A Location-Aware Architecture Supporting Intelligent Real-Time Mobile Applications

by

Sean J. Barbeau

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Co-Major Professor: Rafael Perez, Ph.D.
Co-Major Professor: Miguel Labrador, Ph.D.
Hyun Kim, Ph.D.
Thomas Weller, Ph.D.
Dewey Rundus, Ph.D.

Date of Approval:
June 15, 2012

Keywords: global positioning systems, location-based services,
mobile phone, Java Micro Edition, Android

DEDICATION

This work is dedicated to my family and friends, especially my wonderful, beautiful, loving, and supportive wife Carlene. I love you more than you will ever know. This is also dedicated to Zach, my new son – I hope that this work will inspire you and show that with hard work, dedication, and the support of loved ones anything is possible

.

ACKNOWLEDGMENTS

I would like to thank my major professors, Dr. Rafael Perez and Dr. Miguel Labrador, for their mentoring, patience, and guidance throughout my untraditional doctoral journey. I would also like to acknowledge the feedback from my committee, including Dr. Rafael Perez, Dr. Miguel Labrador, Dr. Hyun Kim, Dr. Thomas Weller, and Dr. Dewey Rundus, who helped shape and revise this research.

I would like to thank Phil Winters, my supervisor at the Center for Urban Transportation Research (CUTR), for his trust and supervision as I built my research career. Thank you to Nevine Georggi and Ed Hillsman for their partnership on many research projects and the rest of the CUTR Transportation Demand Management Team for their support and collaborations. Thank you as well to CUTR Management for their support of the many research projects that I have been a part of at CUTR. I would also like to thank the many Research Experience for Undergraduates (REU) and graduate students who have contributed in many different ways to the research projects I have been a part of surrounding the work in this dissertation: Alfredo Perez, Isaac Taylor, Marcy Gordon, Khoa Tran, Leon Augustine, David Aguilar, Josh Kuhn, Ismael Roman, Oscar Lara, Narin Persad, Dmitry Belov, Jeremy Weinstein, Paola Gonzalez, Tiffany Burrell, Francis Gelderloos, Joksan Flores, Jorge Castro, Richard Meana, Theo Larkins, Hector Tosado, and Marcel Munoz.

I would like to thank the organizations that contributed funding to the many research projects I have worked on, especially the Travel Assistance Device and TRAC-IT projects that built on my dissertation research, including the National Center for Transit Research, Florida Department of Transportation, U.S. Department of Transportation Research and Innovative Technology Administration, Federal Transit Administration, Transportation Research Board, and National Science Foundation.

I am grateful for the support of Sprint-Nextel's Application Developer program, and in particular the assistance of Ryan Wick, Sprint's Lead Developer Advocate, in providing access to the Location API on Sprint devices as well as facilitating the donation of cellular service and devices to USF that supported our research projects.

Last but certainly not least, I would like to thank my family and friends for their love and moral support throughout my years as a full and part-time student. Thank you to my wonderful wife Carlene, without whom this dissertation and the rest of my graduate work never would have been completed – she has put in more hours supporting me than I have working on my research. Thanks to Zach, my son, for being my inspiration. Thank you to my loving and supportive Mom and Dad, who provided crucial support and instilled a love of learning early in my life, and Momma Brown and Matt who have provided love and support in my college and post-college years. Thank you to my brother Ryan, for his many years of friendship and humor, and sister-in-law Daphna for her constant moral support despite juggling med school (Congrats Dr. Daphna!). Thanks to Sugar (I miss you!) for her companionship during the many hours studying, even if she spent most of it barking at squirrels.

