Designing Public Safety Mobile Applications for Disconnected, Interrupted, and Low Bandwidth Communication Environments

Peter Erickson, Andrew Weinert, and Dr. Paul Breimyer
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02420-9108
Email: peter.erickson@ll.mit.edu

Matt Samperi, Jason Huff, Carlos Parra, and Dr. Scarlett Miller
Pennsylvania State University
101 Hammond Building
University Park, PA 16802

Abstract—**Public Safety emergency communication systems are crucial to effective incident and disaster response. Lack of situational awareness and communications are of the most cited factors that hamper Public Safety's ability to make critical decisions. In partnership with the Department of Homeland Security Science and Technology Directorate (DHS S&T), MIT Lincoln Laboratory (MIT LL) developed the Next-Generation Incident Command System (NICS) to improve collaborative situational awareness. NICS is a reliable, distributed, and scalable architecture that enables a common situational awareness picture to enhance collaboration. The NICS user interface is web-based and Public Safety is increasingly adopting mobile devices, which are likely to operate in disconnected, interrupted, low-bandwidth environments. To address these issues, this paper will describe a Public Safety mobile application based upon feedback from Public Safety personal.**

Due to the strong open-source development community and API flexibility, the Android operating system was selected for development. Development of the native Android application was driven by popular NICS features and Public Safety feedback. Foremost, the application was required to provide and display geolocation information, enabling basic situational awareness. In addition to geolocation, the application enables users to report incidents, broadcast messages, and transmit images. These functions and their designs were driven by interviews with Public Safety personnel from different organizations (law enforcement, fire, etc.). Usability testing with Public Safety personal was conducted. Users envision that the application will enhance the situational awareness capability for mobile Public Safety personnel while laying the foundation for future work leveraging mobile technologies. Based on this testing and lessons learned from development, a set of Public Safety mobile application design considerations were developed.

The nation's emergency response capabilities have received significant attention because of the continued need to prevent, protect from, respond to, and recover from natural disasters, technological compromises to critical systems, and terrorist attacks. These events involve thousands of responders from multiple jurisdictions and agencies working on response, relief, and recovery efforts. Organizing, coordinating, and commanding these efforts are significant technical and operational challenges, requiring timely collection, processing, and distribution of accurate information from disparate systems and platforms. The difficulties in effecting efficient disaster response during large-scale rapidly-evolving events are well documented. Lack of situational awareness and communications is one of the most cited factors that hampers Public Safety's abilities to make critical decisions. Currently, it is difficult to enable shared situational awareness and collaborative command and control across all participating responders and agencies, particularly when infrastructure outages destabilize the communication environment [1].

The Public Safety community requires an integrated sensing and command and control systems that enable collaborative command during disasters. In partnership with the Department of Homeland Security Science and Technology Directorate (DHS S&T), MIT Lincoln Laboratory (MIT LL) developed the Next-Generation Incident Command System (NICS) to address this need [2]. NICS is a reliable, distributed, and scalable architecture that enables a common situational awareness picture to enhance collaboration. Currently, users can access NICS via a browser web-based interface. With users increasingly adopting mobile broadband devices [3], the benefits of NICS can be enhanced with support for these devices.

Previous research has indicated that mobile devices can have a positive impact on the work of law-enforcement [4]. Recently, the New York City Police Department distributed approximately 400 dedicated Android smartphones to its officers that enable them to retrieve a person's criminal history and verify their identification by quickly gaining access to computerized arrest files, police photographs, and state Department of Motor Vehicles databases [5]. FirstNet, responsible for the future nationwide broadband Public Safety network, is also planning a Public Safety specific application store [6].

Furthermore, users are likely to operate mobile devices in disconnected, interrupted, low-bandwidth environments where access to a central server is limited or non-existent. To meet this need, an Android native application was developed for testing NICS in poor communication environments and to identify key features and design choices for any Public Safety mobile application. It was developed as part of the airborne remote communications (ARC) communication project [7].

This work is sponsored by the U.S. Department of Homeland Security under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.

I. Application Requirements

To develop a useful mobile application, a series of customer interviews were conducted to understand Public Safety officials' needs during an emergency. These interviews were carried out with a group of first responders consisting of three police officers, four firefighters, and two Emergency Medical Technicians (EMTs). Their ages ranged from 18 to 50, and their experience varied from one year to more than 30 years. The information was collected through semi-structured conversational interviews, with a list of specific topics to cover.

