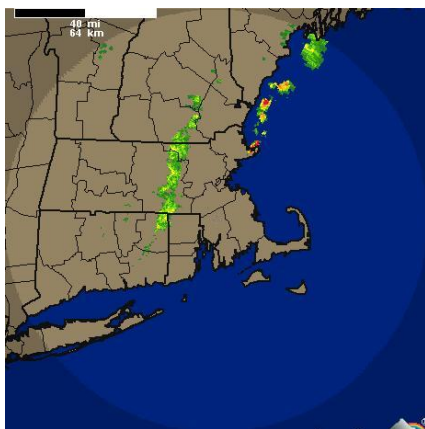


Figure 22. Original radar image

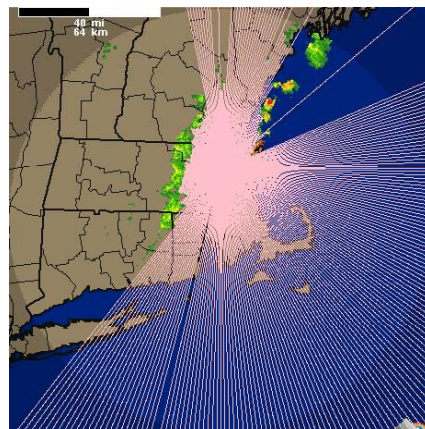


Figure 23. Ray traced radar image

1. Download radar image sequence from Internet;
2. Check any cloud in the latest image of downloaded radar image sequence;
- 3-1, If no rain, Return no rain.
- 3-2, Else if contains rain, Do following
 - {
 - 4. For all images in the sequence
 - {
 - 5. For all angle (1~360)
 - {
 - 6-1, If first time see rain in this angle,
 - 7. Save the distance between rain and observation location for this angle
 - 6-2, Else if not first time see rain in this angle,
 - {
 - 8. Compare the new distance with the stored one to tell if the rain is coming, leaving, or staying still.
 - 9-1. If a moving direction (coming or leaving) is stored, compare the new one with stored one
 - 10-1. If they are opposite,
 - Replace the stored direction with new value.
 - 10-2. Else if they are the same,
 - Increase the counter that stores the consecutive same direction movement
 - 10-3. Else
 - Store the moving direction.
 - 11. Store the new distance and calculate the moving speed for this angle.
 - }
 - }
 - }
12. Sort all angles with rain in sight according to its time of arrival, i.e. distance/speed.
13. If the time of arrival is the same, the angle with higher value of consecutive same direction movement counter is picked.
14. Compute the movement vector according to the angle and the speed associated with it.

Table 1. Pseudo code of Radius Scan Method

3.4.2.2 Center Of Gravity Method

Six slightly different algorithms are used in this Center of Gravity Method, which is also an image processing-based storm movement algorithm. The images needed for these algorithms are from the same downloaded image sequences used in the radius scan algorithm; therefore, no extra data collection actions are required.

All these image processing algorithm variations are based on extracting a representative point, named the center of gravity (CoG). The main difference between these variations is whether to consider only a single storm, usually the largest storm, or the whole storm system, including all storms in the radar image. The second difference is which method to use to determine the CoG; non-weighted, weighted, and overlapped. So, these variations are 1) non-weighted CoG of the largest storm in the whole storm system, 2) weighted CoG of the largest storm in the whole storm system, 3) CoG of the overlapped largest storms in the whole storm system, 4) non-weighted CoG of the whole storm system, 5) weighted CoG of the whole storm system, and 6) CoG of the overlapped whole storm systems. Usually, only the first three variations, those using the largest storm to compute the CoG, are used, unless the largest storm changes too much in consecutive radar image due to either A) The largest storm in Image_i is no longer the largest storm in Image_{i+1}; it either fragmented or another larger storm appeared, or B) The largest storm in Image_i is merged with nearby storms. Both of these situations could change the CoG of the largest storm dramatically, which should be ignored. The current solution to this problem is to make up to two attempts to look for the second largest storm instead; if no storm is suitable as the largest storm, then the whole storm system, or variations 4, 5, and 6, will be used to compute the CoG.

The largest storm-based CoG methods are used prior to the whole storm system-based method unless the data collection agent fails to determine the largest storm for computation three times due to the aforementioned situations A) and B). They are used in this order because the whole storm system-based method cannot handle storms at the radar image boundaries very well. For example, suppose a huge storm enters the radar image from the left and moves to the right while some small storms move to the right and disappear from the image. In this case, the CoG computed based on the whole storm system-based method will be dragged to the left instead of showing movement to the

right. This example shows a common case in which storms develop in areas outside the area covered by the radar image, which is a typical storm pattern in a Massachusetts summer.

Formulas 2 and 3 below show how the non-weighted and weighted CoGs are computed respectively, while (X_{CoG}, Y_{CoG}) is the CoG's pixel coordinate in the radar image, (X_i, Y_i) is the coordinate of each pixel belonging to a storm, and W_i is the weight of that pixel, which also represents the strength of the storm in that pixel. The overlap here refers to overlapping the newer image, $Image_{i+1}$, to the previous one, $Image_i$, which should shift the CoG a bit toward the newer image's CoG if the two images are different. However, because the storm strength of overlapped images is hard to determine, only non-weighted CoG is computed in this case.

$$(X_{GC}, Y_{GC}) = \left(\frac{\sum_{i=1}^n X_i}{n}, \frac{\sum_{i=1}^n Y_i}{n} \right)$$

Formula 2. Formula to compute non-weighted CoG

$$(X_{GC}, Y_{GC}) = \left(\frac{\sum_{i=1}^n W_i \times X_i}{\sum_{i=1}^n W_i}, \frac{\sum_{i=1}^n W_i \times Y_i}{\sum_{i=1}^n W_i} \right)$$

Formula 3. Formula to compute weighted CoG

Once the CoG of each radar image is computed - whether it is done using the largest storm or the whole system - the movement vector can be calculated by differentiating the CoG of each image. However, each algorithm variation - non-weighted, weighted, and overlapped - still generates one movement vector, which means the system needs to select one of the three movement vector candidates as the representative movement vector. Therefore, a comparison process is introduced for the movement vector selection. By differentiating the CoG of radar image sequences, $Image_1$ to $Image_n$, with each variation, a correspondent movement vector sequence, MV_1 to MV_{n-1} , is generated. The movement vector sequence with the most consecutive movement vectors in the same direction is selected. If more than one movement vector sequence has the

same amount of consecutive movement vectors in the same direction, the one with the fastest speed, calculated by averaging the movement vectors, is selected. After the movement vector sequence is selected, the representative movement vector is computed by averaging the movement vectors inside that sequence. The combination of pseudo code in Tables 2 and 3 represents this process.