TABLE OF CONTENTS

LIST OF TABLES iii

LIST OF FIGURES v

ABSTRACT xi

CHAPTER 1: INTRODUCTION 1
1.1 Mobile Applications 2
1.2 Positioning Technologies 2
1.3 Location-Aware Mobile Applications 4
1.3.1 Cross-Platform Application Environments 5
1.3.2 Multitasking Virtual Machines 6
1.4 Problem Statement 7
1.5 Contributions 11
1.6 Structure of Dissertation 14

CHAPTER 2: KNOWN LBS ARCHITECTURES 15
2.1 Commercial LBS Applications 15
2.2 Known Location-Aware Architectures 18

CHAPTER 3: PROPOSED ARCHITECTURE – LOCATION-AWARE INFORMATION SYSTEMS CLIENT (LAISYC) 29
3.1 Note to Reader 29
3.2 Architecture Overview 30
3.3 Mobile Device-Side Components 31
3.3.1 Positioning Systems Management Modules 33
3.3.1.1 GPS Auto-Sleep 33
3.3.1.2 Location Data Signing 49
3.3.2 Communications Management Modules 51
3.3.2.1 Session Management 51
3.3.2.1.1 Available Communication Protocols 52
3.3.2.1.2 LAISYC Application Data Transport 54
3.3.2.1.3 LAISYC Location Data Transport 59
3.3.2.1.4 Device-Side Implementation of Session Management 60
3.3.2.2 Adaptive Location Data Buffering 63
3.3.2.3 Critical Point Algorithm 70
3.3.2.4 Location Data Encryption 79

3.4 Server-Side Components 83
3.4.1 Communications Management 84
3.4.1.1 Session Management 84
3.4.1.2 Adaptive Location Data Buffering 86
3.4.2 Data Analysis 86
3.4.2.1 Critical Point Algorithm 86
3.4.2.2 Spatial Analysis 87

CHAPTER 4: EVALUATION 89
4.1 Note to Reader 89
4.2 Evaluation Overview 90
4.3 LAISYC Component Evaluation 90
4.3.1 GPS Auto-Sleep 91
4.3.2 Location Data Signing 117
4.3.3 Session Management and Adaptive Location Data Buffering.... 121
4.3.4 Critical Point Algorithm 129
4.3.5 Location Data Encryption 150
4.4 Innovative Location-Aware Applications Developed Using LAISYC .. 151
4.4.1 TRAC-IT 151
4.4.2 Travel Assistance Device (TAD) 161

CHAPTER 5: SUMMARY AND CONCLUSIONS 179
5.1 Note to Reader 179
5.2 Summary of Problem Statement and Needs 179
5.3 Summary of Contributions 181
5.4 Future Work 187
5.4.1 Location-Aware Mobile App Development 187
5.4.2 Potential LAISYC Improvements 188
5.4.2.1 GPS Auto-Sleep 188
5.4.2.2 Critical Point Algorithm 192
5.4.2.3 Location Data Buffering 192
5.4.2.4 Position Estimation 193
5.4.2.5 Privacy Filter 193

LIST OF REFERENCES 196

APPENDIX A. REPRINT PERMISSIONS 212

ABOUT THE AUTHOR .... END PAGE

LIST OF TABLES

Table 1 - The Location-Aware Information SYstems Client (LAISYC) modules are designed to meet the various critical needs of intelligent real-time mobile applications in Location-Based Services..12

Table 2 - SOAP-encoded messages add a significant amount of overhead to web service requests, approximately 3.7 times as many characters, as shown in this example..56

Table 3 - GPS Auto-Sleep state machine values chosen for experimentation..................95

Table 4 - Horizontal error statistics for indoor GPS accuracy tests...............................109

Table 5 - While the positional error between the two devices is substantially different, the error in speed is much less dramatic..110

Table 6 - When using the 0.1 meters per second min_speed_threshold, the Critical Point Algorithm is able to produce significant data filtering savings with only a slight impact on accurate walking paths.....................................138

Table 7 - Resulting statistics from a walk and a car trip that were both processed using the Critical Point Algorithm with different angle thresholds 147

Table 8 – The Critical Point Algorithm is able to reduce GPS datasets by more than 77% on average while maintaining an average distance error percentage under 10%. 148

Table 9 - TRAC-IT was used as part of a USDOT-funded research project to collect over 4 million GPS data points from 30 users over 2 months.............157

Table 10 - 95% of sessions had less than 3.95% of lost UDP packets 157

Table 11 - When TRAC-IT used LAISYC, device battery life nearly doubled while reducing overall location data packet loss by 2.16% and adding encryption..158

Table 12 - Field tests of the TAD app in Tampa, Florida produced ideal prompts 87% of the time at random stops...173

