6-1-2013

A Context-based framework for mobile applications

Omkar Kolangade

A Context-based Framework for Mobile Applications

by

Omkar Kolangade

A Thesis Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
Master of Science
in
Computer Science

Supervised by

Dr. Rajendra K. Raj

Department of Computer Science

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York

June 2013

The thesis "A Context-based Framework for Mobile Applications" by Omkar Kolangade has been examined and approved by the following Examination Committee:

Dr. Rajendra K. Raj
Professor
Thesis Committee Chair

Dr. Sumita Mishra
Assistant Professor

Dr. Joe Geigel
Associate Professor

Dedication

To my parents ... For things that cannot be expressed in words ...

Acknowledgments

I would like to thank my committee chair Dr. R. K. Raj for the support and guidance he has given me not only when I was working on my thesis but throughout my Masters program. I would also like to thank my friend Mihir Chitnis for providing me valuable feedback and the required hardware for the completion of this thesis.

Abstract

A Context-based Framework for Mobile Applications

Omkar Kolangade

Supervising Professor: Dr. Rajendra K. Raj

Mobile applications have become commonplace in the past few years. In domains such as healthcare, app user experience becomes even more critical. If healthcare apps are not user friendly, patients simply stop using them, or worse still, they may use these apps with disastrous consequences. One approach to improving app user experience is to keep track of mobile context, which is any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or object.

In this thesis, a system was designed using the following features. Relevant, context-based alerts are provided to the users via the mobile application. The mobile context of the user is tracked completely automatically. Alerts prompt the user to interact with an underlying information system and also improve the user's domain knowledge of the system. A proof of concept was constructed for the healthcare domain. Existing literature shows that user experience of apps is improved by ensuring the existence of the following features: Implicitly sensed contexts, providing the right information at the right time and reduction of user-input requirement. To establish the improvement in the user experience of the system, the features were measured against the research recommendations from existing literature. Based on these measurements and analysis, the hypothesis that the quality of user experience can be enhanced using context is found to be valid.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Medical Information Systems, just like any other information systems are adjusting to the new paradigm that is mobile computing. Traditional desktop-based systems have effectively facilitated the transition from paper-based systems to electronic ones. But they have certain inherent limitations due to the lack of mobility. Mobile computing, however, has changed the scenario by allowing medical entities to access medical information systems whenever, from wherever they are. While this has certainly enhanced the features and capabilities of such systems, certain issues that have plagued medical information systems since their inception, continue to exist.

Lack of awareness and privacy concerns are the main factors hindering the successful adoption and implementation of Medical Information Systems. As a patient, the concerns regarding the extent of personal medical data being available to doctors, nurses, technicians and other entities, is a serious one [4]. Moreover, many medical information systems rely on accurate data input from the patient's side (via logging applications) for analysis. The timing of the input is an important factor in medical data logging. For example, a patient on heavy medication at home might be required to note down symptoms thrice a day at specific intervals.

A context-based patient information system can enable the patients to see the right medical information at the right time and also provides inputs to the patients regarding the

policies and regulations concerning medical data. The system can help the patients in understanding the existing medical information systems clearly and enable them to interact with them as expected by the designers of the medical information systems.

Such a system also provides important contextual health alerts at relevant times. The doctor, while initiating the patient into the system can enable relevant contexts for the patient. For example, a patient undergoing treatment for a gastrointestinal ailment will be reminded of the doctor's advice when they enter a restaurant. It is possible to turn off all the alerts and notifications in the information system, if the user chooses to do so.

1.2 Background

1.2.1 Smart Mobile Devices and Mobile Operating Systems

Context-based mobile applications have been around since the last decade. The importance of contexts other than location has also been discussed [15]. With the development of iOS and Android mobile operating systems and the smart-phones that support them, the creation of context-based applications has become more streamlined. This is mainly due to the APIs provided by the mobile operating systems to access the different types of sensors in the smart devices such as accelerometers, light sensors, GPS co-ordinates etc. Also, these operating systems and smart devices have allowed for the development of complex, purpose-built mobile applications that leverage the connectivity and the processing power of the devices. Coupled with web-services and/or cloud computing, the mobile devices have become the user interaction points of systems akin to their desktop counterparts in the past years.