1. Download radar image sequence from Internet;
2. Check any cloud in the latest image of downloaded radar image sequence;
- 3-1, If no rain, Return no rain.
- 3-2, Else if contains rain, Do following
 - {
 - 4. For all images in the sequence
 - {
 - 5. Find the largest storms in Image_i and the largest overlapped storm in Image_{i+1}
 - 6-1. If their sizes do not change too much
 - 7. Compute the CoGs of these two storms with non-weighted, weighted, and overlapped variations
 - 8. Differentiate these two CoGs to get movement vector, and save the movement vector
 - 6-2. Else if the size change too much
 - 9. Use the next largest storm and go back to 5.
 - 10. If failed to find the largest storm twice, ignore this image.
 - }
 - }
11. Select the representative movement vector sequence from candidate movement vector sequences.
- 12-1. If the representative movement vector sequence can be found
 - 13. Compute the representative movement vector
- 12-2. Else if it cannot be found
 - 14. Run the center of gravity method for the whole storm system

Table 2. Pseudo code of Center of Gravity computation method for the largest storm

1. For all images in the sequence
 - {
 - 2. Compute the CoGs of these two storms with non-weighted, weighted, and overlapped variations
 - 3. Differentiate these two CoGs to get movement vector, and save the movement vector
 - }
4. Select the representative movement vector sequence from candidate movement vector sequences.
- 5-1. If the representative movement vector sequence can be found
 6. Compute the representative movement vector
- 5-2. Else if it cannot be found
 7. Return no movement vector can be found

Table 3. Pseudo code of Center of Gravity computation method for the whole storm system

3.5 Weather Prediction Algorithms

Once the system is able to compute the storm movement vector using either the storm tracker information or the image processing-based methods, it is possible to do weather prediction, i.e. to predict whether the storms and users will intersect according to their future movement estimations. This can be done by comparing the storm movement vector with the user movement estimation sent by the user movement estimation agent running on the cell phone.



Figure 24. It's not raining yet at the user's current location.



Figure 25. It's raining at the user's current location.

The basic concept of the weather prediction algorithm is to map the user's location (latitude/longitude), to the corresponding pixel in the radar image, and check if it is under a storm's territory, as shown in Figures 24 and 25. Thus, the system can conduct the weather prediction with the following steps. First of all, the system estimates the user's future location at a given time according to his/her movement estimation provided by the user movement estimation agent. Second, the system estimates where the storm will be by utilizing the storm movement vector. Finally, the system maps the estimated user location to the corresponding pixel in the radar image, and then checks whether it is within any storm's territory. As mentioned in the previous section, two different user movement estimation algorithms – the route learning and detection algorithm, and linear approximation – are used in the user movement estimation agent. When the user goal-maintaining agent runs the weather prediction process, it deals slightly differently with user movement estimation results that come from different algorithms. Tables 4 and 5, and Figures 26 and 27 show how the weather prediction process is run when the user movement estimation results are computed by different algorithms.

- 1-1. If user is going toward to the end
 2. For all iPoints from current position to the end iPoint
 3. Get the delta time from current position to iPoint
 4. Shift storm's boundary according to the delta time and storm movement vector
 5. Convert the iPoint latitude/longitude to pixel in radar image.
 6. Check if that pixel intersects with any storm
 7. If yes, return the delta time as the time of storm intersection
- 1-2. Else if user is going toward to the front
 8. For all iPoints from current position to the front iPoint
 9. Get the delta time from current position to iPoint
 10. Shift storm's boundary according to the delta time and storm movement vector
 11. Convert the iPoint latitude/longitude to pixel in radar image.
 12. Check if that pixel intersects with any storm
 13. If yes, return the delta time as the time of storm intersection
- 1-3. Else if user is staying at this position
 14. Shift storms boundary from 1 to 60 minutes according to the storm movement vector.
 15. Convert the current position's latitude/longitude to pixel in radar image.
 16. Check if that pixel intersects with any storm
 17. If yes, return the delta time as the time of storm intersection
18. If user will not intersect with rain according to simulation, return will not intersect with rain.

Table 4. Pseudo code weather prediction process when route learning and detection result is used.

- 1-1. If user is moving
 2. Shift storms boundary from 1 to 60 minutes according to the storm movement vector, also shift user's position from 1 to 60 minutes.
 3. Convert the shifted user position's latitude/longitude to pixel in radar image.
 4. Check if that pixel intersects with any storm
 5. If yes, return the time as the time of storm intersection
- 1-2. Else if user is staying at this position
 6. Shift storm's boundary from 1 to 60 minutes according to the storm movement vector.
 7. Convert the current position's latitude/longitude to pixel in radar image.
 8. Check if that pixel intersects with any storm
 9. If yes, return the delta time as the time of storm intersection
10. If user will not intersect with rain according to simulation, return will not intersect with rain.

Table 5. Pseudo code weather prediction process when linear approximation result is used.

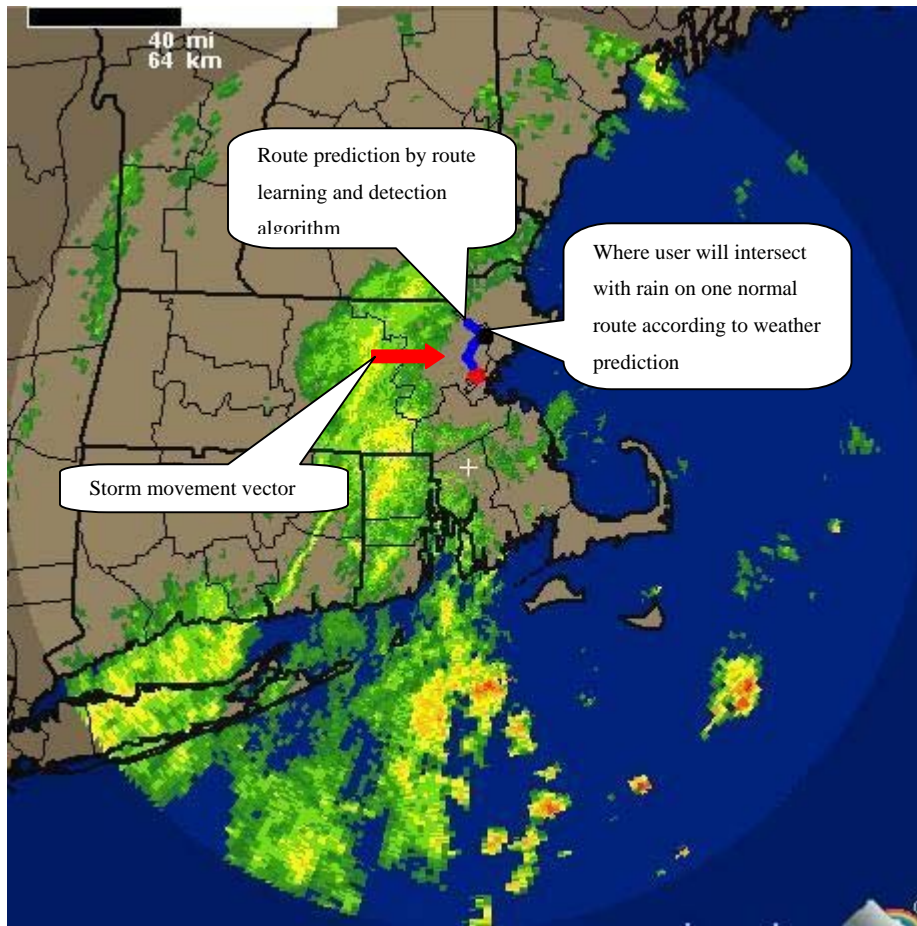


Figure 26. Illustration of the weather prediction based on route learning and detection algorithm