Table 13 - Field tests of TAD with STAGES students were more challenging, primarily due to close proximity of stops near the USF campus 174

Table 14 - The improved bus stop detection algorithm delivered ideally-timed alerts to riders in 33 of 33 tests 176

LIST OF FIGURES

Figure 1 - The LAISYC architecture consists of software on the mobile device and web application server, with a database server holding persistent server-side data 31

Figure 2 - LAISYC mobile phone-based modules 32

Figure 3 - High-sensitivity GPS receivers can acquire a GPS position more rapidly, and with less dependence on the time elapsed since the most recent GPS fix, than low-sensitivity receivers....36

Figure 4 - GPS Auto-Sleep uses a state machine with various logic evaluations that control the transition between states, which represent changes to the GPS sampling interval values 39

Figure 5 - Navigation mode for GPS Auto-Sleep controls GPS sampling interval directly based on a distance-to-goal (e.g., next turn for real-time driving directions)....47

Figure 6 - Relationships between HTTP, TCP, UDP, and SOAP as networking protocols....53

Figure 7 - The Session Management modules use HTTP for application data and UDP for location data for communication between the mobile device and server 61

Figure 8 - A timeline of Location Data Buffering which shows a TCP failure that results in a series of buffered location data fixes, which are transmitted to the server on the next successful TCP transmission....67

Figure 9 - Adaptive Location Data Buffering occasionally checks for an open connection with the server via TCP to increase the probability of successful UDP transmissions 69

Figure 10 - The Critical Point Algorithm filters out GPS fixes that are not necessary to recreate the user's path 74

Figure 11 - Azimuth calculations are used in the Critical Point Algorithm to determine change in direction 75

Figure 12 - The Critical Point Algorithm maintains a reference to three points that are used to determine whether the second of the three points is a critical point 75

Figure 13 - LAISYC uses a hybrid cryptosystem to protect the exchange of the AES key using HTTPS with SSL, and then uses the AES key to encrypt the location data sent over UDP....81

Figure 14 - 128bit AES is used to encrypt the location data in the UDP payload, with the exception of the session ID which is used by the server to identify the correct symmetric key per device session 82

Figure 15 - LAISYC server-side modules 83

Figure 16 - Even modest increases in the interval between GPS fixes produce extended battery life on the order of hours 93

Figure 17 - A growth function for the state[i]$_{interval}$ values was chosen to grow like an x^2 or 2^x function until it reaches the middle state, at which point it quickly accelerates in growth beyond an x^3 function 97

Figure 18 - Sample GPS Auto-Sleep values are chosen for an exponential growth in the interval between GPS fixes, while the timeout values have an upper-bound of 32 seconds; if a GPS fix cannot be acquired, the interval + timeout line illustrates an upper bound for the total time elapsed at each state. 98

Figure 19 - The largest potential loss of beginning travel path is worst-case scenario when the user travel path is sampled just before they begin moving, since the next GPS sample occurs max_gps_activity$_{state[n]}$ seconds later....99

Figure 20 – When high-sensitivity GPS is able to acquire a fix, it tends to deliver this information close to the expected interval value with an average delay of only 9 seconds. 102

Figure 21 - Proactive GPS scheduling (left) starts the GPS hardware slightly before the scheduled interval value expires, while reactive GPS scheduling (right) waits until the interval period has completely expired before attempting a GPS fix. 103

Figure 22 - GPS Auto-Sleep can miss a substantial part of the beginning trip path if it must transition through all states before starting to record high-resolution travel behavior 104

Figure 23 - Speed thresholds for the GPS Auto-Sleep state machine are selected using observations of speed when stationary and indoors 106

Figure 24 - GPS Auto-Sleep can quickly react to real movement using the *high_speed_threshold* and rapidly begin sampling GPS via direct transitions to state[0] to reflect a more accurate travel path 107

Figure 25 - Scatter plots of indoor horizontal positional accuracy tests 109

Figure 26 - Reliability of accuracy estimates for individual assisted GPS data points was shown to be poor on the evaluated devices, the Motorola i580 (left) and Sanyo 7050 (right) 111

