BUILDING MOBILE SENSOR NETWORKS USING SMARTPHONES AND WEB SERVICES: RAMIFICATIONS AND DEVELOPMENT CHALLENGES

ABSTRACT

Wireless sensor networks are composed of geographically dispersed sensors that work together to monitor physical or environmental conditions, such as air pressure, temperature, or pollution. Wireless sensor networks are used in many industrial, social, and regulatory applications, including industrial process monitoring and control, environment and habitat monitoring, healthcare, home automation, and traffic control. Developers of wireless sensor networks face a number of programming and deployment challenges, such as networking protocol design, application development, and security models. This chapter shows how smartphones can help reduce the development, operation, and maintenance costs of wireless sensor networks, while also enabling these networks to use web services, high-level programming APIs, and increased hardware capability, such as powerful microprocessors. This chapter examines key challenges associated with developing and maintaining a large wireless sensor network and presents a novel smartphone wireless sensor network that uses smartphones as sensor nodes. We validate our work in the context of Wreck Watch, which is a smartphone-based sensor network for detecting traffic accidents that we use to demonstrate solutions to multiple challenges in current wireless sensor networks. We also describe common pitfalls of using smartphones as sensor nodes in wireless sensor networks and summarize how we have addressed these pitfalls in Wreck Watch.

1. INTRODUCTION

Developing large-scale sensor networks has traditionally required physically deploying and managing many customized sensor nodes. Likewise, harvesting sensor data efficiently has required complex networking techniques, such as energy-aware routing protocols, data-centric protocols, location-based protocols, or hierarchical protocols (Akkaya) (Zang) (Lee). Once sensor data was collected, moreover, substantial effort was needed to process and visualize the data, or to take responsive actions. Physical upkeep of the sensor nodes also required teams to visit and maintain deployed sensors.

Modern smartphones are sophisticated computing platforms with complex sensor capabilities, such as detecting user location, recording high-quality audio, measuring ambient light, sensing geomagnetic strength, and sensing orientation (Mohan). Due to widespread use of smartphones, it is now possible to develop large-scale sensor networks using cellular network technology and deploy applications on end-user devices to collect and report sensor readings back to servers. End-users also often have a keen interest in maintaining their phones, including repairing broken hardware, re-installing faulty software, and maintaining data synchronization with servers. This end-user maintenance helps alleviate much of the maintenance burden from operators and other network administrators.

Millions of Apple iPhones and Google Android-based phones have been sold by the 3rd quarter of 2009 (Betanews). The potential size of smartphone wireless sensor networks is directly related to the number of smartphones being used daily by end consumers. The large number of purchased smartphones suggests that smartphone sensor networks could contain hundreds of thousands of nodes. Much previous work on sensor networks, such as environmental monitoring and first-responder systems, can be adapted to mobile smartphones, where that work will likely achieve more dispersion and adoption per unit of effort than conventional methods of deploying mobile sensor networks (Leijdekkers).

After sensor data has been collected, it must be processed, visualized, and shared with users. Web service APIs are another emerging trend that can help in this task. For example, Google offers public services for geocoding addresses, sharing pictures and video, displaying maps, and overlaying data across

satellite imagery. Likewise, some services, such as Google's App Engine and Amazon's EC2 compute cloud, offer free or low cost computational grids for analyzing data (Buyya). Utilization and composition of these types of web service APIs has allowed rapid application development.

By combining web services and the advanced computational power of smartphones, applications can contain real-time information filtered using the metadata of individual users, such as location, social relation, or application settings. Data from multiple users can be combined and used in conjunction with available web services to create powerful applications involving real-time, location-aware content. The combined data can also be shared through content distribution networks, such as YouTube and Facebook.

This chapter presents the challenges and promising solution approaches associated with developing large-scale, sensor-based applications using smartphones, such as iPhone and various Android phones, and web services, such as Google Maps, Amazon S2, or the Facebook API. The remainder of the chapter is organized as follows: Section 2 presents the Wreck Watch application as a case study of a smartphone wireless sensor network; Section 3 describes key challenges found in traditional wireless sensor networks; Section 4 offers solutions to these problems utilizing combinations of smartphones and web services; Section 5 introduces new challenges arising from using smartphones and web services; Section 6 discusses solutions to these new challenges in the context of Wreck Watch; Section 7 outlines future research, including new, unsolved problems caused by the introduction of smartphones and web services into wireless sensor networks; and Chapter 8 summarizes key points and lessons learned.

2. MOTIVATING CASE STUDY

To motivate challenges and benefits of developing large-scale wireless sensor systems using mobile phones and web services, we present a case study based on *Wreck Watch,* which is a mobile phone application that runs on Google Android smartphones (Android) and detects car accidents in real-time. As shown in Figure 1, Wreck Watch detects car accidents (1) by analyzing data from the device's GPS receiver and accelerometer to detect sudden acceleration events from a high velocity that may indicate a collision. Car accident data is then transmitted via an HTTP POST request, where it can be retrieved by other devices in the area to help alleviate traffic congestion (2), notify first responders, and provide accident photos to an emergency response center (3). Users of Wreck Watch can also elect to have certain people contacted in the event of an accident via an SMS message or a digital PBX.

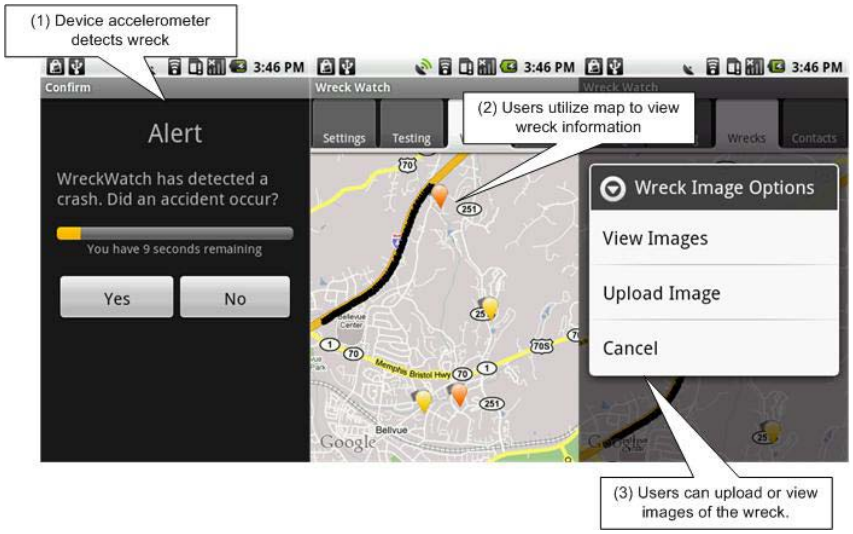

Figure 1 – Wreck Watch Behavior

When smartphone users install the Wreck Watch application on their device it effectively integrates their smartphone into the Wreck Watch wireless sensor network, which we call "SmartNet." Installing