The transition from traditional desktop systems to smart mobile devices has added new paradigms to computing such as physical portability, ubiquitous network connection and environment sensors. As these devices start replacing desktop computers in information systems, the new paradigms can be used to enhance the systems on the user side. Numerous commercial applications on both the iOS and Android platforms are a testimony to what

can be achieved by using the new computing paradigms.

1.2.2 Context-based Systems

Dey and Abowd define context as 'any information that can be used to characterize the situation of an entity where the entity is a person, place or object that is considered relevant to the interaction between a user and an application' [1]. Systems that take context into account in their user interaction or processing logic can be called context-based systems. Context-based systems have been investigated since more than a decade now [1]. Change in environment is an inherent characteristic of mobile computing and hence context-based systems have been investigated and developed for and around mobile devices. With the advent of smart personal mobile devices, more and more environment sensors have been incorporated into the devices. Moreover, the vendors of these devices have actively developed and opened up the application program interfaces (APIs) to these sensors allowing third-party developers to create context-based applications.

The primary appeal of context-based systems is the advanced automation features the systems can potentially provide. In any form of computing, actions that require the system to take input from the user to determine the solution, direction or action, can be partially or completely eliminated with the use of contexts. This is especially true in mobile computing systems where the inherent portable nature of mobile devices allows a great number of contexts to exist. Moreover, with the advancement in the rapidly-evolving field of mobile computing, new sensors are being integrated into the mobile devices. This has and will continue to create new contexts either on its own or by the combination of the new sensors with the existing ones.

1.2.3 Medical Information Systems

With advance in technology and support from legislation, Medical Information Systems have become a vital part of the healthcare industry. These systems have to not only be secure and robust but they must also comply strictly with the privacy regulations associated

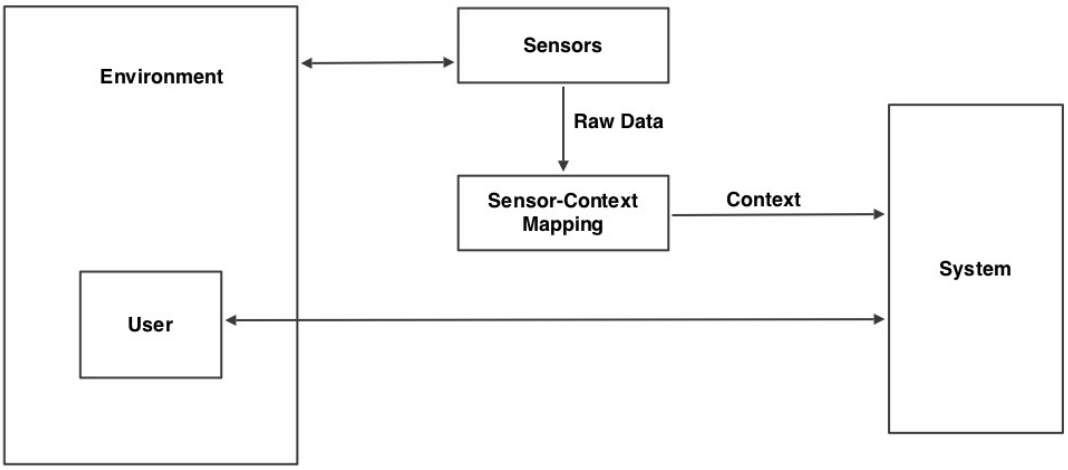

Figure 1.1: A Typical Context-Based System Structure

with Electronic Medical Records such as HIPAA compliance. These systems are typically implemented as client-server systems although there has been research into implementing them in the cloud while maintaining the security constraints [17].

Medical Information Systems are designed and implemented primarily for the creation, storage and management of Electronic Medical Records. Traditionally, all these tasks have been performed by healthcare providers and technicians. After a few years, web portals were developed to give patients direct access to their data using a web browser. With the advent of smart mobile devices like smart-phones and tablets and their widespread use in daily life, these systems now have mobile applications that allow patients to provide and view their personal medical data using their smart mobile devices [11].