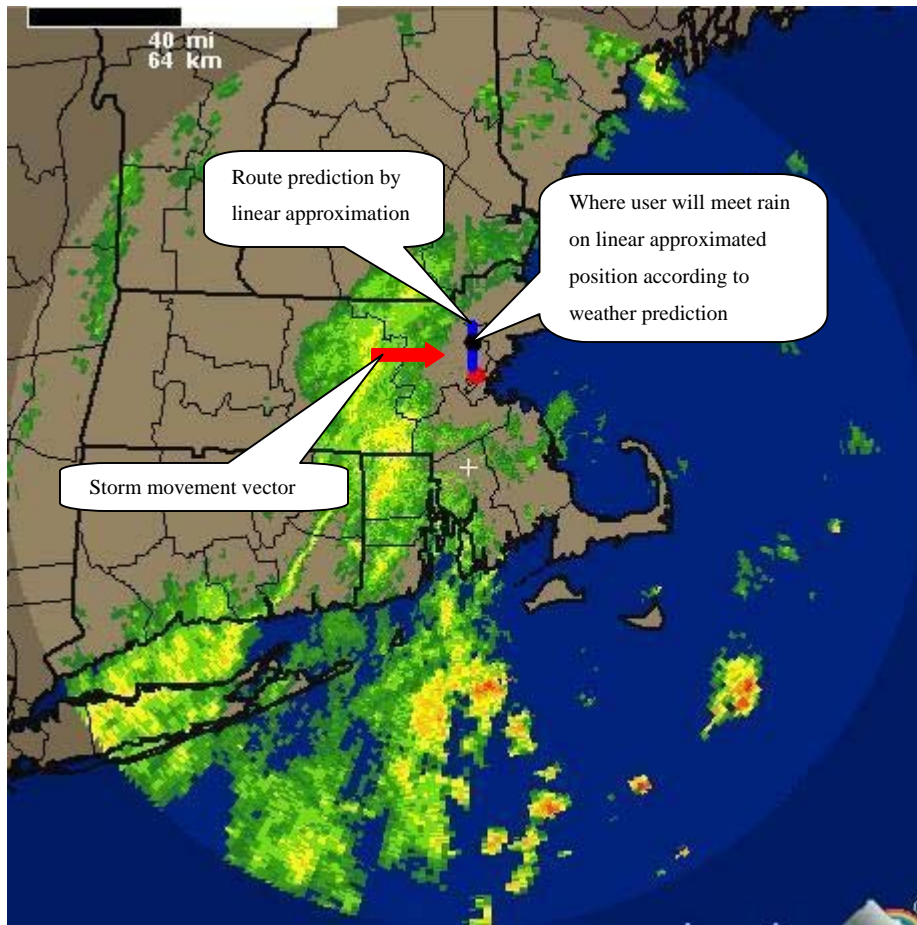


Figure 27. Illustration of the weather prediction based on linear approximation method

Though different storm movement estimation algorithms are used to generate storm movement vectors, they have different priorities when being used for weather prediction. The storm movement vector generated by the storm tracker information has highest priority since it is computed based on highly precise storm estimation data provided by a weather source directly; unfortunately, they are not always available. Thus, when this kind of movement vector is available, it will be used for weather prediction first before any other methods. If there is no movement vector generated by the storm tracker information method, then movement vectors generated by the radius scan method and the center of gravity method's variations will both be used for the simulation. If either one gets "user will meet rain" as the simulation result, the "user will meet rain"

message will be returned to users, along with the time that they will encounter the rain. If both methods return a simulation result of “user will meet rain”, the one with shorter estimated time that users will encounter the rain will be returned.

To sum up the Ringing In The Rain system architecture: A client/server architecture that hosts the proposed agents is used to build the system. The user movement estimation agent is running on the cell phone within the client part of the application to determine the user’s current location and estimate future locations. This agent also communicates with the personal information management agent for calendar entries, which may help to select the correct route for estimation. The other agents, namely the data collection agents, the personal information management agent, and the user goal-maintaining agent, run on the server computer within the server part of the application. Data collection agents keep monitoring new weather information in different formats and computing storm movement vectors using various algorithms. The personal information management agent extracts time and location information from a user’s calendar application and provides this data to the user movement estimation agent. The user goal-maintaining agent conducts weather prediction using the storm movement vector data that it receives from the data collection agents. It also uses the user movement estimation data, from the user movement estimation agent, when it receives periodic requests from client part of the application. Then the simulation result is sent back to the client part of the application to inform users whether or not they will encounter the rain, and if so, at what time this might occur.

Chapter 4

Evaluation

In this chapter, I will analyze the results based on the collected weather data and discuss the user experience in the first section. The second section will focus on some general problems in the system and some special conditions that specific algorithms cannot handle well, as found during field tests. The last section will propose some possible enhancements to the system.

4.1 Results Analysis

4.1.1 Collected Data Analysis

This section will focus on analyzing the performance of the weather prediction process, i.e. how precise the prediction is. Though both the user movement estimation and the storm movement estimation are used in the weather prediction process, only the storm movement estimation results are considered in the analysis of the weather prediction performance. This is because a single pixel on the radar image represents a certain size of physical space; thus, any errors due to the user movement estimation algorithm are not very significant after the latitude/longitude to pixel conversion takes place. In addition, Chung's master thesis [3] shows that the user movement estimation results generated by the route learning and detection algorithm are highly reliable. Therefore, the user movement estimation results will not impact the weather prediction performance too much.. The analysis includes two parts: 1) Storm movement direction estimation analysis and 2) Storm movement estimation performance analysis.

4.1.1.1 Storm Movement Direction Estimation Analysis

The storm movement direction estimation analysis was done to compare the storm movement direction generated by the radius scan method and the center of gravity method with results generated by the storm information tracker method, extracted from professional meteorological data sources considered to be highly reliable. This analysis will show the storm movement direction estimation reliability of the image processing-based storm movement estimation algorithms used in the current system compared with

the storm tracker information data. This is done by considering the movement direction angle of the storm movement vector. The storm movement vector used in the system is in the format $(X^{pixel/minute}, Y^{pixel/minute})$ format, which can be converted to (Speed, Angle) format. Therefore, the angle of the storm movement direction can be extracted for comparison. Only the angle is compared but not the speed because the angle is the main factor in determining whether the storm will hit the user or not; the speed only determines the time that the user will encounter the storm, if applicable.

In the weather data log collected from June 20, 2007 to July 25, 2007, 5,921 radar images were downloaded, and 4,174 entries show that there were some areas in which it was raining; the radar image is considered to be indicating rain when more than 0.1% of the radar image contains rain dots. 2,425 entries out of 4,174 contain storm tracker information. Therefore, the availability of storm tracker information in this period is $(2425/4174) \approx 58.1\%$.

To analyze the direction estimation reliability distribution of the movement estimation algorithms other than the storm tracker information method, only those log entries containing estimation results from all algorithms are used for computation. Therefore, 1,388 log entries were used to compute the reliability. Figure 28 shows the estimation reliability distribution. The x-axis represents the reliability and the y-axis represents the error range from the reference angle, i.e. the angle from the storm tracker information. For example, around 70% of the estimations fall into the range of plus/minus 20 degrees from the reference angle.

