A Testbed for Mobile Social Computing

Ahmed Alazzawe, Anis Alazzawe, Duminda Wijesekera
Department of Computer Science
George Mason University
Fairfax, Virginia 22030
{aalazza1, aalazzaw, dwijeeskra}@gmu.edu

Ram Dantu
Department of Computer Science & Engineering
University of North Texas
Denton, Texas 76203
rdantu@unt.edu

***Abstract*— We present a testbed for mobile social computing that can be used to perform research in security, privacy, and context-awareness policies and mechanisms appropriate for a wide range of applications. We compare several mobile platforms and present the rational for our design choices and reasons we chose Android as the primary smartphone for this testbed. We also discuss some of the possible experiments that can be conducted using the testbed.**

I. INTRODUCTION

There are currently four billion mobile devices with an active subscription; if each person had one device that will be enough devices for two-thirds of humanity [1]. Increasingly, these new generations of mobile devices sport more computational power, have richer media capabilities, and have a wider range of connectivity options. These devices are also exposing more capabilities to the applications that run on them and are increasingly doing so with fewer vendor-imposed restrictions. We believe that the combination of openness, computational power, and diverse capabilities of modern smartphones, can be used to construct applications that may ease and enhance intelligent social communications of an increasingly mobile society.

This explosion of powerful phones appearing commercially will naturally lead to more interest and research in security, privacy, and issues of location centric context-aware mobile applications. We are developing a testbed to provide a basis, so that researchers can focus on the specific research requirements rather than infrastructure to setup the experiments. This testbed will provide us the infrastructure and analytic means to test our theories and experimentally verify their utility.

In section two of this paper, we discuss related work. In section three, we discuss the major existing smartphone platforms against the requirements of our testbed. In section four, we discuss our rationale for selecting the Android platform. In section five, we discuss in detail how the testbed is setup and how it will be used. In section six, we discuss our mechanism for receiving feedback and conducting surveys. Section seven covers the method used to measure our accuracy.

II. RELATED WORK

An important part of our testbed is the smartphones. As a single node, the ContextPhone [2] provides an excellent platform for developing mobile social computing applications. We believe that in order promote a wide range of research projects in the area of mobile social computing we need to have a complete testbed, where the smartphones are one aspect of it. The ContextPhone provides a toolkit that exposes base services on Symbian phones. Automatic event logging would be an example of a base service. For reasons that we discuss later in this paper, we standardized on the Android platform. An API for the platform will be developed that will expose testbed specific services. The research experiments will use these services to integrate seamlessly with the rest of the testbed. Therefore, an application that is built on top of our platform will automatically get services such as logging data to a central server.

Reality Mining is a project at the MIT Media Lab built primarily for understanding social networks. They collected data from 100 Nokia Symbian series mobile phones over a period of 9 months [3]. Their experiments mostly revolved around understanding social networks and understanding information flow. They provide their sanitized collected data available for download for researchers. Our testbed is similar in terms of number of phones used and perhaps overlaps in the sensor information collected. We plan to collect other sensor information such as ambient sounds (or lack thereof) and GPS location. Furthermore, we plan to associate user feedback to specific events. In addition to studying some aspects of social networks, we are also interested in experiments involving security and privacy.

III. WHAT MAKES A GOOD SMARTPHONE?

A. Choosing a smartphone platforms

The most important component of the proposed testbed is the mobile platform. Our plan is to standardize on a single platform chosen from a plethora of possibilities. There are several advantages to choosing one standard platform for the testbed. Firstly, it would allow us to focus on developing the research application rather than worrying about the variations between the different platforms. Secondly, it would ease the effort in terms of cost and time required to integrate with other applications. By choosing a testbed that enables these capabilities, the turnaround time between formulating research concepts, constructing prototypes, collecting data, and verifying the experiment's claims will be shorter. We compare different smartphones from each platform in (Table I).