Figure 27 - To evaluate the accuracy of GPS Auto-Sleep, the ground truth state of traveling was manually coded against the behavior of the state machine 113

Figure 28 - GPS Auto-Sleep is able to successfully track the moving or stationary state of the user with a high degree of accuracy. 115

Figure 29 - Execution time for key generation using DSA and RSA asymmetric cryptography 118

Figure 30 - Signature generation test results show that Location Data Signing using DSA and RSA is feasible for implementation on real mobile devices....119

Figure 31 - Estimated battery life with and without Location Data Signing 120

Figure 32 - The information exchanged between the mobile device and server for the HTTP POST vs. XML-based JAX-RPC battery life tests 124

Figure 33 - XML-based JAX-RPC mobile device to server communication clearly has a substantial negative impact on mobile device battery life when compared to HTTP-POST....124

Figure 34 - The location data format used for the payload contents of UDP and TCP packets in the power consumption tests 126

Figure 35 - (a) While at 4 second transmission intervals TCP and UDP have similar power consumption, (b) at 10 second transmission intervals it is evident that TCP consumes approximately 38% more power than UDP....127

Figure 36 - a) All GPS data points generated from a phone are shown on the left, while b) only the critical points generated by the Critical Point Algorithm are shown in the right....129

Figure 37 - The Critical Point Algorithm can more than triple battery life by filtering GPS data and transmitting at an interval of 60 seconds instead of 15 seconds....131

Figure 38 - The Critical Point Algorithm maintains a constant memory requirement during execution by using at most three location data pointers....133

Figure 39 - We observed the GPS speed recorded while a user was casually walking, which includes some speed values of 0 meters per second....135

Figure 40 - When comparing a) all points to b) critical points using a min_speed_threshold of 0.1 meters per second, the general walking path of the user is preserved, with some filtering at the beginning of the trip (bottom left of each image). 136

Figure 41 - Over 97% of the GPS drift shown here at an indoor stationary location can be filtered out by the Critical Point Algorithm when using a 0.1 meters per second min_speed_threshold 138

Figure 42 - Sampled GPS data points create an approximated path of the user with some uncertainty 139

Figure 43 - The distance of the path generated from Critical Point Algorithm will always be shorter or equal to the distance of the path using all GPS data points 141

Figure 44 - Running the Critical Point Algorithm with increasing angle thresholds gradually reduces the number of points that represent the line, which increases the distance error percentage....143

Figure 45 - As the angle threshold for the Critical Point Algorithm increases, there is a general trend towards fewer critical points being generated and an increase in the distance error percentage for both walking and car 144

Figure 46 - For car trips, the Critical Point Algorithm is able to dramatically reduce the full GPS dataset, a), to far fewer critical points , b), with lower angle_threshold values because of longer straight paths 145

Figure 47 - Location Data Encryption using 128-bit AES encryption for UDP payloads is feasible on mobile devices, although it does have a slight impact on battery life 150

Figure 48 - The TRAC-IT mobile application is based on the LAISYC framework to enable simultaneous travel behavior data collection and real-time location-based services 153

Figure 49 - The TRAC-IT mobile application provides a user interface to record input from the individual for mode of transportation, purpose, and vehicle occupancy as well as location data...155

Figure 50 - Path Prediction compares the traveler's real-time location, shown as yellow push-pin markers, against paths from the traveler’s travel history, shown as yellow shaded buffers, to predict the immediate travel path...159

Figure 51 - Path Prediction successfully demonstrated that real-time location-based messages could be sent to the phone using LAISYC and a history of the traveler’s behavior 160

Figure 52 - The Travel Assistance Device mobile application alerts the transit rider of an upcoming destination bus stop 162

Figure 53 - TAD was implemented using the LAISYC framework to support real-time location-aware services...164

Figure 54 - New transit trip itineraries can be created for the TAD mobile app user via the TAD website...165

Figure 55 - Travel Assistance Device mobile app interface that alerts the rider when to exit the bus 166

Figure 56 - The initial bus stop detection algorithm for the Pull the Cord Now alert was defined by a radius surrounding the destination stop 167

Figure 57 - The TAD website shows real-time location updates from the LAISYC framework supporting the TAD mobile and web app 170