this application provides users with several benefits, e.g., the application will monitor the smartphone it is installed upon and detect collisions. The left-hand side of Figure 1 shows the alert screen that Wreck Watch presents after detecting that a user may have been in an accident. To reduce any possibility of erroneous accident reports, the user is given ten seconds to indicate that an alert should not be triggered.

Wreck Watch allows users to store emergency contacts on the Wreck Net HTTP server. In the event of a collision, a user's emergency contacts can be notified of the accident via email, SMS message, or pre-recorded audio clip. An example SMS message we used for demos was "John Reeds has wrecked! Call 866-901-4463 Ext 123 for details! You are on John's emergency contact list." Automatic information sharing of this sort can allow first responders more time to focus on the accident, and the accident victim, rather than keeping emergency contacts up to date. The SmartNet system can be continuously updated with the latest information, and can re-notify emergency contacts as more data becomes available.

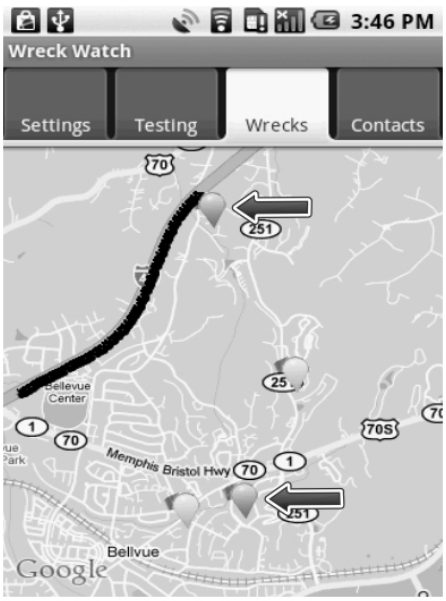

Figure 2 – Wreck Watch Accident Display Screen (arrows indicate more severe accidents)

Another Wreck Watch feature is the ability for users to view other wreck locations, as shown in Figure 2. This feature allows users to route themselves around other accidents, resulting in improvements of traffic flow near an accident location. Wreck Watch users are informed of the severity of wrecks by utilizing a color-coded wreck marker (in Figure 2, two markers are indicated as darker in color, thus implying greater severity).

Wreck Watch users may select a map marker by tapping on that marker. This selection opens an alternate menu that allows users to upload new media to associate with that wreck or to view media currently associated with it. This feature allows bystanders—who are likely not medically trained—a quick and effective method of sharing valuable information with first responders. While a trained medical first responder may be able to ask focused questions to a bystander on the telephone, receiving a photo of the accident scene may convey information more quickly and accurately. SmartNet also acts as a temporary digital storage medium, allowing parties involved in a collision to capture and store media associated with the event for later reference. This type of storage is especially beneficial for insurance claims and is demonstrated in Figure 3.

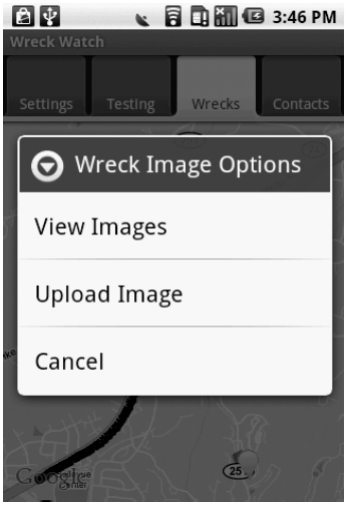

Figure 3 – Wreck Image Options

We designed Wreck Watch to detect/respond to accidents and provide information to other users of the sensor network. We also developed a web browser interface to SmartNet, intended for use by first responders and shown in Figure 4. This interface can be used to allow first responder dispatchers to communicate with field, or mobile, units and adjust their tasks based on current data. For example, while a mobile unit is traveling to an accident scene, an uploaded picture may show the type of injury sustained. The first responder dispatcher can then inform the mobile unit what type of injury it should prepare for, saving valuable seconds upon reaching the accident scene. This browser interface can also be used to display metrics that are of interest to first responders, such as an overall traffic risk level on a given day, number of mobile units deployed and their locations, or type of mobile units deployed.

Figure 4 – Web Browser Interface to SmartNet

Throughout the chapter, we use Wreck Watch to demonstrate the advantages of using a smartphone wireless sensor network over traditional sensor networks.

3. CHALLENGES OF TRADITIONAL WIRELESS SENSOR NETWORKS

This section describes the challenges of building, deploying, and maintaining a large wireless sensor network. While some challenges (such as complex networking protocol decisions) are implementation-specific, we present other issues (such as sensor distribution and maintenance) that are fundamental to a wide range of wireless sensor networks. In addition to defining each challenge, we also provide examples of hardships or failures caused by the challenges.

3.1 Challenge 1: Monitoring Mobile Human Subjects Requires a Large Quantity of Sensors

Many sensor networks currently attempt to measure characteristics of mobile human populations, such as social interactions, package delivery personnel monitoring, or monitoring of healthcare patients. For a wireless network with spatially fixed nodes to measure these properties, that network must have a large number of nodes spread over a large geographical area. Distribution and maintenance of such networks can be costly since the larger the network, the more likely nodes will fail, and the more researchers and developers must make repairs.

Getting human participants to wear a mobile sensor, such as an accelerometer, is challenging and prone to risks, such as dropped devices. For example, participants may be unwilling to wear a sensor clipped to their clothes due to fear of undesirable social responses. Moreover, sensors may be bulky and interfere with the movements of participants. Sensors also often require frequent re-charging, which can burden end users. These same basic issues apply to other sensors, thereby making mobile sensors a large challenge.