1.2.4 User Experience and Privacy in Medical Information Systems

Studies have shown that the developers of Medical Information System software seldom take into account the end users of the system while designing it, which has a direct impact on the user experience aspect of the system [3]. This, in turn, is discouraging healthcare providers and patients from making proper use of the system. Furthermore, in hospitals

Table 1.1: A brief comparison between context-based and non context-based systems

Context-based Systems	**Non context-based Systems**
Systems are aware of one or more aspects of the user's usage environment or scope	Systems require explicit input from user to determine context
Systems are able to modify interaction based on user's context	Systems can modify interactions only on the basis of specific user input
All context-based systems are complex due to the integration and management of multiple, possibly heterogeneous context sensors	Systems may or may not require the integration of hardware sensors, depending upon the purpose of the system
Systems have to implement additional privacy-protection mechanisms	Systems are required to follow standard privacy-protection mechanisms

where the use of Medical Information Systems in mandatory, the improper use of the systems leads to bad data.

Patients are not always aware of the extent to which their medical data is required to be shared with the other entities of the Medical Information Systems and prefer granular access control to their data [4].

1.2.5 Patient Alerts and Information

For time sensitive medical data, it is essential that the patients provide the data inputs at the right time. For example, a patient on heavy medication at home might be required to note down symptoms thrice a day at specific intervals. While simple built-in reminders on the smart devices can remind the patients, the onus is again on the patients to set the reminders. Some medical data is collected at specific times. For example, a patient undergoing treatment for a gastrointestinal ailment might be required to note down every food item consumed throughout the day. To enable reminders in such situations requires the mobile device to detect the appropriate context such as walking into a restaurant in this case.

1.2.6 Mobile Device Sensors

Today's mobile devices have multiple sensors that can be used to determine user context either single-handedly or by a combination of sensor data. Table 1.2 describes the possible usage scenarios for different mobile device sensors.

Table 1.2: Possible mobile device sensor usage scenarios

Sensor	**Scenario**
Global Positioning System	Can be used to obtain sufficiently accurate location information. Can also be used to monitor travel paths or
Wi-Fi	Used to determine user location in both indoor and outdoor environments. Wi-Fi Direct technology also allows for peer-to-peer data transfer.
Accelerometer	Used to determine orientation of the mobile device. Can be used to detect sudden change in acceleration associated with vehicle accidents.
Motion Sensor	Used to identify sudden motion of the mobile device which can further be used to partially detect device failures or medical emergencies.
Light Sensor	Used to determine the environmental lighting in the mobile device's surrounding.
Bluetooth	Used to communicate with known (paired) devices. Can be used to determine location and environment context by identifying neighboring Bluetooth device names and their nature.
Pedometer	Used to measure the number of steps taken by the mobile device uses. This can be used directly in medical information systems. Works with accelerometer in certain cases.
Digital Compass	Used to determine geographic orientation of the mobile device. Can be used with pedometer, GPS sensors for accurate movement tracking and distance calculations.

1.3 Related Work

Context-based medical information systems have been proposed for specific purposes within the medical domain. Fenza et al. [9] describe an integrated environment that provides healthcare services approximately matching the user's context. Their work determines the user context by a robust theoretical approach.

Skov et al. [16] describe a mobile, context-aware hospital ward system for nurses. As nurses move from ward to ward, the context-aware mobile device they carry understands the change in physical location and only displays information related to the patients in that particular ward.

Most of the existing work in the field of context-based medical systems has been associated with medical contexts and specific medical systems as opposed to the proposed context-based information system that focuses on mobile contexts and is expected to work with any underlying medical information system that has a mobile application.

1.4 Problem and Hypothesis

1.4.1 Problem

In a given information system that has mobile applications as user interaction points, it is not always possible to address the user experience and awareness aspects of the systems directly. This may be due to some other constraints the system is bound by.

1.4.2 Hypothesis

The quality of user experience can be enhanced using context.

Chapter 2

Design and Implementation

2.1 Approach

By employing the context-aware nature of smart mobile devices, the user interaction with an existing information system can be improved [1, 2, 6, 7, 10, 13]. The context-aware information system provides relevant information of the existing information system to the user at the right time thereby improving the user's understanding of the system and increasing the chances that the users interact with system as they are supposed to. A context-based information system is designed and a mobile application created to improve the user experience of an existing Medical Information System. A real-world Medical Information System is not used as it is not feasible to get access to the inner workings of such a system. A similar system is implemented to simulate the interaction with the user.
The system is designed to do the following:

- Provide the user with the right medical information at the right location.
- Provide the user with context-based alerts pertaining to the right medical context.
- Provide the user with appropriate policy information while interacting with a healthcare entity.