The interview questions were designed to identify the most common tasks for Public Safety during emergencies. For example, they were asked what kind of feedback is vital to incident response, what type of information needs to be sent when requesting help, and what other types of features would they require in order to efficiently and accurately report a Public Safety incident. Based on the information collected during these interviews, an initial set of base capabilities for the application was planned. These features were centered around the sharing of voice, text, location, and multimedia data, with an emphasis on error-proofing the input and transmission of this data.

A. Geolocation and Situational Awareness

Foremost, it was essential that the application provide location information to the user, enabling basic situational awareness. A user needs to be able to identify their own location, and find the locations of other Public Safety officials or landmarks quickly and easily. The locations of dangerous or suspicious activities, or other areas in need of Public Safety support should also be available to users through the application. Locations disseminated by this service should be error-proof, to prevent false location information from interfering with Public Safety operations. This geolocation functionality will also address a National Institute for Occupational Safety and Health recommendation that the Incident Commander receives pertinent information, such as location, from occupants on scene and information is relayed to crews during size-up [8].

B. Support/Resource Requests

The ability to request support or resources in the event of criminal activity, personal injury, or fire ranked highly among desired capabilities from interviewed first responders. Broadcasting these types of requests to all Public Safety officials involved in an event provides an efficient way of reaching all available personnel, spanning multiple organizations (Law Enforcement, Fire, etc.).

C. Reporting and Messaging

Similar to requesting support or resources to help mitigate adverse Public Safety events, the capability of reporting and collaborating via messages or multimedia was identified as a top priority from these interviews. This capability would enhance situational awareness for mobile Public Safety personnel, while laying the foundation for future work in leveraging mobile technologies. The ability to share images was a common feature request. For example, an image sharing capability would augment the evaluation of suspicious packages.

D. Standard Interfaces

The geolocation, requests, and messages capabilities interface with NICS, which leverages a variety of open standards and formats [9], including the Open Geospatial Consortium Web Feature Service (WFS) standard [10] and JavaScript Object Notation (JSON) format [11]. By leveraging these standards and common practices, NICS can integrate new data sources by adding different data services. For example, due to NICS's modular and loosely coupled architecture, it was able to quickly ingest information from NORTHCOM and the National Guard to support operational use [9]. A Public Safety mobile application should leverage open standards and common practices to maximize integration and minimize integration development time with Public Safety architectures, such as NICS.

E. Disadvantaged Communication Capability

An additional consideration for first responders was the status of the communication environment during an emergency or Public Safety event. The proposed application should be able to function, at least minimally, in a disadvantaged communication environment, with limited bandwidth and possible network outages.

II. Application Modality

Once the requirements of the application were identified, the next step was to evaluate whether a native or web application was appropriate. A native application is one that is downloaded and installed onto a mobile device. A web application is analogous to a web site, in that it "lives" on a remote server that is accessed via a mobile web browser via a network. Both native and web applications offer unique benefits. Since a native application "lives" and operates on the device itself, it has special access to the device's underlying hardware (connectivity, sensors, and computing architecture). A web application benefits from ease of deployment, capability of maintenance, and platform independence.

The advantages of web-based applications are especially attractive to the Public Safety domain. The ability to rapidly deploy a web application to a large group of first responders, all using heterogeneous mobile platforms, could greatly improve common situational awareness. For some of these reasons, NICS was created as a web application, intended for desktop/laptop users. While NICS benefits from many of these advantages, there are drawbacks to this application modality, particularly in the mobile development space.

First, a web application requires an Internet connection that may not be available in a disadvantaged communication environment. A native application can run on a completely disconnected platform once installed on a mobile device [12]. This is a common occurrence in Public Safety operations. Second, the client side of a web application exists inside the sandbox of the mobile web browser, which restricts access to the device hardware. A native application is given broad access to underlying hardware through powerful application programmer interfaces (APIs).