3.2 Challenge 2: Sensor Distribution and Maintenance is Time-consuming and Costly

Many wireless sensor networks must contend with the cost of maintenance. Sensors are often located in remote areas, either because the effect to measure is in a remote area or the sensor must be safe from—and not interfere with—individuals. Sensor network maintenance is thus often time consuming and expensive (Intanagonwiwat). For example, network maintainers must travel to a malfunctioning or damaged device and attempt a field repair, or remove the device, return to the laboratory for repair, and then return the device to its original location. Moreover, maintenance schemes, such as battery change schedules or solar arrays, must be developed to power the devices during the long periods of time that they may be deployed in the field.

Sensor maintenance can be particularly problematic for traditional spatially-fixed wireless sensor network, such as those used to detect car collisions. For example, when utilizing fixed-location sensors, multiple sensors would be deployed throughout a large geographical area to detect collisions along the numerous streets in a city and its surroundings. These fixed sensors would be exposed to harsh environmental conditions, and potentially to damage from the accidents they were intended to monitor. Every time a sensor was damaged, that sensor must be identified as faulty or unresponsive, geographically located, and retrieved for repair. If multiple geographically dispersed sensors were repaired frequently, the cost of maintaining the sensor network would be large.

3.3 Challenge 3: Limited Sensor Node Computing Capabilities Require Complex Low-level Software Optimization

In traditional wireless sensor networks, sensor nodes typically have meager processing capabilities to minimize cost and limit power consumption. Many sensor node hardware constraints directly impact the complexity of the software that can be run on the node. These limited hardware resources make it hard to implement complex sensor data harvesting and processing platforms, and force researchers and developers to create highly optimized software platforms using the low-level primitive operations available on the node. Developing software stacks on top of these low-level sensor node platforms can consume valuable

researcher time, introduce errors in the end node software or hardware, and increase the overall cost of the project.

Less powerful sensors, such as those used in traditional sensor networks, is hard to utilize to detect car collisions accurately. To accommodate less powerful hardware, complex low-level software optimizations must be performed to fit within the limited processing capabilities of the device. Sensor nodes also have limited amounts of memory available to store software, resulting in further software memory space optimization. Heavy, immensely complex optimization of the node software is thus necessary to achieve the same network that Wreck Watch provides.

3.4 Challenge 4: Sensor Network Data Communication Requires Complex Communication Protocols

Many wireless sensor networks use some type of *ad hoc* networking, in which nodes cooperate to relay valuable information back to a base station connected to the Internet, which in turn relays information to researchers (Madden) (Manjeshwar), as shown in Figure 5. This relay of information is typically complex, with multiple potential points of failure in this networking protocol. If a sensor node fails, other nodes must be able to detect the failure and intelligently restructure routing within the network. The nodes closest to a base station are typically used much more heavily than other nodes, and thus tend to fail more frequently due to hardware errors or battery exhaustion.

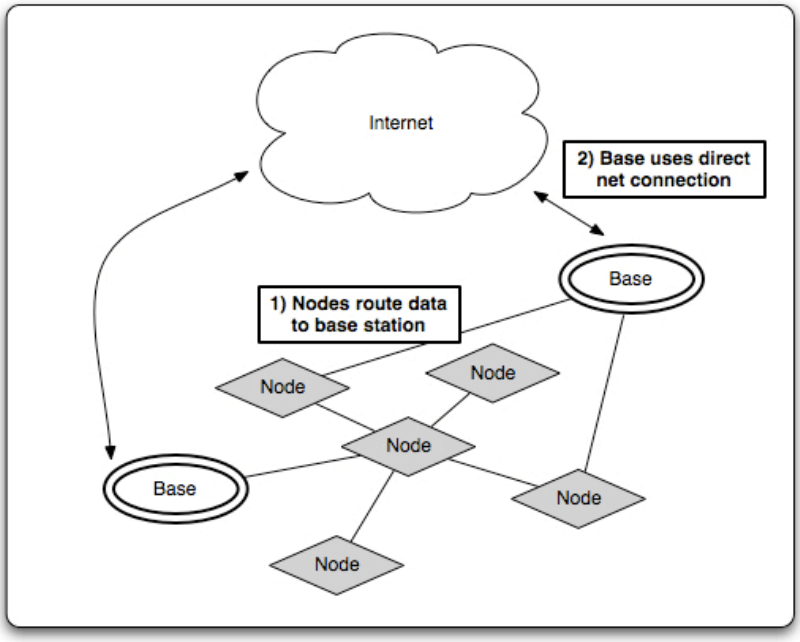

Figure 5 – Traditional Wireless Sensor Network Routing

4. SMARTPHONE AND WEB SERVICE SOLUTIONS FOR TRADITIONAL WIRELESS SENSOR NETWORK CHALLENGES

This section presents a novel architecture, called SmartNet, for building sensor networks. After we present SmartNet, we show that this smartphone-based sensor network architecture addresses the challenges raised in the Section 3, such as sensor maintenance costs, software optimization, and communication complexity. The Wreck Watch application utilizes the *SmartNet* framework, with specific details added to allow wreck detection, reporting, and visualization. Below we describe an example of SmartNet

to showcase the differences between it and traditional sensor networks. The SmartNet smartphone sensor network architecture is shown in Figure 6.