Figure 58 - LAISYC Spatial Analysis module on the server compares the real-time location of the user against spatial buffers surrounding the rider's planned route, to determine if the user has become lost 171

Figure 59 - The planned travel path of the bus is used to detect if the rider is lost, versus an estimated path created by connecting bus stop locations, since an estimated path can produce false-positive lost alerts 172

Figure 60 - Some TAD alerts were given early or late due to incorrectly geocoded bus stops, where the actual bus stop position (marker "A") differed from the database location of the bus stop (blue bus icon)............................174

Figure 61 - An improved algorithm for notifying the user when to exit the bus is based on detecting the departure from the second-to-last bus stop [126]......176

Figure 62 - Battery life issues related to GPS appear to be an even bigger challenge with smart phones, including Android devices 185

Figure 63 - Future work on LAISYC can include the addition of two new modules: Privacy Filter and Position Estimation..188

ABSTRACT

This dissertation presents LAISYC, a modular location-aware architecture for intelligent real-time mobile applications that is fully-implementable by third party mobile app developers and supports high-precision and high-accuracy positioning systems such as GPS. LAISYC significantly improves device battery life, provides location data authenticity, ensures security of location data, and significantly reduces the amount of data transferred between the phone and server. The design, implementation, and evaluation of LAISYC using real mobile phones include the following modules: the GPS Auto-Sleep module saves battery energy when using GPS, maintaining acceptable movement tracking (approximately 89% accuracy) with an approximate average doubling of battery life. The Location Data Signing module adds energy-efficient data authenticity to this architecture that is missing in other architectures, with an average approximate battery life decrease of only 7%. The Session Management and Adaptive Location Data Buffering modules also contribute to battery life savings by providing energy-efficient real-time data communication between a mobile phone and server, increasing the average battery life for application data transfer by approximately 28% and reducing the average energy cost for location data transfer by approximately 38%. The Critical Point Algorithm module further reduces battery energy expenditures and the amount of data transferred between the mobile phone and server by eliminating non-essential GPS data (an average 77% reduction), with an average doubling of battery life as the interval of

time between location data transmissions is doubled. The Location Data Encryption module ensures the security of the location data being transferred, with only a slight impact on battery life (i.e., a decrease of 4.9%). The LAISYC architecture was validated in two innovative mobile apps that would not be possible without LAISYC due to energy and data transfer constraints. The first mobile app, TRAC-IT, is a multi-modal travel behavior data collection tool that can provide simultaneous real-time location-based services. In TRAC-IT, the GPS Auto-Sleep, Session Management, Adaptive Location Data Buffering, Critical Point algorithm, and the Session Management modules all contribute energy savings that enable the phone's battery to last an entire day during real-time high-resolution GPS tracking. High-resolution real-time GPS tracking is critical to TRAC-IT for reconstructing detailed travel path information, including distance traveled, as well as providing predictive, personalized traffic alerts based on historical and real-time data. The Location Data Signing module allows transportation analysts to trust information that is recorded by the application, while the Location Data Encryption module protects the privacy of users' location information. The Session Management, Adaptive Location Data Buffering, and Critical Point algorithm modules allow TRAC-IT to avoid data overage costs on phones with limited data plans while still supporting real-time location data communication. The Adaptive Location Data Buffering module prevents tracking data from being lost when the user is outside network coverage or is on a voice call for networks that do not support simultaneous voice and data communications. The second mobile app, the Travel Assistance Device (TAD), assists transit riders with intellectual disabilities by prompting them when to exit the bus as well as tracking the rider in real-time and alerting caregivers if they are lost. In the most recent group of TAD field tests in Tampa, Florida, TAD provided the alert in the ideal location to transit riders in 100% (n = 33) of tests. In TAD, the GPS Auto-Sleep, Session Management, Adaptive Location Data Buffering, Critical Point algorithm, and the Session Management modules all contribute energy savings that enable the phone's battery to last an entire day during real-time high-resolution GPS tracking. High-resolution GPS tracking is critical to TAD for providing accurate instructions to the transit rider when to exit the bus as well as tracking an accurate location of the traveler so that caregivers can be alerted if the rider becomes lost. The Location Data Encryption module protects the privacy of the transit rider while they are being tracked. The Session Management, Adaptive Location Data Buffering, and Critical Point algorithm modules allow TAD to avoid data overage costs on phones with limited data plans while still supporting real-time location data communication for the TAD tracking alert features. Adaptive Location Data Buffering module prevents transit rider location data from being lost when the user is outside network coverage or is on a voice call for networks that do not support simultaneous voice and data communications.