Another key difference between native and web applications is the design and implementation of user interfaces. Most native platforms provide simple, yet powerful, abstractions for common user-interface controls and experiences. Native user interface design patterns are typically familiar and intuitive to mobile device users, since they are typically published by manufacturers, and meant to appear visually and behaviorally similar to the desktop operating system applications. The limitations of mobile browsers and API access to web rendering engines restricts the ability to develop user interfaces in web applications [13]. Furthermore, a web application user interface designed to appear visually similar to a native application runs the risk of falling prey to the "Uncanny Valley" problem [14]. This occurs when an application that appears visually similar to the real thing – in this case a native application – but behaves slightly differently, causes a user to be uncomfortable and have a difficult time interacting with the system.

For these reasons, a native application was prototyped, and this application modality enabled access to the device's sensors, the local storage on the device, and a more responsive application than a web-based alternative.

III. Operating System

Once the decision to pursue native application development was made, mobile operating systems were evaluated to determine an appropriate platform given the application's requirements. Android and iOS were the two logical candidates, since together they composed 91.1% of the mobile operating system market in the fourth quarter of 2012, shown by Table I [15]. Also, Android and iOS collectively gained 15.2% of market share between 2011 and 2012, indicating market growth. Due to highly available and well documented developer support, as well as wide consumer adoption, only Android and iOS were considered.

Android is an open source mobile operating system, built on Linux and developed by Google and the Open Handset Alliance, that uses Java for application development [16]. Hundreds of smartphones and tablets from various manufacturers run the Android operating system (OS). The phenomenon of devices with different form factors and capabilities using a single operating system is known as fragmentation [17]. Fragmentation can hinder rapid application development by requiring developers to make special considerations for different devices and capabilities. Since fragmentation is widely prevalent in Android, libraries are provided to developers to transparently mitigate the cost of supporting different devices.

iOS is a closed source operating system developed by Apple that uses Objective C for application development [18]. The iOS platform is restricted to iPhone, iPad, iPod, and AppleTV devices. Due to the limited number of supported models, fragmentation is less of an issue for iOS development. However, iOS provides fewer transparent options for supporting these devices, and can therefore in many cases be more complicated than the Android equivalent. For example, multiple views are often necessary to support both iPhone and iPad devices, whereas Android view libraries make unified support for smartphones and tablets transparent to the developer.

Fragmentation benefits users by providing a selection of thousands of devices with different capabilities. The device fragmentation experienced by Android OS has helped it to dominate market share through widespread consumer adoption.

With cost, ease of development, and consumer adoption in mind, Android was selected as the target platform. The Android OS offers minimal cost of entry, a well-supported API, and a strong open source development community. Fragmentation, which introduces some development complexities, also provides significant device diversity that's important for the Public Safety community.

IV. Android Platform

After selecting the Android platform, it was important to identify the base version that the application will support; selecting a base Android version directly affects the device models that the platform can support because devices are only compatible with certain versions of Android. One developer has observed over 2000 different device models that downloaded their application [19]. Newer versions of the platform provide better capabilities, but are less widely distributed to devices, since older devices will not necessarily be upgraded to the latest version (often due to technical limitations). The Android platform API is backward compatible; older API features are available in future APIs. To help developers, Google provides a bi-monthly report of platform versions, screen sizes (and densities), and Open GL versions using data collected by all devices that access the Google Play Store, shown in Table II [20].

In deciding which Android version to target, the capability requirements of the application were considered. Since GPS, camera, and Wi-Fi connectivity capabilities are supported by all versions of the platform, these played no role in this decision. However, newer versions of the platform provide easier to use, more intuitive user interface design patterns. Therefore, the newest version of the operating system that reaches a large percentage of the device distribution was desirable. Selecting Android version 2.3.2, "Gingerbread," enabled the application to be compatible with 98.4% of reported devices. This was a calculated tradeoff between the higher functionality of newer platform versions and wider device support.