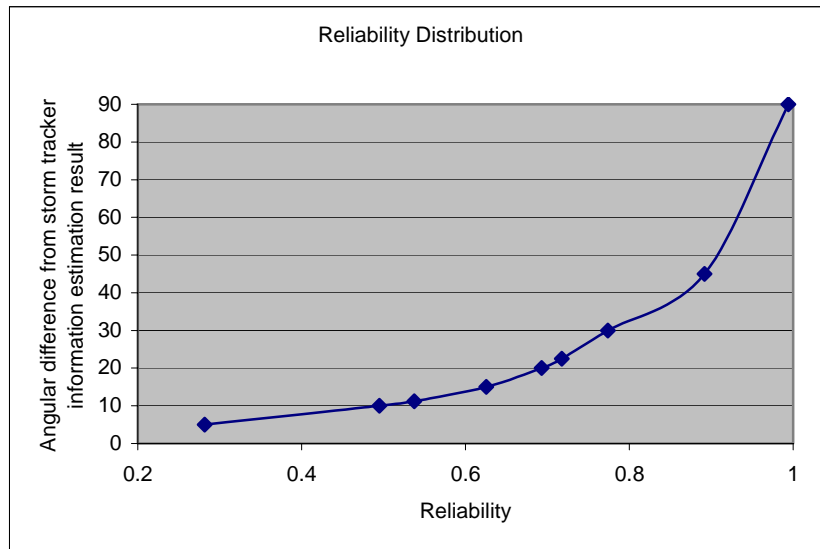


Figure 28. Reliability of the estimation result

Since the weather source used in the Ringing In The Rain system divides the storm movement direction into 16 directions, i.e. 22.5 degrees for each direction, the estimation results generated by the radius scan and the center of gravity method are considered correct if they fall into the same direction as the estimation result generated by the storm tracker information. In this case, 746 entries out of 1,388 fall in the same direction as the result given by the storm tracker information (plus/minus 11.25 degrees). In other words, these other algorithms have a reliability of $(746/1388) \approx 53.7\%$ in direction estimation.

The storm tracker information is usually available in severe storm weather conditions, though sometimes it is not available. According to the analysis results in the previous section, it can be assumed that the image processing-based storm movement estimation algorithms provide around 53.7% accuracy in storm movement direction estimation under severe weather conditions. If that accuracy can also apply to mild weather situations, which do not have storm tracker information, the overall weather prediction may be computed by the following formula:

$$\text{(System Estimation Reliability)} = \text{(Availability of storm tracker information)} + \text{(Reliability of other algorithm times when storm tracker information is not available)}$$

Using the above analyzed data and the assumption that the estimation accuracy in severe weather conditions can apply to mild ones, we can conclude that the overall system estimation reliability based on data collected between June 20, 2007 and July 25, 2007 is $58.1\% + ((100\% - 58.1\%) \times 53.7\%) \approx 80.6\%$. This figure offers users a significant result with which to make their transportation decisions.

4.1.1.2 Storm Movement Estimation Performance Analysis

The storm movement estimation performance analysis tries to compare the estimated storm position with the collected data. The comparison is done with the following steps: 1) Set up a fixed observation location, 2) Conduct the storm movement estimation with collected radar images, 3) Predict the weather based on the storm movement estimation, and 4) Compare the weather prediction results with results generated based on collected radar images. In my example, the Media Lab was chosen as the observation location. From the data collected between 20:00 and 20:30 on June 21, 2007, the weather prediction shows that it is going to rain within one hour of 20:30, based on the storm movement estimation results. Therefore, the radar images between 20:30 and 21:30 are used to check whether it really rained or not at the observation location of the Media Lab.

The prediction result is classified into one of the following categories: 1) The prediction is either for rain or for no rain; if the prediction matched the real result, this estimation is considered as “matched.” 2) If the prediction says it is going to rain but it did not rain according to the actual result, this estimation is considered to be a “false alarm.” 3) In the case that the prediction says it is not going to rain but the real result shows that it did rain, this estimation is marked as “missed.” 4) The last category is named “miscellaneous” and includes cases not mentioned in the above classifications. These usually arise from some exceptions that occurred in the weather prediction process, which returned no prediction results. The exceptions may be caused by defective radar images or irregular storm movement estimation results.

After conducting this analysis with the data collected between June 20 and July 25, 2007, and after ignoring the clear sky cases, i.e. no rain shown at all in the radar image, a total of 4,517 estimations were conducted. According to the aforementioned

classification rule, 3,971 predictions out of 4,517 matched the results, which is around 87.9%. 503 predictions out of 4,517 are false alarms, which is around 11.1%. Missed and miscellaneous predictions occurred at rates of 0.46% and 0.48%, respectively.

Analysis results from the aforementioned two analysis processes show a certain reliability of the storm movement estimation results and weather prediction results. To place these results in context, the summer precipitation pattern in the Boston area is dominated by rapidly moving groups of thunderstorms, as well as isolated or widely scattered rain cells. Prediction is therefore more challenging than during the rest of the year when large rain storms covering much of New England dominate. Under these difficult conditions, nearly 90% of the weather predictions turned out to be correct. The vast majority of the errors were false alarms, i.e. when predicted rain did not occur. This error bias probably suits the target users population; a cyclist might miss some riding opportunities due to false alarms, but this is much less distressing than getting soaked by an unpredicted shower!

4.1.2 User Experience

After some computer simulations and refinement of the user interface, some user field trials of Ringing In The Rain were conducted during June and July within the greater Boston area. One of the few trial users commutes between his home and office with a journey time of more than one hour. He walks and bicycles for some portion of the journey and takes public transportation for the rest. During an interview about his user experience, he stated that he was able to plan his trip according to the information displayed on the Ringing In The Rain client application and the reliability of the weather prediction results it offered.. Also, he used the feature to display the radar image quite often to get an overview of the current weather situation in addition to the weather prediction. In this example, Ringing In The Rain is meeting its goal, the user was able to make better-informed decisions about which routes to take or when to depart in order to avoid encounters with nasty weather conditions.

4.2 Problems Found For The Current System

4.2.1 General Problems

Though the analysis of the collected weather data and the movement estimation log shows a certain reliability of the overall estimation results, there are still some problems that the current system has a hard time dealing with. These may explain why the weather prediction performance is not as good as expected. The first problem is that the radar image is not very continuous. In severe storm situations, the radar image updates every 5 or 6 minutes while in mild weather, it updates only every 10 minutes. However, during this interval of waiting for a new image update, the weather may change dramatically., Either suddenly a new storm may appear close to some users or some nearby storms may disappear. These suddenly appearing and disappearing storms may cause missed or false alarms in the system.

The second general problem is the clutter problem. Clutter is the noise in the weather radar scans, and is also an on/off option in the weather data source for the Ringing In The Rain system. In Figure 29, the clutter option is off, and in Figure 30, the clutter option is on. These Figures illustrate a good case for the pros and cons of the existence of clutter. The pro of clutter is that it can show some areas of light rain surrounding a storm, and thus completing the storm area. However, clutter may also cause some false alarms in areas where there is no rain.