B. Platform Characteristics of Interest

Modern cell phones have similar capabilities, such as a GPS receiver, multi-dimensional accelerometer, high-resolution cameras, and impressive computational and graphics capabilities. Even though the sensory capabilities and computing power currently available is an important factor in choosing a smartphone for mobile social computing, we also considered other factors such as how well the smartphone

TABLE I.
PLATFORM DEATAILS

Platform	Developer	Model	Deployed Units	CPU (MHz)	RAM (MB)	External Storage	GPS	Accelero-meter	Wi-Fi	Bluetooth	Camera (MP)	Language
Android	Google	G1	1 Million (2008) [4]	Qualcomm 528	192	Yes	Yes	Yes	Yes	Yes	3.2	Java Like
iPhone	Apple	iPhone 3G	10 Million (2008) [5]	Samsung 667	128	Internal 16 GB	Yes	Yes	Yes	Yes	2	Objective C
Java ME	Sun controls spec	N96	720 Million (2007) [6]	STMicroelectronics 264	128	Yes & 16 GB Internal	Yes	Yes	Yes	Yes	2.8	Java
Openmoko	Openmoko Inc	Neo FreeRunner	Unknown	Arm9 400	128	Yes	Yes	Yes	Yes	Yes	2.8	Python, Java, C++
Symbian	Nokia	N96	77.3 Million (2007) [6]	STMicroelectronics 264	128	Yes & 16 GB Internal	Yes	Yes	Yes	Yes	2.8	Python, C++
Windows Mobile	Micrsoft	HTC Touch HD	11 Million (2007) [6]	Qualcomm 528	288	Yes	Yes	Yes	Yes	Yes	5	C#, C++

exposes the underlying hardware capabilities, as well as the complexity of the programming model. This has a direct effect on how applications are built and how much effort goes into making the software work on the smartphones. The original platform developers may not have considered some of the use cases that our experiments require. Consequently, we do not want the smartphone to be a limiting factor in the type of research that can be conducted on the testbed. Therefore, an area that we deemed important is the openness of the platform.

C. Integration with other services

We believe that location based services is one of the most important feature a researcher would want from this testbed. All the devices we considered have GPS receivers and their respective platforms expose APIs that provide location information. We are interested in obtaining more data than those provided by GPS to form a richer geographic context for the mobile social applications. Location Based Services (LBS) provide context to the location information provided by GPS.

The most interesting LBS provider we found is Google Maps. It provides features that are useful to researchers in the mobile social computing arena. These include overlays that provide annotation capabilities, direction mapping, and satellite views of the area. These mapping applications are available for mobile devices as online services. Currently the application is not compatible with Brew based devices. In addition to the online applications, there are dedicated programmatic components for the Android and the iPhone. The Android in particular has a well-integrated map view component that comes with the official SDK.

We also considered integration with other aspects of a person's digital life. This includes integration with email, contacts, and calendar information. We found that Windows Mobile, Symbian based phones, and the iPhone integrate well with commonly used enterprise level services such as exchange. Conversely, Android does not currently have such integration capabilities, but is well integrated with Google-based applications, such as Gmail and the Calendar. For use in the testbed, the integration with Google will be more useful, because most users of the phone will more likely use Google apps in their daily life.

D. Openness of the Platform

Another important factor in choosing the standard platform for the testbed is the openness of the platform. We consider this important because historically both the phone manufactures and the service providers have restricted or limited the features that have been available to the applications. This kind of openness comes in two forms. The first is in terms of the capabilities exposed to the application. The second, is the availability of the platform source code released under an open source license.

Each platform developer and smartphone manufacturer deals with openness differently. On one extreme, Apple at one point restricted developers from divulging the contents of the iPhone's SDK. On the other end, Openmoko [7] released everything from the operating system to the CAD design of the phone. Another platform, Java ME, provides good programmatic abstraction for developing applications, but discourages direct access to the device capabilities, and instead only provides functionalities defined in the specs. The Android platform strikes a good balance. The Open Handset Alliance released the source code for the Android under the Apache license. This license is friendly to manufacturers, so they can integrate their own changes to the smartphones. This license will also allow us to modify the platform source code when the exposed API does not provide some required functionality.