TABLE I. Top five smartphone operating systems [15]

Operating System	4Q12 Market Share	4Q11 Market Share
Android	70.1%	52.9 %
iOS	21%	23%
BlackBerry	3.2%	8.1%
Windows Mobile	2.6%	1.5%
Linux	1.7%	2.4%
Other	1.3%	12.1%

TABLE II. Android Platform Versions from June 3, 2013 [20]

Version	Codename	Distribution
1.6	Donut	0.1%
2.1	Eclair	1.5%
2.3 – 2.3.2	Gingerbread	0.1%
2.3.3 – 2.3.7	Gingerbread	36.4%
3.2	Honeycomb	0.1%
4.0.3 – 4.0.4	Ice Cream Sandwich	25.6%
4.1.x	Jelly Bean	29%
4.2.x	Jelly Bean	4%

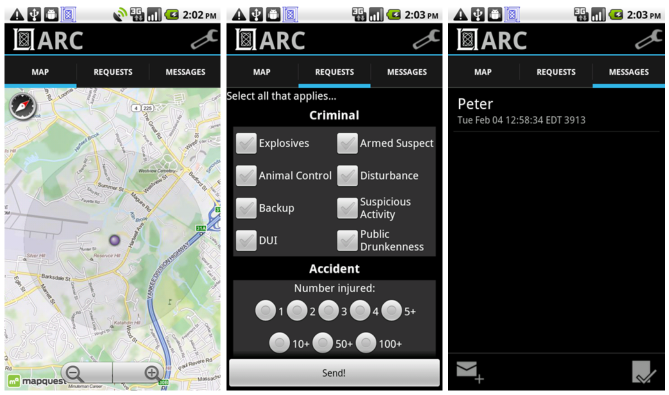
V. User Interface Design

Google and the Open Handset Alliance provide extensive documentation for Android user interface (UI) best practices. These range from style recommendations to interface patterns and building blocks. For example, themes, colors, typography, icon design, and writing style, as well as the action bar, tabs, lists, dialogs, and input fields are included. Additionally, guidelines for effective UI content, such as simplicity, clarity, and brevity are described in the context of the Android OS and the provided UI libraries [21].

Using the requested capabilities as a goal, and with the Android UI best practices in mind, a user interface was prototyped on paper. The application would feature three unique views: Map, Requests, and Messages. A tabbed layout, provided by the the standard Android UI library, was used to switch between these different modes. Separating these three major views simplified the design. This is important because it allows first responders to quickly find what they need and conforms to the Android design principle of "only show what I need when I need it [22]." Existing images stored on mobile devices and on-demand photography were chosen as a sharable forms of imaging data instead of video in order to conserve bandwidth in communication-limited environments. However video sharing can easily be incorporated in the future if the user requirements warrant it. Figure 1 provides screen shots of the basic map and request tabs.

A. Action Bar

One of the features of newer APIs that is not available in Android 2.3.2 is the action bar. The action bar is a reserved portion of the screen that shows application information and allows for navigation between different windows or tabs. It is meant to reduce user interface clutter and maximize the consistency of user interaction with the application. This feature is available with version 3.0 and above and is a key feature of the current Android design principles. Fortunately, since the action bar was received very positively by the Android developer community, a library was created to provide the action bar features to platforms down through version 2.0. This library, ActionBarSherlock [23], was incorporated into the application.



Peter
Tue Feb 04 12:58:34 EDT 3913

(a) Map (b) Requests (c) Messages

Fig. 1. NICS mobile application screenshots

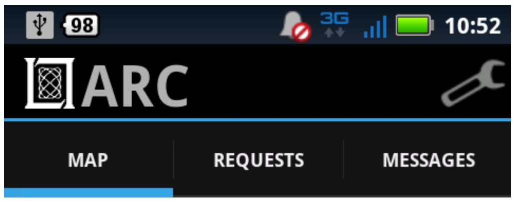


Fig. 2. Mobile NICS Action Bar

The action bar library provides multiple methods of changing application modes, including drop-down list selection, fullscreen list selection, and tabs. Tabs were chosen as an easy way for users to both see the current application mode (the currently selected tab), and see or select other application modes. A settings button is also displayed on the action bar. The same action bar interface is displayed across all operating modes of the application for consistency. According to the Android design guidelines, this interface provides ease-of-use and intuitiveness expected by mobile device users. Figure 2 shows the action bar for this application.

B. Map Tab

One of the most powerful features of NICS is the shared map display, enabling a common situational awareness picture. When developing the mobile application, it was important to include access to the same map information and allow users to collaboratively modify content. Map APIs where evaluated for their ability to support common overlays (lines, polygons, text, etc.) with a visually intuitive interface.