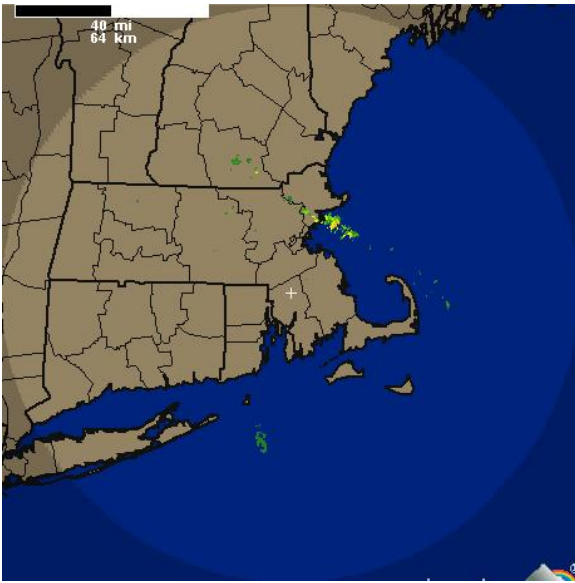


Figure 29. Radar image with clutter option off

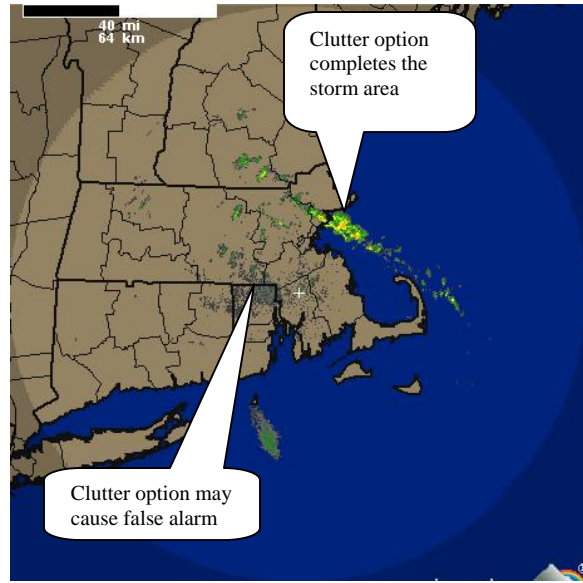


Figure 30. Same radar image as Figure 29 with clutter option on

Figures 31 and 32 show another similar but frequently seen situation: the image in which the clutter option is off shows no rain at all, while the image with the clutter option on shows quite a few areas of rain.

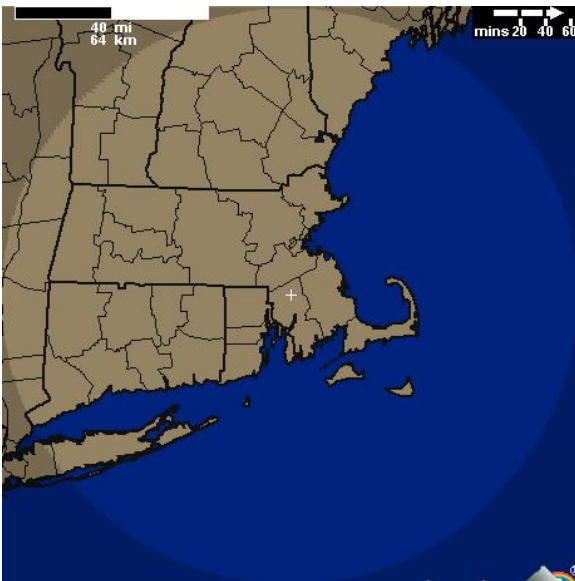


Figure 31. Radar image with clutter option off

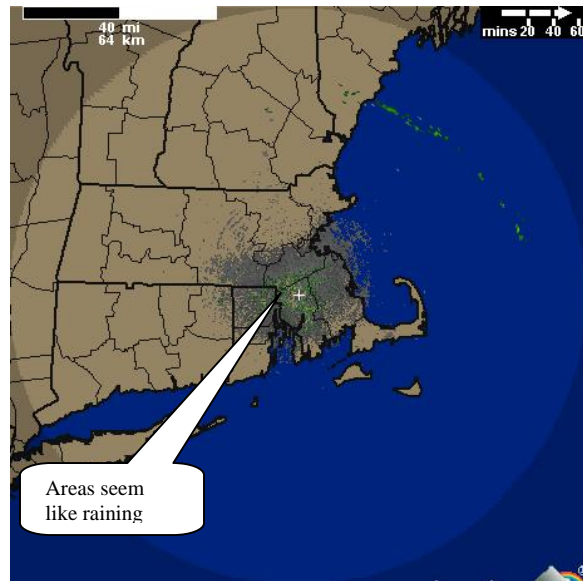


Figure 32. Figure 31 with clutter option on

Since Ringing In The Rain relies on radar images for weather prediction, the incompleteness of a radar image will cause problems. However, the downloaded radar

image is not always valid for use. Figure 33 shows one example of this kind of situation in which the radar image is a partially scanned result. Figure 34 is an example in which the radar station is down for maintenance, i.e. no valid radar image data is available for estimation at all.

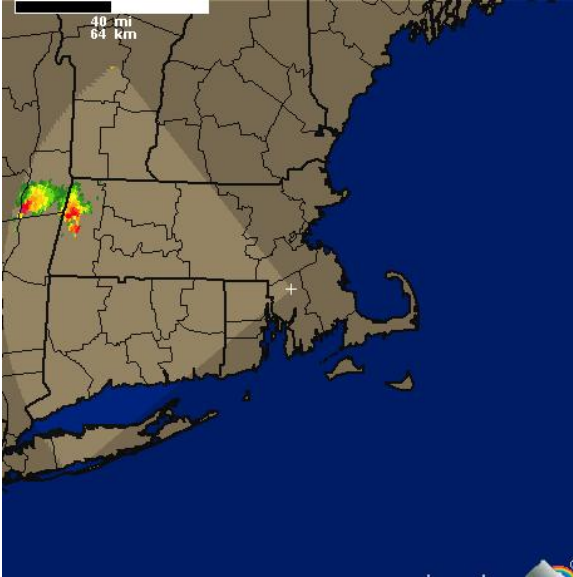


Figure 33. Partially scanned radar image

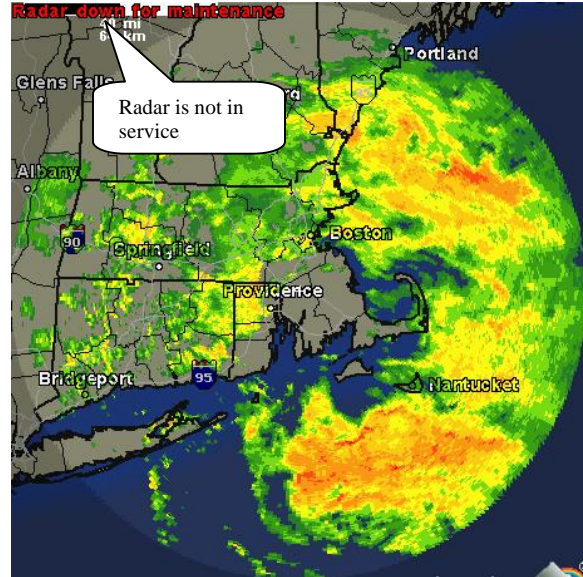


Figure 34. Radar station is down

The last general problem is that users may be located near the boundaries of storms. Since the storm movement vector changes all the time, if users are around any storm boundary, the estimation results may fluctuate between “no rain” and “will meet rain” very rapidly and frequently.

4.2.2 Problems For Storm Tracker Information Method

Though the storm tracker information method may be the most precise algorithm of all the algorithms used in the Ringing In The Rain system, there is one condition that it may not be able to handle well. This condition is when there are multiple storms detected and most of them are moving toward different directions. Figures 35 and 36 depict this kind of situation.