E. Concerns, Issues, and challenges

Each platform that we considered presented its own unique challenges.

1) Android: Our biggest concern with Android is that it is relatively new and untested in the market place. This is mitgiated by the resources Google and the Open Handset Alliance are putting into the platform and its promotion. Our

other concern is that the first device with Android, the G1, requires HTC to sign any software updates to the platform. This concern has also mitgated when an unlocked development version of this device was released.

2) Brew: The model they use to distribute applications concerns us because there doesn't seem to be a way to install applications on smartphones without being an "authenticated developer." This restrictation prevented us from reviewing this platform more thoroughly [8].

3) iPhone: Apple has historically been protective of its products and consequently the means to develop applications for them. They released the iPhone SDK months after releasing the iPhone. An unofficial SDK was developed by an independent developer to fill the void. Furthermore, some application developers used undocummented APIs to access some of the iPhone's capabilities [9]. Also, the iPhone does not provide a means to run background processes using the official SDK. We envision that the applications developed for use on the testbed will require processes to run in the background. Apple recommends a work around using push notifications, though this suitable for the testbed.

4) Java ME: Does not provide the needed low level functionality or access to devices that may be needed to access features of newer genereation smartphones.

5) Symbian: A market leader which was recently acquired by Nokia. It has a large base of developers and has proven itself in the marketplace. We do envision that we will need to make low level changes to the testbed platform; and these changes cannot be made until Symbian is open sourced. Nokia plans to release Symbian under an open source license during the first quarter of 2009.

6) Windows Mobile: It is closed source, though it has extensive documentation, excellent tools, and is widely used and thus has a large community. The platform is closed source and we foresee difficulties in developing some types of applications.

IV. Discussion of Android

By choosing Android as the platform to be used for the testbed, we get a collection of rich capabilities, and yet are able to make changes to the underlying platform if the need arises.

A. Architecture

Conceptually, Android is composed of multiple layers [10]. The layer that sits on top of the hardware is the Linux kernel. This allows Android to leverage the work of the Linux kernel developers and the device manufacturers who have already written drivers for Linux. The layer above this is a set of libraries that provide software services to the applications. These include OpenGL for graphics, SSL for cryptographic capabilities, and SQLite for enabling access to embedded database.

The runtime, Dalvik VM, is comparable in purpose to the Java VM. It was designed for resource-constrained devices. A program written for Android is converted to the Dalvik bytecode before it can run on an Android device.

The layer above the libraries and runtime is the application framework. This framework provides an API for Android development. Developers use Java and rely on this API for building their applications. The framework is designed in this way, so that it will feel familiar to Java developers.

B. Abstraction & the Programming Model

Android applications use several constructs [11] such as intents, services, activities, and content providers to perform their tasks. An application expresses its computing needs using an "intent". For example, let us take the case of an application that wants to display a specific website. The application generates an intent to view the URL. Then Android takes that intent and finds the best application that can fulfill this request, such as using the built-in browser.

An activity is a construct that usually represents one screen. A service is a long living UI-less process. This may be useful in playing back audio or performing background tasks. A content provider is a way for an application to provide a uniform view of data. Hence, if an application that stores contact information in a database may want to provide that data to all other application on the Android device; it would do so through a content provider.

V. Building A Testbed

A. Testbed Description

There have not been many mobile phone testbeds setup in the past. The only one we are aware of is the Reality Mining [3] project, used to study human social behavior. We believe that our testbed to be quite unique, in that it is the first to utilize a new class of open source smartphones. This gives us the flexibility, control, and capacity to conduct a wide range of experiments.