2.1.1 Advantages of an Independent Solution

While it has been explained why existing medical information systems find it difficult to simply integrate mobile contexts, an independent solution that does this has certain advantages. An independent solution here refers to the information system on top of the medical information system. The proposed system can be implemented over any medical information system that has a mobile application end-point for users. Also, for the system to be most effective, there will have to be a tight coupling of certain features or APIs of the medical information system with the proposed system's mobile application. This enables the system designers to gain a better insight into the workings of multiple, possibly different information systems. As a result, the proposed, context-based information system can continuously evolve and incorporate advanced features and context-based capabilities and in turn prompt the medical information systems to enable or support these features. There might be legal issues with feature replication across different medical information systems; especially if patents are involved in the system design.

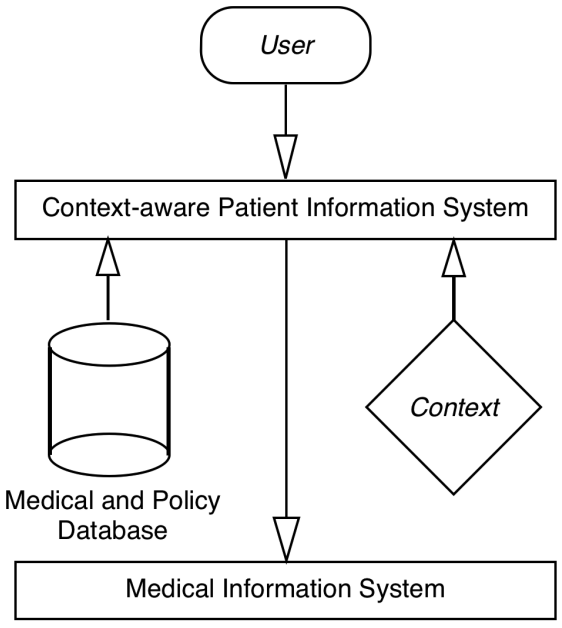

Figure 2.1: Position of the system

2.2 Design

The context-based patient information system adds context-awareness on top of existing Medical Information Systems, thereby improving the user experience of the system. The working of the system is described below followed by the components of the context-based information system.

2.2.1 Mobile Application

The mobile application runs on the patient's smart-phone. It will constantly monitor the mobile context of the patient. As soon as there is a match between the current mobile context and the medical context specified by the doctor, the application will display appropriate information. The server will push the relevant context data to the application. The application will have to run continuously and will require user permission for access to location and other sensors to be effective. It uses the Global Positioning System, Bluetooth and Wi-Fi sensors to determine contexts. The contexts are designed to be used as follows -

Global Positioning System (GPS)

The GPS co-ordinates are the best location indicators on a mobile device. A GPS system has the ability to determine the position of a mobile device accurately within a few meters. However, since the GPS system depends upon communication with GPS satellites, determining position in indoor locations can be difficult.

Bluetooth

Commercial mobile devices have had Bluetooth capabilities since the beginning of the last decade. Since Bluetooth has been designed for short-range mobile usage, it works very well for the purpose of location context determination. Research has been done on this [5] and applications have been using Bluetooth for context determination. Having a particular Bluetooth device in the surroundings will enable a mobile device to determine context specific to that device. For example, a Bluetooth device with the name 'Doctor's Office'

can easily help the mobile device understand that it is likely in a doctor's office. It is worth noting that in such cases, device pairing is not required. This minimizes the initial setup requirements.

Wi-Fi

Wi-Fi networks have become very common for daily use at not only large organizations but also at home. The use of Wi-Fi for determining location in indoor environments has existed since quite some time [12]. This is especially useful when GPS signals are weak or unavailable. Also, just the SSID of the Wi-Fi network can be used to determine the location of the mobile device. A very practical use case scenario for this type of context determination is to partially check whether a user is at their home. Since home Wi-Fi names are typically (but not always) unique.

2.2.2 Doctor's Portal

Doctors access the system by using a web portal. They register patients into the system using the portal and also specify the appropriate medical contexts. The portal is hosted on the central server.

The doctor's portal is a accessed via HTTPS links and has a simple, web-page layout with a navigation menu for the doctor to initialize new patients into the system, remove existing ones and modify medical contexts for patients. The portal has a drop-down list of medical contexts that can also be specified in a text field for the same. The text field has an auto-complete feature for ease of use.