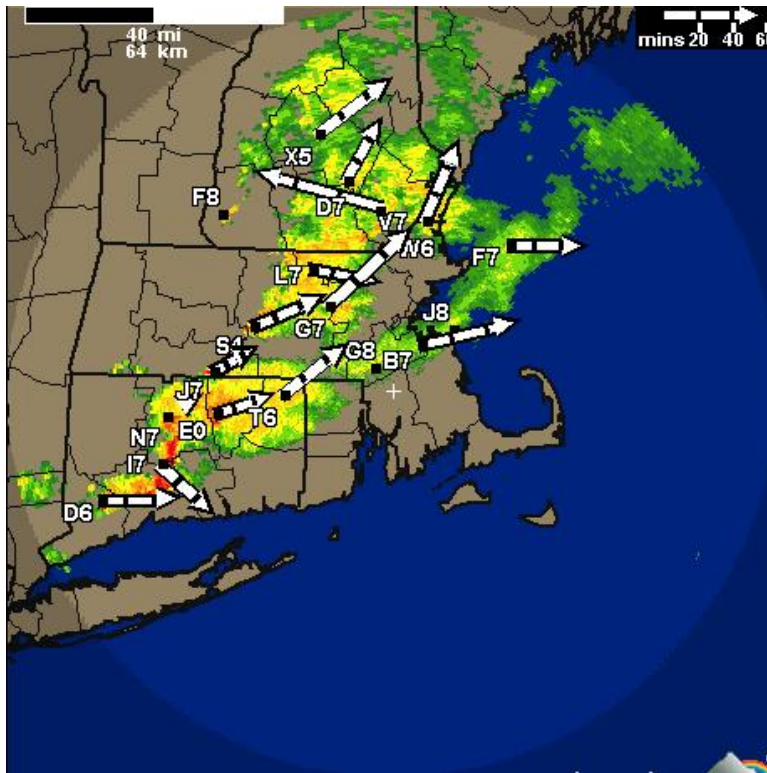


Figure 35. Radar image of a special condition that the storm tracker information method may not handle well

NEXRAD Storm Table

▼ = Tornado Vortex Signature ◆ = Mesocyclone ■ = Hail Storm

ID	Max	Top	VIL	Chance of Severe Hail	Chance of Hail	Max Hail Size	Speed	Direction (from)
I7	54 dBZ	18,000 ft.	20 kg/m ²	0%	0%	0.00 in.	25 knots	NW (312)
E0	54 dBZ	16,000 ft.	18 kg/m ²	0%	0%	0.00 in.	21 knots	WSW(252)
D6	53 dBZ	21,000 ft.	21 kg/m ²	0%	10%	<0.50 in.	27 knots	W (267)
S4	49 dBZ	13,000 ft.	8 kg/m ²	0%	0%	0.00 in.	27 knots	WSW(246)
N7	49 dBZ	13,000 ft.	4 kg/m ²	0%	0%	0.00 in.	15 knots	WSW(238)
J7	48 dBZ	13,000 ft.	7 kg/m ²	0%	0%	0.00 in.	18 knots	WSW(243)
W6	48 dBZ	15,000 ft.	6 kg/m ²	0%	0%	0.00 in.	29 knots	SSW (203)
L7	46 dBZ	5,000 ft.	3 kg/m ²	0%	0%	0.00 in.	24 knots	WNW(282)
J8	46 dBZ	13,000 ft.	2 kg/m ²	0%	0%	0.00 in.	New Cell	
F8	45 dBZ	11,000 ft.	5 kg/m ²	0%	0%	0.00 in.	New Cell	
V7	45 dBZ	13,000 ft.	5 kg/m ²	0%	0%	0.00 in.	43 knots	ESE (106)
D7	44 dBZ	12,000 ft.	5 kg/m ²	0%	0%	0.00 in.	24 knots	SSW (211)
G7	44 dBZ	13,000 ft.	3 kg/m ²	0%	0%	0.00 in.	38 knots	SW (227)
H8	44 dBZ	13,000 ft.	2 kg/m ²	0%	0%	0.00 in.	New Cell	
G8	44 dBZ	12,000 ft.	2 kg/m ²	0%	0%	0.00 in.	New Cell	
U5	43 dBZ	8,000 ft.	2 kg/m ²	0%	0%	0.00 in.	30 knots	S (191)
K8	43 dBZ	11,000 ft.	1 kg/m ²	0%	0%	0.00 in.	New Cell	
L8	42 dBZ	11,000 ft.	1 kg/m ²	0%	0%	0.00 in.	New Cell	
I8	41 dBZ	7,000 ft.	2 kg/m ²	0%	0%	0.00 in.	New Cell	
X5	40 dBZ	13,000 ft.	3 kg/m ²	0%	0%	0.00 in.	31 knots	SW (232)
A8	39 dBZ	15,000 ft.	3 kg/m ²	0%	0%	0.00 in.	23 knots	ESE (111)
T6	39 dBZ	7,000 ft.	2 kg/m ²	0%	0%	0.00 in.	28 knots	SW (231)
E8	39 dBZ	8,000 ft.	1 kg/m ²	0%	0%	0.00 in.	18 knots	SSE (148)
F7	38 dBZ	11,000 ft.	1 kg/m ²	0%	0%	0.00 in.	26 knots	W (271)
C8	38 dBZ	6,000 ft.	1 kg/m ²	0%	0%	0.00 in.	31 knots	SW (235)
B7	36 dBZ	5,000 ft.	1 kg/m ²	0%	0%	0.00 in.	34 knots	WSW(256)
M8	34 dBZ	12,000 ft.	1 kg/m ²	0%	0%	0.00 in.	New Cell	

- More Information About Storm Data -

Figure 36. Storm tracker table of a special condition that the storm tracker information method may not handle well

In the case shown in Figures 35 and 36, even though WSW will be the storm movement direction candidate for further computation according to the current storm tracker information method, it may not be a representative direction because this storm system is very chaotic overall. Therefore, the result may not be good for estimation under these kinds of weather conditions.

4.2.3 Problems in the Radius Scan Method

The biggest problem that the radius scan method faces is easier to explain by illustration. Figures 37 and 38 show a case in which the radius scan method may get an incorrect storm movement vector. The storm is moving in a northeast direction, but because the boundary is wider in the middle part of the storm, this causes the radius scan method to think that the storm is moving closer to the observation location for some period of time. Therefore, this may sometimes result in a false estimated storm movement vector.

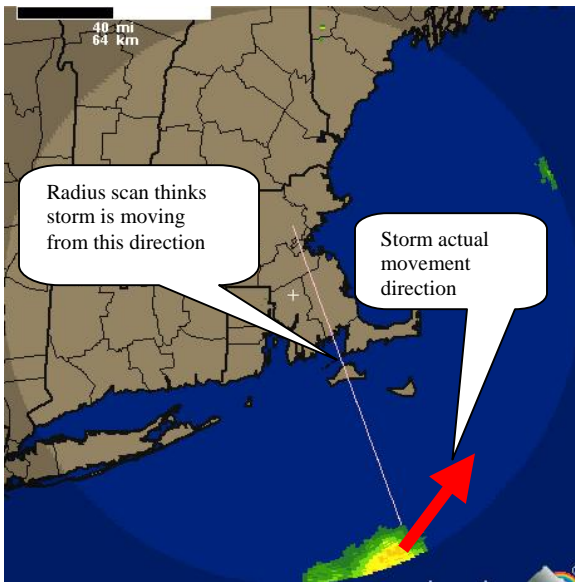


Figure 37. Radar image after processed by radius scan method

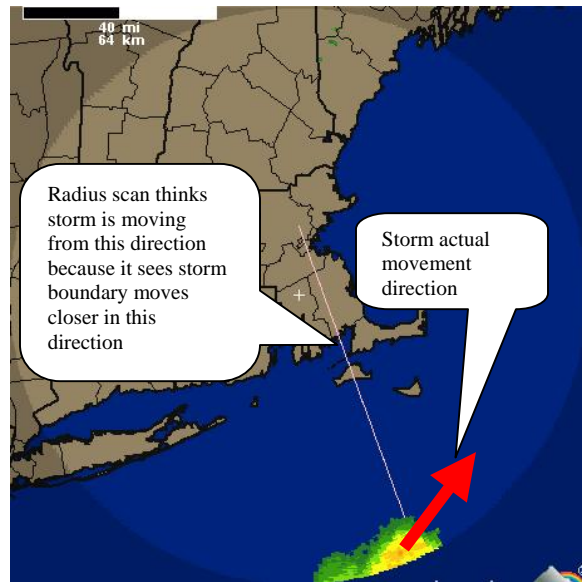


Figure 38. Radar image one frame later than Figure 37 after processed by radius scan method

4.2.4 Problems in the Center of Gravity Method

The main problem in the center of gravity method is the “size” of the storm. The first problematic situation is when the whole storm system is used for calculation as mentioned in the previous chapter, i.e. some large storms come in from the image boundary while some small ones move out. This usually drags the center of gravity toward the direction of the large storms, which causes the movement vector to point to the opposite direction.