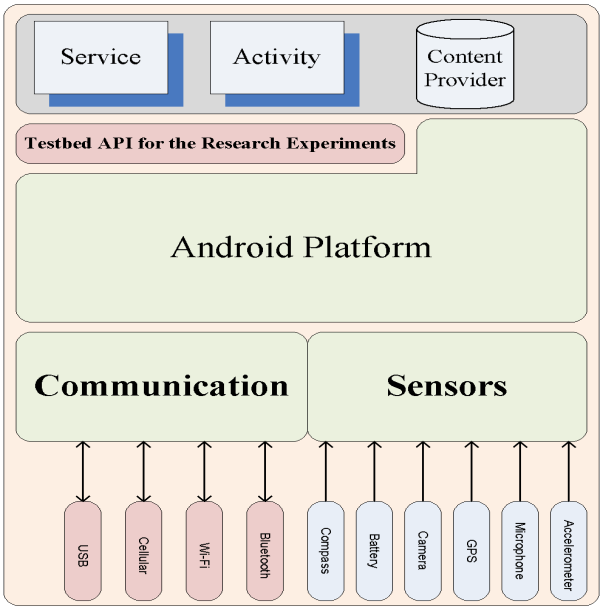

Figure 1. Android Internals

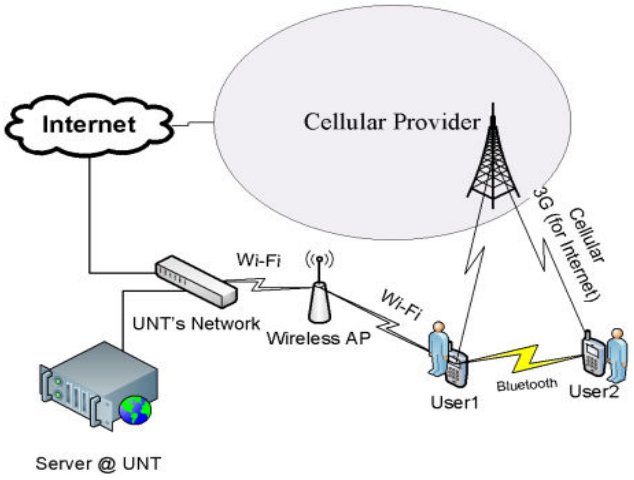

Figure 2. Testbed Overview

We plan to use either the HTC G1 or the HTC Magic [12]. We will be using approximately 100 smartphones distributed to a homogeneous academic community of students, professors, and faculty staff. The main participants will be from the University of North Texas and some from Columbia University and George Mason University. We will develop a specific API for the smartphones. The purpose of this API is to ease integration of the research application with the rest of the testbed (see Fig. 1). As part of the service provided by the API, the smartphones will have a background process running at all times. This process will start once the phone is turned on, and restart itself if terminated. The process is activated anytime an event occurs such as making or receiving a phone call, as well as recording to a log every (say) X minutes. When an event occurs, everything about that event is captured. For example, when a phone call is initiated by the user, different aspects of the call, such as the number dialed, the date and time of the call, the duration of the call, and the location (using cell and GPS information) where the call was initiated from and where it ended is captured. In addition to uploading logs, the phone downloads a new behavior process instruction file. These files are created by the researcher on a need be basis to fine-tune their experiments. All communication between the smartphones and server is encrypted to ensure privacy.

In (see Fig. 2) we show the four primary communication interfaces. They are listed as follows:

- Human-to-Human: This would involve how a human subject would behave or interact with other humans. A subject may at times wish to accept calls from certain callers, while ignoring or sending to voice mail those same calls, at other times. This same subject for example, might always accept calls coming from his elderly parents, no matter where or what time the call occurs.
- Human-to-Phone: This would be a user interacting with a phone. For example, a user taking a survey on the phone, similar to what is shown in Fig. 3.
- Phone-to-server: In two of the three cases, internet is needed to connect to the server. The server would be running an application that would allow the smartphones to connect in order to upload their logs as well as download a new behavior process instruction file.
    1. Using Cellular provider's 3G network for internet connectivity.
    2. If on campus, using Wi-Fi to connect to the University's network. If off campus, using any Wi-Fi to connect to the Internet.
    3. Coming close to the server's Blutooth sensor. This requires the user to be in close proximity to the server.
- Phone-to-Phone: This can be done through regular cellular phone calls, using the phone to send emails, Short Message Service (SMS), using Bluetooh, or chat clients such as Google Talk.