Patient Registration

When the doctor registers a patient in the system, the doctor's portal generates an alphanumeric code. The doctor gives this code to the patient. Upon installing the application and running it for the first time on the mobile device, the application asks the patient for their name and the alphanumeric code. This is a one-time process. The current design of

the system assumes that the patient will use just one mobile device as their primary device.

2.2.3 Central Server

The central server is the main backend entity of the system and is responsible for providing medical and policy information to the patients. All nodes in the system are connected to the central server. The server is accessible over the web for communication with the mobile application. The central server's responsibilities are -

- Hosting the doctor's portal
- Database connectivity
- Communication with the mobile application

In the current version of the system, the central server mainly pushes mobile context information to the mobile application. The doctor can update a patient's information, in case the medical context needs to be changed or the patient is no longer required to be in the system. The updated context(s) are pushed to the mobile application. In case the patient is no longer required to use the application, the application locks and asks the patient to contact their doctor. This is generally expected to be done on the patient's final follow-up visit unless the doctor sees it fit to do so at an earlier stage.

2.2.4 Databases

The system has a medical domain database and a backend database. Two databases are used to improve the security of the system by providing isolated access to medical experts for the medical domain database while maintaining patient records (not Electronic Medical Records) on a separately administered database. Non-relational databases can be considered if the system is to be expanded for context-mining or other large scale data operations. The purpose of each is as follows -

Medical Domain Database

The medical domain database contains a list of medical ailments mapped to the appropriate mobile-contexts. The database is maintained by a team of medical and mobile experts. When a doctor initiates a patient into the system, she specifies appropriate medical contexts for the patient. The appropriate mobile context, for those medical contexts, are sent to the mobile application on the patient's smartphone. This allows the smartphone to trigger alerts only on the specified events. This design saves battery life by simply ignoring non-specified mobile context.

The medical domain database also contains policy information pertaining to specific roles in the medical domain. The sources of the information are the laws and the directives specified by the government.

Backend Database

The backend database stores the records for mobile contexts that are bound to patients and the doctor's customized notes provided while initiating the patient into the system. As of now, that is the only purpose of the backend database. If the system is to include additional features such as mobile context mining and logging, this database cab be used to provide the functionality. It is important to note that the database does not store the medical contexts bound to a patient. This prevents the database from being a source of medical information leaks. The backend database also maintains the system activity log records.

Mobile Context - Medical Context Binding

When the doctors are initiating a patient in to the system, they provide relevant medical contexts and specific instructions to the central server. The system does not expect the doctors to be aware of the mobile contexts. The mapping between the mobile and medical contexts is done by the central server by connecting to the Medical Domain Database. It is worth noting, however, that is the doctors are aware of at least the basic workings of the system, they would specify medical contexts in such a way that the system has the best

possible understanding of the patient's medical condition and would hence know which mobile contexts, or their combinations, need to be monitored on the mobile device.

2.2.5 System Nodes

A Bluetooth 4.0 Low Energy (BLE) based system node is placed at all the physical locations of the entities of the Medical Information System. For example, a typical Medical Information System has entities like doctors, nurses and technicians. Nodes are therefore be placed in the doctor's office, technician's office and near the nurse's station. The nodes periodically check for a paired device. Every node in the system is connected to the central server.

Since the mobile application only requires the Bluetooth device names around it to determine the context, pairing is not required. The systems assumes that Bluetooth nodes corresponding to different medical entities in a hospital are placed at a sufficient distance from each other so as to avoid a range overlap.