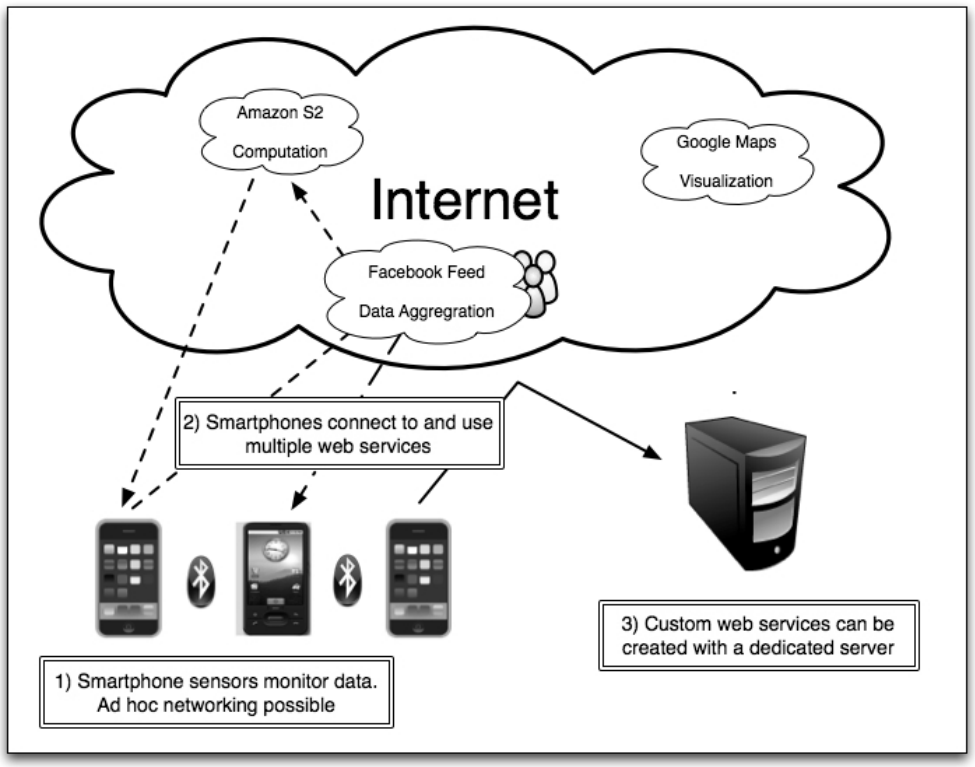

Figure 6, SmartNet, a Smartphone Sensor Network Architecture

The sensor network architecture uses smartphones as sensor nodes to capture environmental information, such as acceleration, ambient light, and imagery. These smartphones are fully capable of networking in an *ad hoc* manner or directly connecting to the Internet. Moreover, these devices have permanent storage space, processing power, and battery life comparable to an older laptop (low gigabyte memory range, 400–700 Mhz, 1 day charge). Relative to many current sensor nodes, these smartphones are capable of greater processing, data transfer, and battery life.

For smartphones to become part of a wireless sensor network, smartphone users can opt to install an application on the phone. On many smartphones, users are encouraged to only download and install applications from the *de facto* distribution location, such as Apple's iTunes or the Google Android Application Market. Typically, this installation process takes the form of an application store installed on the phone, with the store itself being another application. Network creators are thus alleviated from the burden of distributing, and upgrading remote sensor nodes by providing an over-the-air provisioning mechanism.

The sensor node application provides the instructions to allow the smartphone to communicate with the wireless sensor network and also provides instructions on how to connect with web services to retrieve or visualize data. These instructions are typically written on top of high-level programming APIs (such as the Java Software Development Kit or iPhone Core Location) to simplify smartphone software implementation effort. Smartphone operating systems (such as Mac OSX and Android) often provide standard system libraries that allow rapid application development for simple tasks.

To augment the feature set of these high-level smartphone programming APIs, web services can be used as smartphone sensor network information aggregators that provide reusable data processing

pipelines. Many web services also provide visualization of data sets, allowing complex visualizations to be quickly implemented. While web services force applications to use the service APIs, combining multiple web services into a single application can overcome the limitations of only using a single web service.

Data generated from smartphone sensors, as well as from the results of web services data aggregation, can be stored into a single data storage server. This presence on the Internet allows researcher and developers to effectively generate a web service over which they have complete control. This controlled web service can be used to power the smartphone sensor nodes in complex ways that public web services may not offer.

The remainder of this section focuses on addressing the Challenges raised in Section 3 by specifically utilizing mobile smartphone solutions. We provide multiple examples in the context of Wreck Watch.

4.1 Addressing Challenge 1 by Using End-user Smartphones to Monitor Mobile Human Populations

Many current fixed location sensor networks attempt to measure inherently mobile humans by placing multiple sensors in a dispersed geographical region. The SmartNet approach to these problems provides much greater sensor mobility. This provides equal coverage in busy locations, where many smartphones are traveling, and possibly much better coverage in rural locations. The same mobile approach could be applied in multiple other sensor networks to similar advantage, such as disaster relief worker tracking, measuring air quality, or measuring noise pollution levels. In general terms, smartphones can be used to power wireless sensor networks designed to measure any mobile, largely dynamic, and un-predictable properties. In these inherently mobile tasks, smartphones are more appropriate sensor network nodes, and can perform better than multiple fixed-position sensors.

In creating Wreck Watch, we wanted to measure an inherently mobile property, namely car collisions. The Wreck Watch application utilized smartphone accelerometers to detect and report car wrecks to a centralized server. This smartphone approach was much more feasible than deploying traffic collision sensors at multiple locations around the city, and had the added advantage of allowing users to either opt in or out of using the network.

4.2 Addressing Challenge 2 by Relying on Smartphone Owners for Sensor Hardware Maintenance

In previous sensor networks, the maintainers of the network would have to manually locate, travel to, and repair faulty nodes. With a smartphone sensor network, the owner of the individual phone has an investment in the device that leads them to maintain the phone, and therefore maintain the sensor. Moreover, the end user provides the power maintenance for the device by charging it to avoid battery depletion. Software errors can also be found after deployment in wireless sensor networks. Most sensor network software errors would previously require physically contacting the device to update the hardware. Many smartphone operating systems, however, provide methods to upgrade applications remotely. Moreover, these upgrade methods are not transparent to the device, and an application running on the mobile device is knowledgeable about which ‘version’ it is, which allows all data emerging from a smartphone network to include the node software version so each block of data can be traced to a specific version of software.

This software upgrade feature thus allows a researcher to not only remotely push upgrades to a smartphone node, but it also allows a very precise count of exactly which nodes have upgraded, and knowledge about which blocks of data can be attributed to which version of the smartphone software. In the event of a flaw being found in the software, not only can the error be remotely corrected, but also the update can occur without taking the network offline. While some smartphone nodes may take longer to update, they will also report their currently running software version, allowing a researcher to filter data from a corrupted version of the application.