Google Maps is a well-known candidate for a map API because Google is the main developer of Android and it is well supported. However, the newest version of the API lacks support for older device versions, specifically devices running OpenGL 1.1 or below, which includes some devices at or above the targeted OS version of 2.3.2 [24]. Upon further research into available libraries, the MapQuest Android API [25] satisfied all overlay requirements and supports devices running older Android platforms (including older versions of OpenGL). The MapQuest API is provided under a free license, and allows access to free, open map data, such as OpenStreetMap. Additionally the Mapquest API is designed as a "drop-in" library; incorporating the Google Maps API in the future would be a straightfoward integration task if desired.

NICS provides access to geospatial collaboration data via numerous standard formats, including Web Map Service WMS and Web Feature Service endpoints [26]. In order to reduce communication overhead, the mobile application was designed to load data from this endpoint using JSON. JSON is well supported in Android due to the native JSON encoder and parser, org.json [27]. JSON tends to be a more lightweight communication format than other alternatives, such as eXtensible Markup Language (XML). The loading and parsing of NICS data was designed to be independent from the chosen mapping implementation, to allow for plug-and-play geospatial visualizations. This allows the map icons, features, and colors to be identical between the mobile application and the NICS web-based interface. This architecture demonstrates the common software principles of low coupling and high cohesion [28].

The MapQuest API was leveraged for the display of NICS geospatial data. A fullscreen map is presented to the user under the 'Map' tab, and allows for panning and zooming. The user's location is detected via the GPS sensor and automatically sent to NICS at a configurable interval. Overlays from NICS are also overlaid, which includes other users' locations.

C. Support/Resource Requests Tab

The 'Requests' tab was designed to provide an easy way for Public Safety officials to report incidents and request support or resources. Various types of incidents were identified from the customer needs interviews, ranging from criminal activity to personal injury to fire. As such, this tab is divided into three categories: Criminal, Accident, and Fire. Originally, each of these categories occupied its own tab, but this was later reduced to a single 'Request' tab for multiple reasons. First, limited mobile device screen real estate made it difficult to see all of these categories without requiring the user to scroll the action bar horizontally, which was deemed less intuitive by users. Second, there was confusion about whether a single request could be made that spanned multiple categories at once: for example, a criminal incident that resulted in personal injuries and a fire. Users indicated that requesting resources and reporting incidents are similar activities, and therefore the unified 'Requests' tab was more intuitive.

Reports can be made for incidents including explosives, driving under the influence, and suspicious activity. Requests for backup, rescues, fire control, and many other options were deemed important by interviewed first responders. Additionally, a text field can be populated by typing or using speech-to-text. Users indicated that the speech-to-text capability was particularly valuable given the small form factor of mobile devices and the prevalence of protective gloves for first responders. The application allows users to attach a photograph by either selecting an existing image from the device's picture gallery or taking a new picture. The user location is also included with the submission.

D. Messages Tab

The 'Messages' tab consists of three parts: an inbox of received messages, an outbox of sent messages, and a form to compose new messages. The Messages tab title displays the number of new, unread messages in the inbox for convenience. This is visible at all times, regardless of which tab is active on the screen, to draw the attention of the user.

Tabs within the Messages screen were initially intended to allow users to switch between the inbox and outbox. However, tabs within tabs are not natively supported by Android, and are discouraged by the Android design guidelines, due to concerns about confusing the user. Instead, the "spinner" pattern, or drop-down list, was adopted. The currently selected mode is displayed and a list allows the user to change to a different mode. Incoming messages are received when the map overlays from NICS are updated. The update rate can be configured by the user in the application's settings page.

A button to compose a new message is always present at the bottom of the Messages window. The new message screen allows for a typed or speech-to-text message to be composed and users can attach images as well. Like requests, messages always include the location of the user at the time of submission.

The inbox is populated with any requests or messages sent from any other user. The outbox is populated with any requests or messages sent by that device. As previously mentioned, requests and messages have no specific destination; they are broadcast to all users participating in the incident response/recovery.

VI. Usability Testing

A usability study of the mobile application was conducted with nine participants: four firefighters, four EMT's and one police officer. Future studies will strive to expand the user validation group. Each subject was asked to sign a confidentiality form and participants were provided fifteen dollars per interview.