B. *Description of Experiments*

As smartphones' prices drop, and their features increase, people will grow more dependent on them. Smartphones not only increase productivity by allowing anywhere, anytime access to email, web and people, but are now being used for ecommerce transactions [13], voting in elections [14] and even purchasing products from vending machines, [15] as well as maintaining one's healthcare records [16]. For this trend to continue and for users to entrust their smartphones with sensitive health, financial, and corporate information, privacy and security concerns are of immense importance and must be researched.

Smartphones are in essence small computers, and as such, they too are susceptible to spyware, viruses, worms, and trojans. According to F-Secure, a company that protects against malware for personal computers and mobile phones, as of November 2008, there were more than 400 viruses in circulation [17]. Some of these viruses are simply annoying, locking up phones, while others are more malicious, such as continuously sending premium rate texts or tracking the locations of users by pulling GPS information off their phone.

We plan to conduct a number of experiments lasting 3 to 6 months utilizing this testbed. This should generate from 90,000 to 180,000 hours (~20 years) of continuous data collection. This is assuming that our 100 users are active on average, 10 hours a day, 7 days a week. We will conduct a variety of experiments including those that will investigate the security and privacy implications of compromised smartphones.

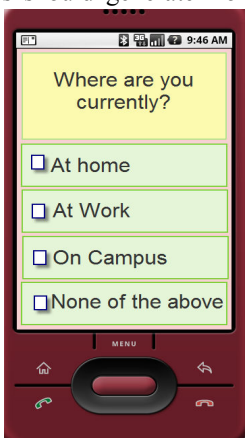

Figure 3. Example Survey

1) Security Experiments: Security on mobile smartphones is as important as security on desktop computers and produces different types of challenges.

Take a scenario where hackers create a worm that exploits some vulnerability on the phone. These smartphones can then be exploited to deploy a distributed denial of service attack (DDOS) on PSTN or VoIP targets. Security on smartphones opens up a new class of issues and challenges that must be researched thoroughly. Some issues that we plan to research include:

- What are the security implications of open source platform such as the Android?
- What security issues arise when moving between a 3G and a Wi-Fi network?
- What mechanisms can be used to secure smartphones against runaway processes or memory leaks that can potentially render the smartphones useless?
- What harm can be caused by having root kits on smartphones, and what types of root kits may be developed on smartphones?

2) Privacy Experiments: Another research area that some of the experiments will focus on, is privacy. There is so much information that can be gathered from a user's smartphone. Spyware can be used to track movement, analyze phone calls to identify social networks, and even enable the microphone to listen in on unauthorized conversations. Some topics of interest include:

- How can we ensure installed applications only access what they claim to access?
- What context-related information will be considered private?
- How can access to sensors on the device be limited to specific trusted applications?

3) Call Predictions: Another type of experiment we plan to conduct is call predictions based on human behavior. The phone will learn a user's behavior sufficiently enough to understand where and when a call is acceptable and not a "nuisance" [18]. This will involve artificial intelligence techniques coupled with reading environmental sensors to help the phone become more context-aware, and respond to incoming phone calls based on the callee's context. Some aspects that will be looked into include:

- How to use the smartphone's different sensors (accelerometer, GPS, etc) coupled with caller's historical behavior to predict call receiving pattern?
- Which non-sensor variables (calendar events, date & time) may be used in conjunction with sensor data to produce a higher accuracy for call prediction?
- What are the ways in predicting calls with a better accuracy in a shorter period of time? This may include having the artificial intelligence program read in a caller's phone bill of previous months to shorten the learning period.