4.3 Addressing Challenge 2 by Utilizing Smartphone Application Distribution Networks for Sensor Node Software Distribution and Upgrade

Before the introduction of smartphones as a possible medium for sensor networks, sensor networks capable of spanning similar geographical distributions required massive effort and cost. Utilizing smartphones to power a wireless sensor network can offer a huge advantage in the number of nodes attained per unit of cost. Many modern smartphones support a common application distribution mechanism. For example, Apple's App Store is searchable on the iPhone and traditional computers, and accessing the App Store is the only method for most iPhone users to receive software. These distribution centers typically host the applications themselves, and allow users to download directly from the distribution network.

This over-the-air provisioning removes the need for a network creator to manually create a geographical distribution of nodes. Excluding certain specialized sensor networks that require additional hardware, the distribution of smartphone applications is largely free to the application creator. Most distribution networks also offer capabilities to perform software upgrades, which alleviate the need for a network maintainer to travel to the node to upgrade the node software. Over the lifetime of a smartphone-powered wireless sensor network, the main costs incurred by the researcher include the cost of creating an application that would turn a smartphone into an end node, the cost of maintaining data collection servers, and the possible cost of marketing the application. Resource-constrained researchers and developers can thus budget less money on gathering data and more on analyzing the results.

Wreck Watch, for example, is fully deployable using over-the-air provisioning. Users can download the application from the Android Market, and run it on their device. As we decide to add new features to the network, such as monitoring new data measurements of interest, or as we find errors in the software that require software upgrades, we will use the Android Application Distribution Network to provision the software upgrade. Each user of Wreck Watch will be notified of the upgrade and can update their node version at their convenience. Furthermore, irregular upgrade schedules are not a problem for the data integrity of SmartNet, as every node can reliably add its version number to any data reported.

4.4 Addressing Challenge 3 by Leveraging Powerful Smartphone Processors and High-level Programming APIs to Simplify Sensor Software Development

In smartphone sensor networks, the user has purchased hardware with processing capabilities more comparable to old laptop hardware, typically including a processor above 400 Mhz, multiple sensors (including input and output devices), and multiple networking connections. In general, smartphones are more powerful than most current commodity sensor nodes, such as the Intel Mote (Nachman). The extra processing power of these nodes would allow raw data collection and data processing in levels that were previously impractical.

Sensor nodes typically have limited resources, such as memory and processing capabilities, which often result in software that is stove-piped and tightly coupled to a specific sensor network requirement set. This tight coupling makes it difficult to reuse software components across sensor network projects, requiring costly reinvention and rediscovery of existing software solutions. The improved hardware of smartphone devices make it possible to write software that is more loosely-coupled to the underlying hardware, allowing researchers to focus on creating only domain-specific parts of the sensor software, such as statistical analysis models, rather than all operations. Additionally, some platforms allow the running application to reuse installed third-party components, such as advanced video decoding libraries, to accomplish tasks. For example, Google Android's Intent system allows developers to call code from other applications, or even from a software library. This design allows a development model based on reusing and aggregating components into new applications.

When creating Wreck Watch, we utilized multiple external libraries. The Android developer libraries were heavily used to access the sensor nodes, create the user interface, and detect car accidents. Additionally, the Java system libraries were used to perform timing operations and manipulations. When a user

attempted to attach an image to a collision location, Wreck Watch utilized the Android Intent system to allow the user to take a picture with any camera library of their choosing. Utilizing the large software libraries available allowed the researchers to focus on the domain specific components of Wreck Watch, such as reacting to collision events, and to complete the implementation of the entire Wreck Watch system in 4 weeks.

4.5 Addressing Challenge 3 by Utilizing Web Services to Supplement Smartphone Visualization and Processing Capabilities

After sensor data has been collected, it must be processed, visualized, and shared with users. In many cases, the limited processing capabilities of smartphones make it difficult to process large quantities of data, and the relatively slow network speeds (3G, WiFi) make sharing a large amount of data difficult. Additionally, visualization of data on a device can involve complex programming and large amounts of time. Web service application programming interfaces (APIs) are an emerging trend that can help in these tasks. For example, Google offers public services for geocoding addresses, sharing pictures and video, displaying maps, and overlaying data across satellite imagery. Some services, such as Google's App Engine and Amazon's EC2 compute cloud, offer free or low cost computational grids for analyzing data. Utilization and composition of web service APIs allow rapid sensor network and application development.

Besides processing and visualization, another strong advantage of using a web service for a wireless sensor network is the capability to re-package the solution rapidly on multiple smartphone platforms. If the brunt of processing and visualization is done by utilizing public web APIs, then porting the smartphone application to another platform can be as simple as programming needed user interfaces specific to that platform. Following the code reuse principle, most of the visualization of common web services tasks, such as placing markers on a Google Map, can be accomplished by using libraries packaged with the smartphone operation system by default.

Using web services, and the advanced computational power of smartphones, applications can contain real-time information filtered using a broad range of metadata describing individual users (such as location, social relation, or application settings). Data from multiple users can be combined in an arbitrarily complex manner, and used in conjunction with available web services to create powerful applications involving real-time, location-aware content. The combined data can also be shared through content distribution networks, such as YouTube.

Creating a third party application to tie together multiple web services can be completed quickly and easily. For example, displaying the hometowns of Facebook friends on a Google map simply requires connecting to both API's, retrieving the data from Facebook, and passing it to a Google map. The complexity of web services can be utilized to power large parts of the application's data processing and visualization, saving time and money.

While creating Wreck Watch, we utilized Google's Maps API to display reported accidents on a web browser page intended for use by a first responder. A distinct advantage of using the web API to do this visualization, rather than a custom component, was that displaying the same data in the Wreck Watch application on the device took little time to program, as the smartphone operating system already includes classes to allow Google Maps functionality. The collected data was, in turn, useful to end users who were interested in routing around wrecks, which enticed them to download the application and become a participant in the sensor network.