During the test, the subjects were asked to perform eight different tasks on the medium fidelity prototype on an android phone. The number of clicks each subject required to complete each task was recorded, as well as the time it took them to complete each task. The order in which the tasks were given to each subject was randomized in order to avoid any ordering effect. The subjects were videotaped while performing the tasks, and the student organizers read the tasks, recorded the number of clicks, and timed each task. The eight tasks are as follows:

- Report a house fire, with two people inside;
- Report an incident of public drunkenness;
- Report a suspicious activity with a brief description and take a photo;
- Report a car accident where one person is seriously injured;
- Request a helicopter;
- Send a message indicating you are going on break;
- Check date and time of last sent message; and
- Find the nearest intersection.

After completing the tasks, each user was asked the following questions:

- How realistic were the tasks to your typical needs?
- How useful do you feel each feature was?
- In relation to other software used, how does this prototype compare?
- What do you think about the application's design and layout?

The results from the post-test questionnaires showed that the user group participants found the concepts of the system to be favorable and useful in performing their tasks as first responders. Often during conversations with the participants they openly discussed potential situations where such a system would have been beneficial. The Public Safety users deemed the application interface to be sufficiently intuitive for navigation and use by any first responder, from complete novices to proficient Android users. All users completed all assigned tasks without major concerns or confusion.

Two of the EMT's we studied were search and rescue specialists, and they found the application map and personnel tracking capability to be extremely helpful in performing a successful grid search. Several subjects emphasized that the ability to send pictures of suspicious packages or wanted individuals to all first responders in the area was extremely useful.

The developed mobile Android application was deemed intuitive to learn and operate. First responders were able to quickly use the application without any prior instruction. Many of the features in the application were found to be useful, with the map feature universally deemed the most impactful.

VII. Conclusion

Public safety is increasingly adopting mobile devices and applications to enhance incident and disaster response. These applications must support existing architectures, such as NICS, and operate in disconnected, interrupted, and low-bandwidth communication environments. A mobile application was developed with Public Safety feedback to identify important mobile application development considerations. Based on lessons learned and usability testing, the following Public Safety mobile application development high level considerations were identified:

- When developing a native application, consider developing for Android first because of its significant market share. Android fragmentation also prevents Public Safety users from being committed to a single device manufacturer;
- Although Android fragmentation exists, developers can take advantage of backwards compatibility and third party APIs to maintain a consistent and current design across Android devices;
- It is not recommended to build a web-only application if it is intended for poor communication environments. Most current web applications are limited without a consistent network connection; and
- Although Public Safety organizations have different missions, their basic mobile requirements are very similar.

In a separate and independent effort, Metron Inc. recently released SARApp, a search and rescue smart phone application [29], sponsored by the Defense Advanced Research Projects Agency (DARPA). SARApp, which also supports NICS integration, follows many of the identified considerations. It is compatible with Android 2.1 and forward; designed for general Public Safety users; communicates with NICS; and can operate in a poor communication environment. The independent implementation of the design considerations provides some validation of the results presented in this paper.

Future work will focus on leveraging the computational power of the mobile device for processing, exploitation and dissemination of shared Public Safety information; developing algorithms and techniques to maximize effectiveness in poor communication environments; and developing deeper NICS integration.

Acknowledgment

The authors greatly appreciate the support and assistance provided by Dr. Bob Griffin, Marc Caplan, Jose Vazquez, and K. Phil Waters from the U.S. Department of Homeland Security Science and Technology Directorate. Additionally the authors wish to thank the Student Space Programs Laboratory and Dr. Sven Bilen. Finally, the authors would also like to thank MIT LL staff members Kimberly Chang, Gregory Hogan, and Charles Rose.

References

[1] B. S. Manoj and A. H. Baker, "Communication challenges in emergency response," *Communications of the ACM*, vol. 50, no. 3, pp. 51–53, 2007.

[2] A. Vidan and G. Hogan, "Integrated sensing and command and control system for disaster response," in *2010 IEEE International Conference on Technologies for Homeland Security (HST)*, 2010, pp. 185–189.

[3] J. Cohen, "AppComm.org the application community," Westminster, CO, Jun. 2013.