4) Callee's availability: We plan to conduct another experiment where we concentrate on the caller versus the callee. This experiment involves determining the presence of the callee at the time of a call with as few direct or explicit measurements of the caller as possible. For example, we can determine that a callee is probably available for a call, before actually making the call, if we know that the callee is sitting in his office and there is no background noise. In this case, we would detect the presence of a conversation, simliar to how Shazam can identify music [19]. If no sound is detected, than an assumption can be made that there is a good probability that the callee is available for a call. We can determine the location of the callee through information, retrieved with permission, from the callee's cell phone that may include GPS coordinates, cell tower IDs, and/or nearby access points. The final outcome of this project will be a graphical user interface (GUI) for the caller that will provide real-time presence information of contacts in the address book. Such a GUI would be a useful and efficient way of avoiding disturbance and improving the usefulness of smartphones. Some research questions that need to be addressed include:

- How can the presence of specific Bluetooth devices be used to determine availability? For example, a Bluetooth device of another phone deemed to be the callee's boss might indicate not being available.
- How can we identify and categorize ambient noise delivered from different environments to determine business? For example, background noise determined to be a conversation might indicate the callee is not available.
- What algorithms are available to fuse data from the different sensors that may help in determining business/availability? For example, using data provided from the phone's GPS and accelerometer can help determine if the callee is jugging or driving a car.

C. Addressing Privacy Concerns

Continuously mentoring and recording user's daily activities and behavior over an extended period of time creates significant privacy implications. There might be times where the user might not wish to have monitoring enabled; therefore, we will allow the user to disable the application for a predefined time. For example, we would have a pull down menu listing how long the application should be off for (5 min, 30min, 1 hour, 2 hours, 5 hours, user specified). For the user specified option, the user can input how long he would like the application to remain disabled. After which the application will send a message to the user that it is reactivating. In this way, we can ensure the user does not disable monitoring and forgets turning it back on. All captured data can be viewed by the user and all private information is sanitized before being sent to the server. For example, all phone numbers will be hashed, generating corresponding unique identifiers in the analysis.

VI. Feedback & Surveys

There will be instructions posted on a dedicated website, introducing these experiments and expectations from our human subjects. We will also have a kick-off seminar that will help answer any questions and clarify any points regarding the experiments. Recorded instructional video of these events would also be made available on the website. We will use online feedback forms as a mechanism for users to express any issues or irritations they may be experiencing Some of the experiments will require user input to validate information collected and to help some programs learn, such as in the presence/availability experiment. In these experiments, we will solicit user feedback on the phone at the moment an event takes place. Context-Aware Experience Sampling (CAES) is the term coined for this method of data collection [20]. We propose to use CAES in order to minimize retrospective recall, thereby maximizing the validity of collected data. For example, after a call is completed, a multiple choice feedback survey is conducted, as in Fig. 3.

VII. Measurements & Accuracy

In this type of testbed, some of the data points of interest may not make it from the device to the collection point. This can be due to device malfunction, corruption, or the user disabled logging for period of time. We believe that these occurrences will not have a significant impact on the results because we can use startup time and gathered data to determine the nature of the missing components in the log file. There is also a possibility that some of the data logged regarding location might not be very accurate. Even though the Android uses Assisted GPS (AGPS) which leads to faster location acquisition and is usually very accurate (typically 5m-50m), there is still a margin of error [21]. Using Wi-Fi and cell tower triangulation to determine location is also not perfect. We can verify some of this information in the survey questionnaire the user answers.

Every morning, the researchers will have access to all the data that was gathered the day before. This includes phone-generated data such as sensor and phone logs and user generated data from CAES. User responses are compared with program predictions to help determine accuracy. For example, in the presence experiment, our program might indicate it detects the user to be in a library, thus being unavailable. Based on the result of a user feedback, we can determine if the program was correct in its assessment. If not, the researcher can tweak the behavior process instruction file and set it to upload the next time there is a file exchange. The phone will behave according to the new instruction file uploaded by the researcher the next morning. Researchers can continue tweaking their instruction file until the desired results are achieved. We would measure the success of the experiment by how much we were able to reduce false positives and false negatives.