4.6 Addressing Challenge 4 by Relying on Multiple Networking Connections to Avoid Networking Complications

Challenge 3 in Section 3 described the problem of communication, both amongst nodes and between nodes and the Internet. Many smartphones have hardware to support both ad hoc protocols, such as Bluetooth, and direct Internet connections, often over WiFi, 3G, 4G, or WiMAX. The direct Internet connec-

tion can be used as the main connection for relaying data back to data storage facilities. Having multiple possible networking protocols removes the forced necessity for many of the complex ad hoc wireless sensor communication schemes, such as energy, location, or data aware routing protocols, which we see in use currently, without removing the possibility for ad hoc networking amongst the nodes (Akkaya). This flexibility in the choice of networking protocol allows a node to dynamically switch the protocol it is using for network connectivity, based off of high-level decisions about communication range, speed, battery usage, etc, rather than always using a single ad hoc protocol.

For example, a wireless sensor network intended to measure the amount of unique social interaction most individuals have on a fixed-time basis would likely use Bluetooth to detect when two individuals came within close contact of one another. The application could occasionally intelligently switch to using a 3G connection to the Internet, allowing it to report the interaction metrics of interests back to a central data collection server quickly, without wasting valuable resources.

5. ISSUES INTRODUCED BY POWERING WIRELESS SENSOR NETWORKS WITH SMARTPHONES AND WEB SERVICES

Using smartphones as nodes in a sensor network raises many concerns and problems. This section identifies key challenges associated with using smartphones for sensor networks and proposes solutions to these problems. We focus primarily on the issues and problems that are unique to creating sensor networks with smartphones and web services, reifying these issues in the context of Wreck Watch.

5.1 Issue 1: Use of Web Services Restricts Flexibility

Although utilization of web services, such as Google Maps, is a significant benefit to many areas of wireless sensor networks, those same web services can restrict development flexibility. A consequence of using web service APIs is a loss in customization of what the API provides. For example, some web services do not provide full API access to their data or algorithms, to preserve proprietary intellectual property. Many web services do not, and will likely never, offer their most valuable features. For example, Google will likely never release its AdSense algorithms, or its routing algorithm. While it might be feasible to develop a similar routing algorithm, there is no immediately feasible method to get marketing data similar to the data that powers AdSense. Developers also face the problem of repetition. Web services are available to many developers, and many simplistic ways of connecting APIs and platforms have already been programmed.

5.2 Issue 2: Protecting Network Security, Privacy, and Data Integrity

While previous sensor networks had to deal with potentially malfunctioning sensors, users of phones in a smartphone sensor network may intentionally or unintentionally sabotage sensor network data. These concerns are not the only security issues with using smartphones as a sensor network data source. Much interesting sensor data compromises end user security and privacy, if not handled properly. Many sensor measurements of interest contain GPS location data, which reveals real-time, accurate information about not only the smartphone, but also the individual using the phone. While users may consent to this data being shared with the sensor network, the creators of the sensor network must ensure that this data stays safe from third parties.

Many web services have a catch-all clause that reserves the full legal rights to any data their service operates on, or any results their service produces. This presents significant privacy issues for personal user data. Additionally, for many government funded research sensor networks, this ownership of secure data may not be acceptable. Researchers interested in utilizing web services should be aware of these clauses, and aware of the types of data they are passing to a web service at any given time.

5.3 Issue 3: Smartphones Have Responsibilities Other Than Powering the Sensor Network

Traditional wireless sensor network hardware tends to focus solely on powering the wireless sensor network. When creating sensor networks using smartphones, however, researchers and developers must consider that end nodes cannot be fully dedicated to the sensor network. In particular, many other functions of smartphones, such as performing like a mobile phone, take priority over powering the wireless sensor network. While there are many advantages of using smartphones for sensor networks, there are other possible uses for the end node, so researchers and developers must be respectful of other applications or processes. If a smartphone is deployed for the sole purpose of powering the sensor network, this issue may not be problematic, though many smartphones will be used for other purposes, as well.

Figure 7, Sensor Applications Must Share Resources

We developed Wreck Watch so that it is aware of other responsibilities and occasionally evicts unnecessary components when the system is low on memory. Wreck Watch also allows users to set the amount of data they wish to store for their GPS trail. This feature has a direct impact on the amount of permanent memory the smartphone has available.

6. SOLUTIONS AND RECOMMENDATIONS FOR POWERING WIRELESS SENSOR NETWORKS WITH SMARTPHONES AND WEB SERVICES

While there are many issues created by utilizing smartphones and web services to power wireless sensor networks, most of these issues can be addressed when designing the smartphone network and programming the sensor node. We present many solutions to the issues raised in Section 5.

6.1 Addressing Issue 1 by Integrating Multiple Web Services with Smartphone Sensors to Ensure Sensor Node Application Uniqueness

Although utilization of web services is a significant benefit to many areas of wireless sensor networks, those same web services can restrict development flexibility. A consequence of using web service APIs is a loss in customization of what the API provides. For example, some web services do not provide full API access to their data or algorithms, to preserve their own unique value proposition. Many web services do not, and will likely never, offer their most valuable features. For example, Google will likely never release its AdSense algorithms, or its routing algorithm. While it might be feasible to develop a similar routing algorithm, there is no immediately feasible method to get marketing data similar to the data that powers AdSense. Additionally, a developer faces a problem of repetition. Web services are available to many developers, and many simplistic ways of connecting APIs and platforms have already been programmed.

While web services are powerful components that provide large computational power, visual presentation, and data collection to a smartphone, the user of the web service has no control over the API presented. More importantly, the user has no control over what the web service implementers choose not to add in the API. While an API such as Twitter's may provide a large amount of data, the Twitter API provides no easy method of visualizing that data.

To achieve the full goals of the application, a combination of multiple web services can be used in conjunction to create a more valuable end product. Additionally, with the programming capabilities of smartphones, sensors can be used to augment and enhance the web service by providing device-aware data. Many examples of this exist, such as using GPS on a phone to automatically filter search results to local venues, using orientation to route the user relative to both their position and their orientation, and using microphones to automatically increase or decrease volume on Internet radio. Finally, many wireless sensor networks generate some metric of interest. By combining these metrics with multiple web service APIs, it is possible to rapidly generate a complex application.