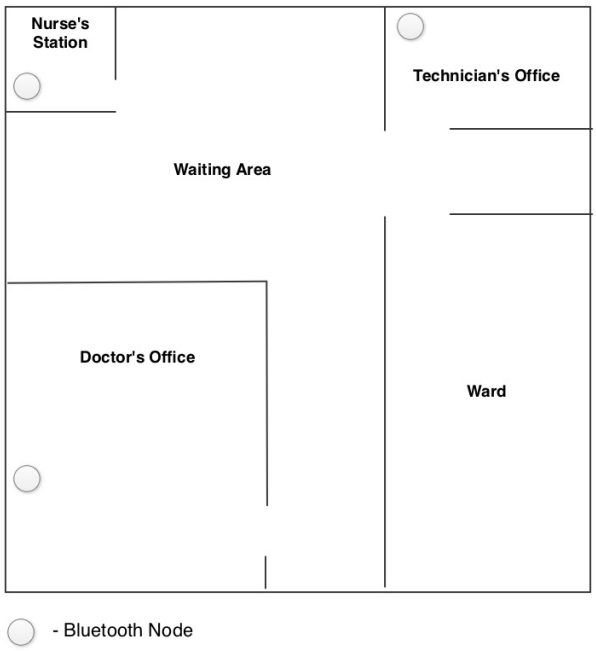

Figure 2.2: Possible Bluetooth node placement in a small hospital

2.3 System Flow

When a patient visits a doctor for the first time, the doctor will register the patient into the context-based information system and ask them to download and install the mobile application. The doctor has a web-based portal for the patient-registration procedure. At the same time, the doctor will enable the relevant medical contexts for the user. For example, for a patient with a gastrointestinal ailment, the doctor will enable the appropriate contexts.

The server will be connected to a database of appropriate mobile context mappings to the medical contexts. In the above example, as the doctor specifies the gastrointestinal ailment and adds specific comments, the server will query the database to learn that the mobile context 'restaurants' needs to be activated for the patient's mobile device. This will result in the mobile application on the patient's smart-phone giving an alert to the patient as they walk into a restaurant. The alert will contain the doctor's specific comments. If the patient is required to enter their meal details into a Medical Information System, they will be prompted to do so as they walk out of the restaurant.

Whenever the patient interacts with a doctor, nurse or a technician, relevant medical information will be shown to the patient. This information will be in the central server's database. In the above example, when the patient goes to a technician for a blood or stool test, information regarding why the test is being carried out and the most important test result will be shown to the patient by the mobile application.

Also, the mobile application will contact the server to obtain the appropriate privacy policies concerning Electronic Medical Records. In the above example, the application will inform the patient that they are expected to make certain information available to the technician such as name, blood group etc. This way, when the patient is expected to interact with the Medical Information System, they will be completely aware of the system's expectations.

The above scenarios will work because the application will understand that it is in the presence of a node in the technician's office and will also have the medical context specified by the doctor.