CHAPTER 1: INTRODUCTION

Mobile phones have become one of the most ubiquitous computing devices in modern history. As a result of mass production, cellular carrier subsidies, and decreasing technology costs, more people have access to mobile phones today than any other time in world history. As of late 2011, there were an estimated 5.9 billion mobile-cellular subscriptions worldwide yielding a global penetration rate of 87%, with a 79% penetration rate in developing countries [1].

In developed countries such as the United States, mobile phones are becoming so common that wireless penetration is reaching the point of saturation with only a small percentage of the population not owning mobile phones. For example, in the United States as of June 2011 there are 322.9 million mobile subscriptions with a penetration rate of 102.4%, indicating that a large number of individuals have multiple subscriptions [2]. A contributing factor to this growth is that many individuals are giving up their landline telephones in favor of mobile phones. In April 2011, 26.6% of U.S. households were wireless–only, meaning that they use only a cell phone instead of a landline telephone to make calls [3]. As a result of increasing penetration and reliance on cell phones for a variety of everyday tasks, mobile phones have become important devices to many individuals around the world. A 2009 survey indicates that 82% of Americans never leave their house without their phone, while 42% stated “they cannot live without their phone” [4].

1.1 Mobile Applications

Cell phones have become immensely popular not only for their ability to make phone calls, but also for their ability to perform general computing tasks that previously required expensive personal computers. Perhaps one of the most popular features of modern smart phones is the ability to execute mobile applications. Mobile applications, or “apps,” are software products that are typically developed by a third-party that does not have a direct relationship with the device manufacturer (e.g., HTC, Samsung, Motorola, Apple, Research in Motion), cellular carrier (e.g., Sprint-Nextel, AT&T, Verizon Wireless), or operating system vendor (e.g., Google, Microsoft). Instead, the mobile app is created by software engineers and then directly sold and distributed to the customer, often through online software vending services such as the Google Android Market [5], Apple AppStore for the iPhone [6], Blackberry AppWorld [7], Amazon AppStore for Android [8], and GetJar for Java Micro Edition and Android [9]. As a result of these vending services and an increasing availability of smart phones, the number of mobile apps downloaded has proliferated over the last few years. An estimated 29 billion apps were downloaded worldwide in 2011 [10], an astounding increase of 20 billion downloads since 2010 [10]. Revenues for app developers are expected to increase rapidly over the next few years, with an estimated global app revenue of $7.3 billion in 2011 and $36.7 billion by 2015 [11].

1.2 Positioning Technologies

One key difference between mobile phones and desktop computers is that mobile phones constantly change geographic location, unlike desktop computers, which are tethered to a single physical location for months or years. Even laptops do not have the level of mobility that cell phones offer. Laptops can be moved from one place to another, but typically they are in operation for only several hours at a time and then shut down before being moved. In contrast, mobile phones typically remain on during the entire day and can be actively used when the user is in motion.

During the emergence of cell phones in the late 1990s, the U.S. Federal Communication Commission (FCC) became concerned that extreme mobility of cell phones could cause problems for emergency responders attempting to locate a mobile 911 caller, since, unlike a landline phone that is associated with a street address, little is known about the real-time location of a mobile phone. Even if the 911 operator knows what cellular tower a mobile phone is communicating with, this information is of little help to responders since the coverage area of a single cell tower can be several square miles. As a result of the lack of positional knowledge for mobile 911 callers, the FCC issued the E911 mandate, requiring cellular carriers to implement technologies that could accurately locate mobile 911 callers within 50 to 300 meters, depending on the underlying technology [12]. U.S. carriers tested a wide variety of positioning technologies for their networks. Global System for Mobile Communication (GSM)-based U.S. carriers such as AT&T and T-Mobile chose network-based Uplink Time Difference of Arrival (U-TDOA) to support E911 position requests [13]. Code Division Multiple Access (CDMA)-based U.S. carriers such as Sprint and Verizon chose handset-based Global Positioning System (GPS) solutions for devices on their networks because GPS technology was already integrated into the network as a time reference for CDMA-based wireless communications [13, 14].