When creating Wreck Watch, we combined generated data about the location and severity of accidents with a Google Maps application. While the visualization aspects of Google Maps were expansive enough to present the data from Wreck Watch, we were unable to present the media that was associated with a wreck using the Google Map. The solution to the problem was to request data from both the Google Maps server, and from the SmartNet server. The information about the collisions, and the associated media, can essentially be considered a web service over which we have complete control. By combining our data-centric web services with the visual services offered by Google, we were able to generate a complex application in a matter of weeks.

6.2 Addressing Issue 2 by Incorporating Fine-grain Security and Privacy Controls

While many security and privacy concerns exist for almost all wireless sensor networks, smartphones can require additional considerations in these domains (Fleizach). Smartphones are inherently a hub for massive amounts of personal user data, such as contacts, email, or social networking sites. In consideration of this, many smartphone operating systems include programs that allow remotely resetting the device. In the event of a lost device, the device can be remotely erased to preserve the confidentiality of the data.

In general, however, remote resetting of devices is an emergency measure. On a daily basis, applications should provide some privacy and security. To address this, many applications have a comprehensive set of application settings that allow users to decide exactly what is being sent into the sensor network. In Wreck Watch, these additional settings allow users to decide which personal information, such as device phone number, emergency contact information, and location data (including the amount of GPS route data posted), to include in a collision report, as shown in Figure 8.

Figure 8, Emergency Contact Privacy Settings in Wreck Watch

6.3 Addressing Issue 3 by Respecting Other Applications' Access to Device Hardware through Smartphone OS Resource Management APIs

Almost all modern smartphone operating systems provide high-level API callbacks that inform the current application of the overall state of the device. These callbacks include information about available memory status, battery life, and changes to phone settings. Programs should be aware of these callbacks, and react to them in an appropriate manner. Developers should follow conventions set for programming for their specific smartphone operating system.

For example, Wreck Watch was initially developed without regard to an Android OS convention that described how many times a user should poll the GPS unit per second. After field testing, we determined that running Wreck Watch in the background on an Android G1 would empty a full battery in less than an hour. Rather than polling the recommended amount (once per minute), Wreck Watch was requesting GPS information as fast as the sensor could deliver it. The operating system returned approximately one GPS reading every few nanoseconds, with required that the GPS hardware and the G1's processor both constantly remain in a powered on state. By modifying the request rate to respecting the convention that the development guide had set, the battery life was extended back to the standard one-day charge.

7. FUTURE RESEARCH DIRECTIONS

Exploring the potential of smartphones to power wireless sensor networks is still in its infancy. As individual smartphones become more technologically advanced, the power and flexibility of using the smartphone as a wireless network sensor increases greatly. The networking and computational power of smartphones has increased greatly in the last decade, and many smartphones are now coming with a plethora of advanced sensors built in (Ballagas). Common sensors include an accelerometer, an ambient light sensor, and a proximity sensor. Some devices contain even more sensors, including a geomagnetic sensor and an orientation sensor. In addition to traditional sensors, many smartphones contain other hardware, such as GPS location hardware, camera hardware, speakerphone hardware, and a microphone.

The number of sensors, combined with inherent phone and smartphone hardware allows previously impossible hardware combinations. These increased abilities open up many unexplored areas of development, especially research pertaining to large-scale sensor networks. When integrated with web services

the joint flexibility of these two approaches is quite large. This section summaries areas in which we encountered difficulties or unexplored issues as potential candidates for further research.

7.1 Mixing Mission-Critical and Enterprise Class Applications or Services

Web services offer many advantages for data processing and aggregation in wireless sensor networks, but they are not without their own issues. Many wireless sensor networks, especially networks with real-time or safety concerns, such as Wreck Watch, require guaranteed availability. Web services vary greatly in the amount of availability they offer, and typically make no guarantees as to service uptime. For example, the Twitter service experienced around 6 full days of downtime in 2007 (Dorsey). Likewise, more popular services are typically built on top of sturdy, business-class architectures designed to handle the load that popular service receives. Unfortunately, during large events (such as a natural disaster), the number of requests on such popular web services could become overwhelming. Similarly, many web services do not offer prioritization of requests, even at a rough granularity, or reliability of computational results. This poses problems for many mission critical sensor networks, and these features will hopefully be added as web services become more common.

7.2 Ensuring Coverage in a Fixed Location

While one of the major benefits of utilizing smartphones in wireless sensor networks is their mobility, this can be a problem for networks that have a need to guarantee coverage in a fixed geographical location. Mobile smartphones cannot easily ensure one specific area of coverage, and are generally better suited to a broad area of impact. For example, while Wreck Watch can provide coverage in many locations, it can never guarantee coverage 100% of the time in any specific location. If there were a busy city that typically had many smartphones traveling on it, then that street would have excellent coverage. However, if a lone car was driving on that street at an off-peak time, such as 3 A.M., then that car could have an accident, and there would be no smartphone node available to detect the wreck.

7.3 Mobile Security, Without User Hassle or Frustration

Current mobile platforms do not provide a clean balance between allowing flexibility, and retaining total security of the device. Apples iPhone discourages application interaction, and restricts functionality, resulting in fewer developer options for complex, interconnected application development. Google's Android platform takes a novel approach of declaring permissions, which the user can choose to accept or deny up front. However, the ability to accept some, and deny others, would be a significant advantage.

7.4 External Sensor Pairing and Battery Life

One of the largest limitations we discovered was the sensor set of smartphones. While there are a number of sensors available, there are also a limitless number of sensors not available. Many additional sensors would be useful, such as pressure and temperature sensors for a researcher interested in measuring large-scale climate conditions. While smartphones can be attached to external devices via protocols such as Bluetooth or ANT, these methods drain battery life and are unintuitive with regard to device pairing. We believe that healthy research in more user-friendly pairing protocols, along with continued research in battery life, would allow attaching sensors to a smartphone to be significantly more user-friendly.

7.5 Middleware Allowing Creation of Applications Intended for Multiple Smartphone Operating Systems

While utilization of web services allows minimal device-specific programming, it would be quite beneficial to utilize middleware that would allow a single programming iteration, which could then be compiled and deployed across multiple smartphone operating systems. The Android OS allows a user to write an application matching the Android API, which can then be run on multiple Android-supporting hardware platforms, but no existing solution exists to allow creation of an application in a higher level language, and compilation of the application for multiple platforms, such as the Android and iPhone OS.