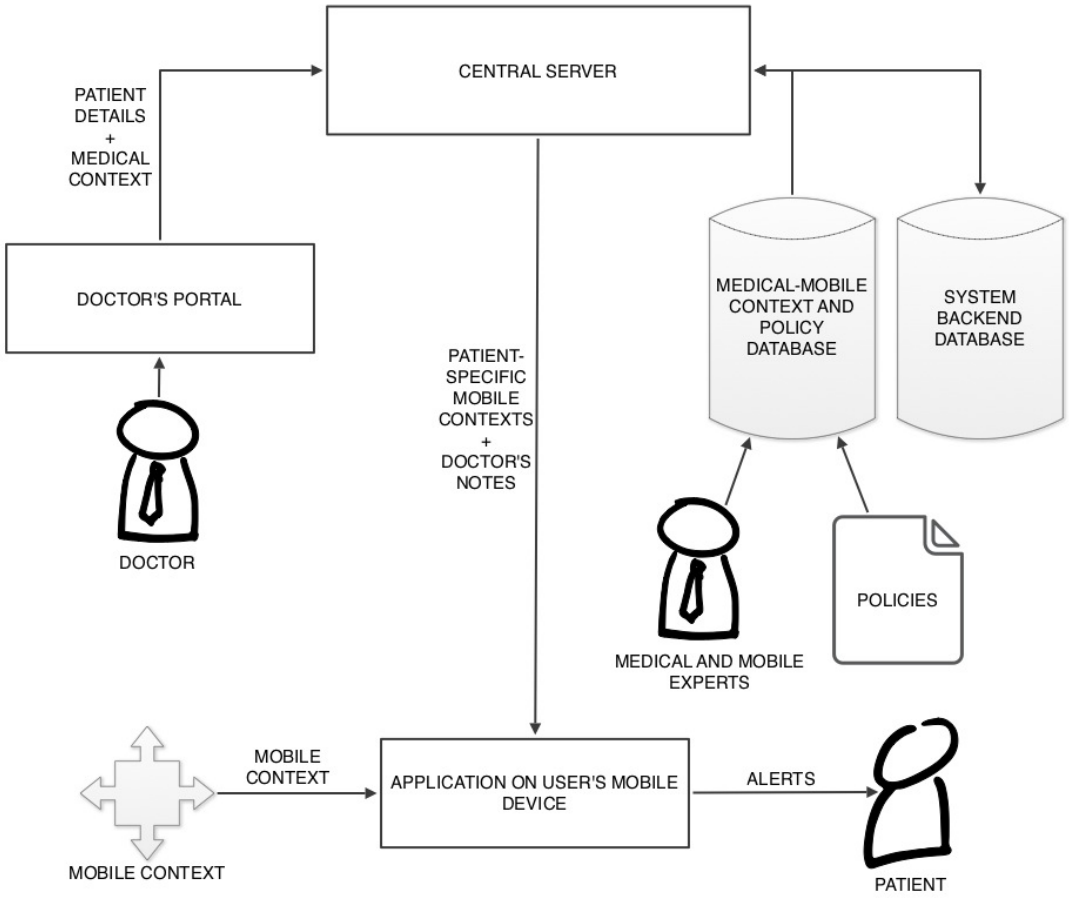

Figure 2.3: Context Based Information System Structure

2.4 Comparison With Existing Work

Existing context-based medical information systems are generally application specific. The use of mobile contexts is for a particular scenario such as using location to determine what information to present [16]. It is important to observe that the context based system proposed is not dependent on any specific mobile context and can even be expanded to accommodate new mobile contexts as new technology becomes available. Moreover, since the system is an independently administered one, it can be used with multiple, possibly proprietary, medical information systems and can be customized to match the technology, terminology and the features of the underlying system.

2.5 Prototype Mobile App Implementation

The mobile application is implemented on an HTC smart-phone running the Jelly Bean (4.0) version of the Android operating system. The communication with the server is simulated by using hard-coded variables and the nodes are simulated on one or more computers since the focus of the implementation is not on implementing a web service or establishing network connections. Geo-location, Bluetooth and Wi-Fi network based contexts are implemented and drive alert events in the application. The user interface elements of the prototype application are sufficient enough to display the relevant information. A production enterprise application will need to implement a more appropriate a user interface. It is important to note that the prototype application or even the production application for that matter will require the user to allow access to all the required sensors. The privacy and security concerns arising are most likely to prevent general distribution of the application of commercial application stores. The application will have to be classified as an enterprise application to be approved for limited distribution.

The server simulation involves providing appropriate mobile contexts to the application. This is done by manipulating pre-defined variables in the application's code. Node simulation involves a Bluetooth client in the proximity of the smart-phone. The client can be any Bluetooth-enabled device. Both context-based and policy alerts are implemented.

The mobile application developed as a part of this thesis is a prototype application to show how existing mobile devices can be used to run the system's mobile application, at a very basic level. The focus of the prototype implementation is the use of existing sensors in a mobile device to trigger and display appropriate alerts. Communication with the central server is simulated by hard coded variables. Web-based, client-server communication is not the focus of this prototype implementation. Details and guidelines for developing a full-fledged application are given in a later part of the chapter.

2.5.1 Android Operating System

The Android operating system has evolved from the Android Project which was taken over by Google shortly after its inception. Android is a Linux kernel based operating system which supports most, but not all, the features of the Linux kernel version 3.0. It has been developed by the Open Handset Alliance which is a consortium of mobile or mobile-related companies. Android is specifically developed for running on mobile phones.