Another problematic case occurs under the conditions of using the largest storm

variations. If the largest storm splits into several smaller storms or merges with other nearby storms, the center of gravity of the largest storm may change dramatically and the movement vector may sometimes point in the wrong direction.

One other situation, though it rare, is the case in which the storm is expanding. In this case, the center of gravity may not have a significant change, which may cause the storm movement vector to stay still.

4.3 Possible Enhancements

Considering the aforementioned storm movement estimation problems found during the trials, several enhancements could be considered to improve the accuracy of the whole system. Finding other sources of data sources might be the first thing to try. The interactive radar and weather map from The Weather Channel [23] is an example; it provides weather data in a finer granularity in the application, though the data are embedded in flash program and are not open to the public to use. The National Weather Service [26] also provides weather data in text format. With a proper data interpreter program for this text data, it could be a good potential data source. Another interesting data source might be personal weather stations. More and more families are installing personal weather stations with Internet connectivity, which uploads the newly collected weather data to a specific website. Data provided by these personal weather stations may be good for examining but not using for the estimation since the density of these personal weather stations is still low.

Another enhancement to try is to consider more factors in the currently collected data, such as the size change or the location distribution of storms. For example, in the case where most storms are shrinking, even if one of them is moving toward the user, it may shrink into nothing before reaching the user's current location. On the contrary, if most storms are expanding, the storm movement vector may need to adjust with the storm size expansion rate.

One other more complicated enhancement is to consider other weather factors in the storm movement consideration. For example, the atmospheric pressure and the weather front are some important factors in the movement of storms. If changes in these variables can be considered in the storm movement estimation algorithms, the system

may yield more precise storm movement estimation results for weather prediction.

Chapter 5

Conclusions

5.1 Conclusions

In this thesis, a distributed multi-agent architecture-based mobile service development framework is proposed. This development framework aims for building a just-in-time information delivery system that delivers dynamic environmental information that may impact users' movement decisions. A more specific example is to deliver storm information to bicycle riders if they are heading toward rainy areas. Therefore, a client server-based weather-warning system, called Ringing In The Rain, is built on this development framework as the demonstration.

The client part of this system, which runs on a cell phone, applies the user movement estimation agent in the framework to learn how users usually travel between locations and to build route databases for user movement estimation. The client also utilizes users' calendar entries containing location and time information provided by the personal information management agent, which runs on the server computer. On the server computer, data collection agents built for this system collect weather information from the Internet, including storm tracker information and radar images. They transform the collected weather data into storm movement vectors for further usage. Finally, the user goal-maintaining agent running on the server computer receives weather prediction requests from the client application running on the phone. It simulates both the user movement - from the user movement estimation agent - and storm movement - from various data collection agents - to determine whether there is any predicted encounter between users and storms, and it returns the simulation results to users. With this information, users can make their transportation decisions accordingly.

Weather data in Massachusetts was collected over a certain period of time for examining the storm movement estimation algorithms used in the data collection agent in the system. Some field trials were conducted to check both the usability and usefulness of this system. The trials showed positive results, both in terms of usability and usefulness. The accuracy of the storm movement estimation offers a certain amount of confidence for the weather prediction results. The user interface provides just enough

information for users to know their risk of encountering rain and it enables them to make decisions accordingly. The interaction between users and the system has been trimmed down for less confusion. Therefore, the Ringing In The Rain system acts as a good reference for building mobile services based on the proposed mobile service development framework.

5.2 Future Work

In addition to the algorithm precision enhancements mentioned in the previous chapter, some existing features can be enhanced in the Ringing In The Rain system to complete the service. One possible feature enhancement is to enrich the radar image display. Due to hardware limitations, the current radar image display function can only display JPEG files. However, if cell phones become capable of displaying animated GIF files in the future, this feature can be extended to display animated radar images instead of static ones. Some useful new features could also be added into the system. For example, the system could provide users the estimated time at which the rain is predicted to stop. This would be useful if a user was stuck indoors because of the rain. This enhancement would give users a better idea about how long to stay indoors before it would be dry enough outside to depart. Another possible feature would be to suggest to users the best departure time. If the system can know a user's destination in advance and when they need to arrive there, it may calculate the best departure time for avoiding the rain if there is any storm brewing. Another possible new feature is for the system to suggest to users an alternate route to take to their destination such that they will not get wet on their way there. All of these features may make the services of this system more complete.

Some enhancements can be made to the user movement estimation algorithm too, such as the linear approximation method. The current linear approximation method computes the movement vector based on the current and previous GPS readings only. If the user is traveling down an anfractuous road, the user's direction of movement may fluctuate frequently, which may affect the weather prediction results. However, if the start location of the user's journey is considered in the computing process, the user movement estimation agent may have a better understanding about the general direction

of the user's movements by averaging the start location with the current location. Combining the general direction with the current implementation may give a more precise user movement estimation result when the linear approximation method is used.

In this thesis, Ringing In The Rain demonstrates the usefulness of the proposed distributed multi-agent architecture-based mobile service development framework. Therefore, another dimension of future work is to apply this development framework to other mobile services, i.e. to develop other data collection agents and user goal-maintaining agents for different goals. The goal of Nozue et. al's Just-In-Time Information Delivery System for Passenger Assistance [13] - delivering transportation schedules to users for making travel decisions - may be a good example for the application of this framework. In this case, new data collection agents that collect transportation information need to be created. New user goal-maintaining agents also need to develop accordingly, to complete such goals as preventing users from being late, or ensuring that users spend the least amount of money with a minimum of traveling time. Ringing In The Rain is an example that showcases the potential of the usage of the proposed development framework.

Once more and more user goal-maintaining agents are created to fulfill users' different goals, some conflicts may occur to keep individual goals from being met. For example, suppose a user is traveling on a route from his home to his office. The agent that tries to keep the user dry detects that he will encounter rain if he keeps moving along the same path. However, if an agent that tries to prevent a user from being late is also running in the system, it may suggest taking the original route in order to arrive on time once the user start to detour to an alternate route. However, this conflicts with the goal of the previous agent of keeping the user dry. Therefore, some negotiation or prioritization scheme between the different user goal-maintaining agents needs to be introduced to resolve these kinds of goal conflicts.

In conclusion, the positive evaluation results of the Ringing In The Rain system, the possible system feature enhancements, and other mobile services along the same direction, combine together to show the success of this thesis. It is indeed possible to offer people information that may impact their current or future activities while they are in a state of mobility with limited information access.