With our expertise in middleware research and application, progress in this domain may come from within our organization.

7.6 Sensor Network Applications Oriented Around Inherent Mobility

Previous work on sensor networks has largely focused on geographically fixed areas. In contrast, the mobile nature of smartphones opens up exciting new research possibilities in many new fields. For example, smartphones would be an excellent fit for a sensor network designed to monitor social interactions. Other fields of application include (but are not limited to): highway traffic, marketing, and societal pattern analysis.

8. CONCLUDING REMARKS

The capabilities of mobile devices have increased substantially over the last several years and with platforms, such as Apple's iPhone and Google's Android, will no doubt continue to expand. These platforms have ushered in a new era of applications and have presented developers with a wealth of new opportunities. Unfortunately, with these new opportunities have come new challenges that developers must overcome to make the most of these cutting-edge platforms.

In particular, understanding the limits of mixing mission-critical and enterprise-class services can be complex. Additionally, researchers face a significant challenge in understanding the orthogonal nature of mobile sensor networks relative to spatially fixed networks. While mobile networks present multiple opportunities, it is easy to mistakenly consider them analogous to fixed sensor wireless sensor networks. From our experience creating a web service and smartphone powered wireless sensor network, we learned the following key lessons:

- Utilization and reuse of existing libraries is a fundamental technique for improving network creation time and decreasing network cost. Most smartphones come pre-packaged with multiple libraries. Also, many third-party libraries are available that can either be statically linked against an application, or dynamically called at runtime.
- Combinations of multiple web services and smartphone sensors result in the most flexible application development environment. While web services are extremely powerful, the combination of web services and smartphone sensors can result in powerful application development at an extremely rapid pace.
- Flexibility can also be achieved by creation of a custom web service. In the event of a need for a complex web service, which cannot be achieved by combining various web services and smartphone sensors, a web service can be manually created.
- The mobile nature of smartphones allows fundamentally different sensor networks. This fundamentally mobile network will require significantly different methods of thought to be correctly designed and deployed.

The WreckWatch application is available under the Apache open-source license and can be downloaded at http://vuphone.googlecode.com.

REFERENCES

K. Akkaya, M. Younis (2005). A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks. 3(3) 325-349.*

Android Home. Android. Retrieved Oct. 10, 2009, from http://www.android.com/.

R. Ballagas, M. Rohs, J. Sheridan, J. Borchers. The Smart Phone: A Ubiquitous Input Device. IEEE Pervasive Computing, 5 (1), 2006.

M. Bathula, M. Ramezanali, I. Pradhan, N. Patel, J. Gotschall, and N. Sridhar. A sensor network system for measur- ing traffic in short-term construction work zones. In Proc. of DCOSS '09, pages 216–230, Berlin, Heidelberg, 2009. Springer-Verlag.

Betanews. Press Release. Apple: 13 M iPhones sold in total, 2.6 M Macs sold last quarter. October 2008. http://www.betanews.com/article/Apple-13-M-iPhones-sold-in-total-26-M-Macs-sold-last-quarter/1224624147.

S.L. Burkhard, *Economies of Application Development Programs*. Internet Economics IV. pp.43 Technical Report No. ifi-2009.01, University of Zurich, February 2009.

R. Buyya, C. S. Yeo, and S. Venugopal. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC 2008)*, Dalian, China, Sept. 2008.

C. Fleizach, M. Liljenstam, P. Johansson, G. M. Voelker, and A. Mehes. Can you infect me now?: malware propagation in mobile phone networks. In WORM '07: Proceedings of the 2007 ACM workshop on Recurring malcode, pages 61–68, New York, NY, USA, 2007. ACM.

C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of Network Density on Data Aggregation in Wireless Sensor Networks", In Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS), Vienna, Austria. July, 2002.

U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi. Efficient Data Harvesting in Mobile Sensor Platforms. In *IEEE PerSeNS Workshop*, Pisa, Italy, Mar. 2006.

P. Leijdekkers and V. Gay, "Personal heart monitoring and rehabilitation system using smart phones," in ICMB. IEEE Computer Society, 2006, p. 29. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109 ICMB. 2006.39

K. Lorincz, D. J. Malan, T.R.F Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, S. Moulton (2004). Sensor networks for emergency response: Challenges and opportunities. *IEEE Pervasive Computing, 16-23.*

S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI)*, Dec. 2002.

A. Manjeshwar and D. P. Agrawal. TEEN: A Routing Pro- tocol for Enhanced Efficiency in Wireless Sensor Networks. In *1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Com- puting*, April 2001.

P. Mohan, V. N. Padmanabhan, and R. Ramjee, Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones, In Proc. of ACM SenSys '08, Raleigh, NC, USA, Nov 2008.

L. Nachman, R. Kling. The Intel Mote platform: a Bluetooth-based sensor network for industrial monitoring. *Proceedings of the 4th International symposium on Information processing in sensor networks.* IEEE Press. 2005

R. Rogers, J. Lombardo, Z. Mednieks, and B. Meike. Android Application Development: Programming with the Google SDK. O'Reilly Media, Inc. 2009.

K. Romer, M. Friedemann (2004). The Design Space of Wireless Sensor Networks. *IEEE Wireless Communications, 11(6), 54-61.*

G. Roussos, A.J. March, S. Maglavera: Enabling Pervasive Computing with Smart Phones. IEEE Pervasive Computing pp. 20–27 (2005). April-June.

H. Salem, N Mohamed (2006). Middleware Challenges and Approaches for Wireless Sensor Networks.

IEEE Distributed Systems Online 7(3).

L. Tong, Q. Zhao, S. Adireddy, 2003. Sensor networks with mobile agents. MILCOM 2003—IEEE Military Communications Con- ference, vol. 22, no. 1. Oct 13–16, 2003, Pages. 688–693.

Twitter. (2008). MacWorld. *Twitter*. Retrieved Oct. 10, 2009 from http://blog.twitter.com/2008/01/macworld.html.

Zhang, H. & Hou, J., Maintaining sensing coverage and connectivity in large sensor networks *Ad Hoc & Sensor Wireless Networks,* 2005, *1*, 89-124.