VIII. Conclusion

This paper presents a testbed for conducting research in mobile social computing with a focus on security, privacy, and context-awareness. We compared several mobile platforms and presented our justification for standardizing on Android for the testbed. These include rich APIs, the openness of the platform, and out of the box integration with a wide variety of sensors and services. We also presented the design choices we made for the testbed and put forth example research opportunities that would benefit from using the testbed.

References

[1] Nokia, "NTI_Sensing_-_Dec_2008.pdf (application/pdf Object)," *http://www.research.nokia.com/files/insight/NTI_Sensing_-_Dec_2008.pdf*, Dec. 2008.

[2] M. Raento, A. Oulasvirta, R. Petit, and H. Toivonen, "ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications," *IEEE PERVASIVE COMPUTING*, 2005, pp. 51-59.

[3] N. Eagle and A.S. Pentland, "Reality mining: sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, 2006, pp. 255-268.

[4] D. Shen and A. Hwang, "HTC raises shipment forecasts for G1 and Touch Diamond handsets," *http://www.digitimes.com/news/a20081124PD204.html*.

[5] J. Cheng, "Apple officially surpasses 10 million iPhones sold in 2008," *http://arstechnica.com/journals/apple.ars/2008/10/21/apple-officially-surpasses-10-million-iphones-sold-in-2008*.

[6] "Mobile Market Facts," *http://morvidgames.com/docs/MobileOverview.pdf*.

[7] "openmoko.com," *http://www.openmoko.com/*.

[8] "BREW Forums - FAQ: Can I get/distribute BREW apps for free?," *http://brewforums.qualcomm.com/showthread.php?t=2476*.

[9] "Daring Fireball: Google Mobile Uses Private iPhone APIs," *http://daringfireball.net/2008/11/google_mobile_uses_private_iphone_apis*.

[10] "What is Android? - Android," *http://code.google.com/android/what-is-android.html*.

[11] "Anatomy of an Android Application - Android," *http://code.google.com/android/intro/anatomy.html*.

[12] "HTC - Products - HTC Magic - Overview," *http://www.htc.com/www/product/magic/overview.html*.

[13] Y.F. Chang and C.S. Chen, "Smart phone—the choice of client platform for mobile commerce," *Computer Standards & Interfaces*, vol. 27, 2005, pp. 329-336.

[14] J. Tanner, "Estonia to vote by mobile phone in 2011 - Computers-msnbc.com," *http://www.msnbc.msn.com/id/28197990/*.

[15] "Emerging Payments Industry Briefing," *http://www.bos.frb.org/economic/eprg/papers/briefings/mobilephone.pdf*, 2007.

[16] T. Scannell, "MIT Researchers Hope to Kick Kerberos Up a Notch," *MIT Researchers Hope to Kick Kerberos Up a Notch*, Sep. 2007.

[17] I. Williams, "F-Secure warns of mobile malware growth," *http://www.vnunet.com/vnunet/news/2230481/f-secure-launches-mobile*.

[18] P. Kolan, R. Dantu, and J.W. Cangussu, "Nuisance level of a voice call," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 5, 2008, pp. 1-22.

[19] A. Wang, "An Industrial Strength Audio Search Algorithm," *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2003)*, 2003.

[20] S.S. Intille, J. Rondoni, C. Kukla, I. Ancona, and L. Bao, "A context-aware experience sampling tool," *Conference on Human Factors in Computing Systems*, ACM New York, NY, USA, 2003, pp. 972-973.

[21] G. Djuknic and R. Richton, "Geolocation and assisted GPS," *Computer*, vol. 34, 2001, pp. 123-125.