Appendix A

Ringling In The Rain Project File Structure

Server Side:

File Name	Descriptions
RinglingInTheRainServer.cs	Main file of Ringling In The Rain server part application, which contains the server user interface, two data collection agents, the personal information management agent, and the user goal-maintaining agent. Two data collection agents are implemented as independent threads to download weather data and compute the storm movement estimation, while the personal information management agent and the user goal-maintaining agent are running in the main thread, which also handles the socket communication. The shared blackboard is implemented as some global variables that can be accessed from all threads.
RadarImageClass.cs	This file contains all radar image processing methods that all implemented in the RadarImageProcessor class. StormMovementVector and StormTrackerInformation classes are also implemented in this file because they are storm movement estimation related classes.
RouteInfo.cs	This file contains classes used for user movement estimation, including Route and iPoint.

Client Side:

SimpleGPSMIDLet.java	Main file of Ringling In The Rain client part application, which contains the client user interface and the user movement estimation agent.
LocationManager.java	This file contains the main part of the route learning and detection algorithm.
CellData.java CellGPS.java	These two files contain the cell related classes.
GPSFactory.java GPSListener.java	These two files are for obtaining GPS data from the phone.
IntermediateData.java IntermediatePointProperty.java LocationData.java	Files contain various data structure declarations for location and iPoint information.
RouteData.java RouteManager.java RouteVisitTracker.java	Route information management related files.
StorageManager.java	This file is to manage the information store on the phone.
UDPclient.java UDPListener.java	UDP communication functionalities related files.

Appendix B

Ringin In The Rain Communication Message

This appendix describes common communication messages between client and server applications in Ringin In The Rain system in following table.

Message	Usage
RITRMSG-SERVICESTATUSINFOREQUEST	Query service status. (Client -> Server)
RITRMSG-SERVICESTATUSINFORESPONSE: Message Conten	Return service status. (Server -> Client).
RITRMSG-FINISHDUMPINGALLROUTES	Indicate the route dumping process is finished. (Client -> Server)
RITRMSG-ROUTEINFO: Short Route ID, Long Route ID, Front iPoint ID, End iPoint ID, Front Name, End Name, iPoint Number, Delta Time	Send the route information. (Client -> Server)
RITRMSG-IPOINTINFO: Short Route ID, Short iPoint ID, Long iPoint ID, Delta Time, Delta Distance, Latitude, Longitude	Send the iPoint information (Client -> Server)
RITRMSG-DUMPINGROUTEREQUEST	Dump route request (Client -> Serever)
RITRMSG-STARTDUMPROUTE	Grant the dump route request (Server -> Client)
RITRMSG-CALENDARINFOREQUEST	Request for calendar information. (Client -> Server)
RITRMSG-CALENDARINFO: Time-Location	Return calendar information. (Server -> Client)
RITRMSG-SHOWRADARIMAGEREQUEST	Request for radar image (Client -> Server)
RITRMSG-RADARIMAGEINFO: URL	Return the URL to download radar image (Server -> Client)
RITRMSG-POSITIONREPORT1: Long Route ID, Short iPoint ID, Direction	Send user movement estimation in (Route, iPoint, Direciton) format (Client -> Server)
RITRMSG-POSITIONREPORT2: Start Latitude, Start Longitude, End Latitude, End Longitude, Delta Time	Send user movement estimation in (Start Latitude, Start Longitude, End Latitude, End Longitude, Delta Time) format. (Client -> Server)
RITRMSG-RAININFO: Message Content	Send weather prediction result. (Server -> Client)

References

- [1] Caglayan, A. , Harrison, C. , Harrison, C., Agent Sourcebook: A Complete Guide to Desktop, Internet, and Intranet Agents, Wiley (1997)
- [2] Chan, J., Chang, T., Continuous Local Information Delivery System and Method, United States Patent Application 20020052674
- [3] Chung, J., Will You Help Me: Enhancing personal safety and security utilizing mobile phones. Master thesis of program in Media Arts and Sciences, MIT (2006)
- [4] Cohen P., Cheyer, A., Wang, M. Baeg, S., An Open Agent Architecture, Reading in Agents (1994)
- [5] Helsinger, A., Thome, M., and Wright, T., Cougar: A scalable, distributed multi-agent architecture. IEEE SMC (2004)
- [6] Iglesias, C., Gonzalez, J. Velasco, J., MIX: A General Purpose Multiagent Architecture, Intelligent Agents II --- Agent Theories, Architectures, and Languages (1995)
- [7] Johnson, W., System and method for proactive content delivery by situation location United State Patent 6456234
- [8] Larsen, J., Bicycle Production Breaks 100 Million:
<http://www.earth-policy.org/Indicators/indicator11.htm>
- [9] Marmasse, N., Schmandt, C., Location-aware information delivery with comMotion. HUC (2000)
- [10] Marmasse, N., comMotion, Extended Abstract, CHI (1999)

- [11] Meyerson, B., Weather Updates Popular Via Cell Phone:
http://www.sci-tech-today.com/story.xhtml?story_id=0320035A1SQO
- [12] Morris, J., Automated Weather Notification System:
<http://www.ibiblio.org/morris/weather/awns.html>
- [13] Nozue, M., Ozaki, N, and Tsuchiya, R., Just-in-Time Information Delivery System for Passenger Assistance, Quarterly Report of RTRI (2006)
- [14] Schmandt, C., Phoneshell: the telephone as computer terminal. ACM Multimedia (1993)
- [15] Smith, R., Bicycle is king of the road as gas costs rise:
<http://www.iht.com/articles/2006/05/05/business/wbbike.php>
- [16] Soltysiak, S., Ohtani, T., Thint, M., and Takada Y., An agent-based intelligent distributed information management system for Internet resources, INET (2000)
- [17] AccuWeather's Wireless Weather Notification Service:
<http://wireless.accuweather.com/>
- [18] Ambient Orb, Weather Watcher, Ambient Umbrella:
<http://www.ambientdevices.com/cat/products.html>
- [19] Berkeley Building in Wikipedia, http://en.wikipedia.org/wiki/Berkeley_Building
- [20] DON'T GET WET .NET: <http://www.dontgetwet.net/>
- [21] Frequently Asked Questions of The ThunderBolt Storm and Lightning Detector System, <http://www.safetyproductsunlimited.com/faq.html>

- [22] Google Mobile SMS: http://www.google.com/intl/en_us/mobile/sms/
- [23] Interactive radar and weather map from The Weather Channel
<http://www.weather.com/weather/map/currentradar/02139>
- [24] Lightning detector in wikipedia, http://en.wikipedia.org/wiki/Lightning_detector
- [25] National Weather Service, Anywhere/Anytime Weather Forecasts:
<http://www.srh.noaa.gov/cte.htm>
- [26] National Weather Service Weather Data <http://www.weather.gov/tg/dataproduct.html>
- [27] ONIX400 – The world’s first handheld GPS with XM Satellite Weather.
http://www.bushnell.com/gps/gps_features_xm_weather.cfm?section=General%20Use
- [28] Ray tracing in wikipedia, http://en.wikipedia.org/wiki/Ray_tracing
- [29] SkyScan, <http://www.skyscanusa.com/main.html>
- [30] The Weather Channel: <http://www.weather.com>
- [31] ThunderBolt Storm Detector, <http://www.spectrumthunderbolt.com/>
- [32] Weather Underground: <http://www.wunderground.com/>
- [33] XM to Introduce First Personal Weather Tracking System, Demonstrate In-Car Video and More at CES: <http://www.lbszone.com/content/view/1507/2/>