[4] S. G. Straus, T. K. Bikson, E. Balkovich, and J. F. Pane, "Mobile technology and action teams: Assessing BlackBerry use in law enforcement units," *Computer Supported Cooperative Work (CSCW)*, vol. 19, no. 1, pp. 45–71, Sep. 2009. [Online]. Available: http://link.springer.com/10.1007/s10606-009-9102-2

[5] W. Ruderman, "New tool for police officers: Quick access to information," *The New York Times*, Apr. 2013. [Online]. Available: http://www.nytimes.com/2013/04/12/nyregion/new-tool-for-police-officers-quick-access-to-information.html

[6] P. Suh, "Applications summary," Westminster, CO, June 2013.

[7] A. Weinert, P. Breimyer, S. Devore, J. Miller, G. Brulo, R. Teal, D. Zhang, A. Kummer, and S. Bilen, "Providing communication capabilities during disaster response: Airborne remote communication (ARC) platform," in *Homeland Security (HST), 2012 IEEE Conference on Technologies for*, 2012, pp. 395–400.

[8] C. Fischer and H. Gellersen, "Location and navigation support for emergency responders: A survey," *IEEE Pervasive Computing*, vol. 9, no. 1, pp. 38–47, Jan. 2010. [Online]. Available: http://comp.eprints.lancs.ac.uk/2259/

[9] R. Di Ciaccio, J. Pullen, and P. Breimyer, "Enabling distributed command and control with standards-based geospatial collaboration," in *2011 IEEE International Conference on Technologies for Homeland Security (HST)*, 2011, pp. 512–517.

[10] "Web feature service | OGC(R)," Jun. 2013. [Online]. Available: http://www.opengeospatial.org/standards/wfs

[11] D. Crockford, "The application/json media type for JavaScript object notation (JSON)," Jul. 2006. [Online]. Available: http://tools.ietf.org/html/rfc4627

[12] J. White, "Going native (or not): Five questions to ask mobile application developers," *The Australasian Medical Journal*, vol. 6, no. 1, pp. 7–14, Jan. 2013, PMID: 23424610 PMCID: PMC3575060. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3575060/

[13] A. Charland and B. Leroux, "Mobile application development: web vs. native," *Commun. ACM*, vol. 54, no. 5, p. 4953, May 2011.

[14] B. Higgins, "The uncanny valley or user interface design," 05 2007. [Online]. Available: http://billhiggins.us/blog/2007/05/17/the-uncanny-valley-of-user-interface-design/

[15] "Android and iOS Combine for 91.1% of the Worldwide Smartphone OS Market in 4Q12 and 87.6% for the Year, According to IDC," Press Release, February 2013. [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=prUS23946013

[16] "Android," Jan. 2013. [Online]. Available: http://www.android.com/

[17] A. Cotas, "Androids fragmentation problem is moving in the right direction," Jun. 2013. [Online]. Available: http://au.businessinsider.com/androids-fragmentation-problem-improves-2013-6

[18] "Apple - iOS 6," Jun. 2013. [Online]. Available: http://www.apple.com/ios/

[19] "Android fragmentation visualized - OpenSignal - OpenSignal," Aug. 2012. [Online]. Available: http://opensignal.com/reports/fragmentation.php

[20] "Dashboards — android developers," Webpage, June 2013. [Online]. Available: http://developer.android.com/about/dashboards/index.html

[21] "Design — android developers," Webpage, Google, Inc., June 2013. [Online]. Available: http://developer.android.com/design/index.html

[22] "Design principles," Jun. 2013. [Online]. Available: http://developer.android.com/design/get-started/principles.html

[23] "ActionBarSherlock - home," Jun. 2013. [Online]. Available: http://actionbarsherlock.com/

[24] "Google maps android API v2 google developers," Jun. 2013. [Online]. Available: https://developers.google.com/maps/documentation/android/start

[25] "Android maps API - MapQuest developer network," Jun. 2013. [Online]. Available: https://developer.mapquest.com/web/products/featured/android-maps-api

[26] C. E. Rose, C. Mayer, and D. P. Breimyer, "Optimizing map server performance in resource constrained networks," in *Free and Open Source Software for Geospatial (FOSS) 2011*, 2011.

[27] "org.json," Jun. 2013. [Online]. Available: http://developer.android.com/reference/org/json/package-summary.html

[28] R. S. Pressman and D. Ince, *Software engineering: a practitioner's approach*. McGraw-hill New York, 1992, vol. 5.

[29] "SARApp home." [Online]. Available: http://www.sarapp.com/