Since U.S. cellular carriers were mandated to invest a significant amount of time, effort, and funds into positioning technology implementations, carriers immediately began to investigate commercial applications of these technologies for mobile phone users so they could recover a portion of their investments through user fees. Early deployments of these technologies for commercial purposes become known as location-based services (LBS), which are a general class of services that provide users with some type of information based on their real-time or historical location.

Of the positioning technologies implemented for E911 purposes, GPS-based solutions are by far the most accurate, with an estimated 3-5 meters of positional accuracy under ideal conditions [15-19]. Since this level of accuracy is also sufficient to provide commercial services such as real-time driving directions to mobile phone users, GPS became an attractive technology not only for E911 purposes but also for general consumer LBS. As a result, U.S. T-Mobile and AT&T have since implemented GPS-based positioning technologies in their handsets in order to provide commercial services based on the technology [14]. Global trends of GPS penetration in handsets to support commercial services have also surged upwards, with 79.9% of cell phones shipped in the fourth quarter of 2011 (318.3M) having integrated GPS [20].

1.3 Location-Aware Mobile Applications

With the availability of positioning technologies such as GPS in mobile phones, and the advent of apps, third-party application developers became interested in utilizing location information within their applications. There were two major developments in mobile phones that made widely deployable location-aware mobile applications possible: the emergence of cross-platform application environments for mobile phones such as Java 2 Micro Edition, now referred to as Java Micro Edition (Java ME), and the ability to run applications in the background (i.e., a Multitasking Virtual Machine). Both developments are discussed below.

1.3.1 Cross-Platform Application Environments

The diversity and rapid evolution of mobile phone hardware creates a significant challenge for application developers. If the developer were to design and implement software that directly interfaced with the hardware and operating system for each phone, they would be forced to redesign the application for nearly every single mobile phone model that is released by each manufacturer, an extremely costly task. To ease the burden on developers and create a sustainable mobile application ecosystem, applications platforms that hide some of the lower-level detail of the hardware and operating system (OS) implementation have emerged. Instead of directly accessing these hardware and OS components, application instead interact with interfaces that abstract the underlying implementation details. This design allows the underlying hardware or OS to change and evolve without modifying the higher-level interfaces. Applications can therefore indirectly interact with the underlying hardware without the burden of rapidly redesigning their applications for every new mobile phone model.

Java ME, designed after the cross-platform Java virtual machines initially created for portability of desktop and server applications, was the first cross-platform application environment to emerge for mobile phones. Google's Android is a newer cross-platform environment for smart phones that has recently emerged, although in this dissertation the majority of focus is on Java ME since at the time of this research Java ME was the primary cross-platform environment that was widely accepted in the telecommunications industry [21, 22].

One drawback to the standardization of high-level application programming interfaces across multiple hardware and operating system platforms is that there must be consensus in the industry for how this interface is designed, and this can take time to develop. For example, the introduction of positioning technologies in mobile phones for E911 purposes in the late 1990s and early 2000s did not mean that this technology was immediately available to third-party application developers. In fact, a location application programming interface (API) was not standardized for Java ME until September 2003 [23]. The Java Specification Request (JSR) 179 Location API for Java ME, and the subsequent JSR 293 Location API 2.0, defined a set of functions that a mobile application developer could use to access location information on a Java ME handset that implemented the JSR 179 or JSR 293 standards [22-24]. For the first time, an application developer could develop a location-aware application that accessed positioning technologies such as GPS and could work on devices from many different manufacturers and cellular carriers without significant modification, a critical development in the emergence of location-aware mobile apps.

1.3.2 Multitasking Virtual Machines

The second major development in the emergence of location-aware mobile applications was the ability to run applications in the background. Many of the first Java ME mobile phones released in the early 2000s did not have Multitasking Virtual Machines (MVMs),