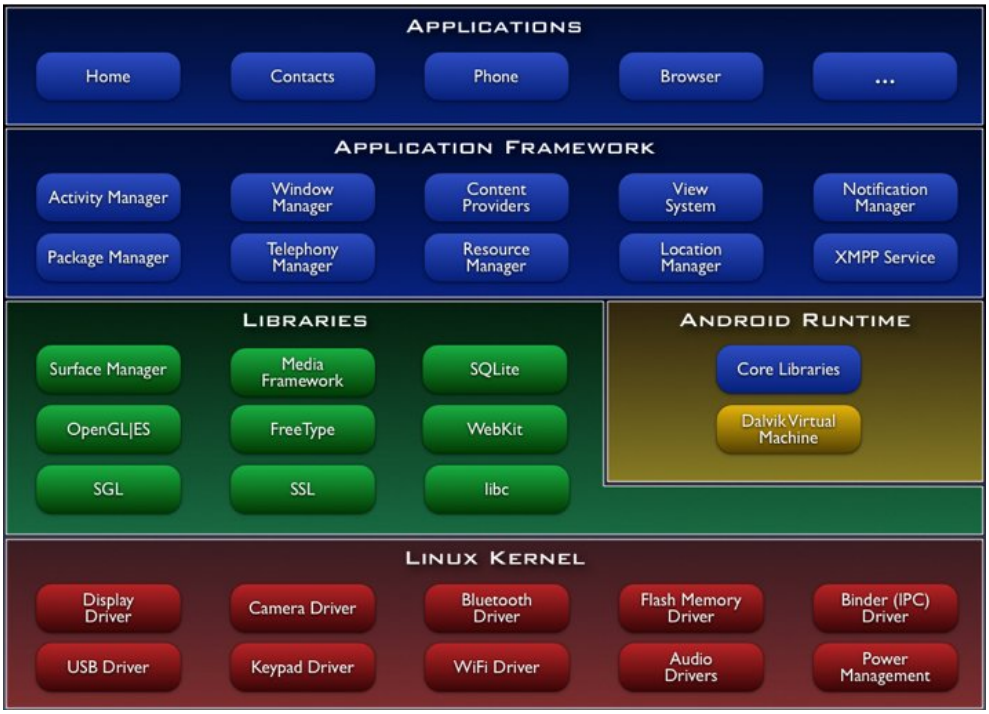

Figure 2.4: Android Operating System Architecture (Source: Google)

As figure 2.4 illustrates, the Android operating system is made up of multiple layers with different levels of abstraction. The application for the proposed context-based information system utilizes the communication and sensor libraries to obtain context. The libraries provide simplified API calls to the underlying hardware sensors and also implement necessary auxiliary functions such has security and reliability. It is worth noting that a specific context-based library can be created for Android to support the context-based information system. This is explained in the Future Work section of the report. Android

also has wrapper libraries for some services such as location service that use and manage multiple sensors to determine a user's location and detect changes as well.

One of the main advantages of Android over other mobile operating systems is that it is open source. This allows us to customize components of the operating system system itself, if required. While the system's mobile application does not require any operating system customization, the option might be useful if purpose-built hardware devices are used instead of general purpose, commercial mobile devices.

2.5.2 Application Design on Android Operating System

The Android application is implemented as a multi-tab application. The main activity is responsible for the user-interface elements while each tab has it's own thread for executing tasks. This design conforms to Google's design guidelines for Android applications. The design allows for asynchronous process execution which is particularly useful for this application as both the context-based alerts module and the policy awareness module can coexist and run independently. Also, the multi-tab design allows the context-based alerts module to continue to present alerts even if the patient is in the vicinity of a Bluetooth node and receiving policy-based alerts.

The GPS-based location in the application is managed by the *LocationManager* class. It periodically monitors for a change in the user's location and can trigger specific application methods based on the user's location. The prototype application just uses the location update functionality. The *LocationManager* class has a method called *requestLocationUpdates()* that allows a *LocationListener* object to be bound to the *LocationManager* object. The *LocationListener* object must implement four callback methods that are invoked by changes in a user's location or the location sensor. The prototype application overrides the *onLocationChanged(Location location)* method to implement the logic for triggering appropriate alerts based on the user's updated location. Precise latitude and longitude coordinates allow for the implementation of geo-fences. This is especially useful if an existing location service or API that provides a predefined list of Points-Of-Interest (POIs) cannot

be used due to design or policy restrictions.

The prototype application uses Wi-Fi network name (if available) to aid in determining the location of the device. The *WifiManager* class has a *getConnectionInfo()* method which provides complete connection details for the current Wi-Fi network. The application just makes use of the Service Set Identifier (SSID) of the network and compares it with a known Wi-Fi network SSID.

The Bluetooth chip in the device is used by working with the the *BluetoothAdapter* class. This Android class has all the necessary methods a developer would require to discover, pair and use the Bluetooth on an Android device. The *startDiscovery()* method scans the device's surroundings for Bluetooth devices. After scanning for approximately twelve seconds, it then does a page scan on each of the discovered devices to obtain the device name. This method is asynchronous. To trigger events based on the presence of certain Bluetooth devices, we use the *BroadcastReceiver* class. An object of this class listens for events (called 'actions' in Android terminology) throughout the system. When a Bluetooth device discovery happens, the *BroadcastReceiver* object's *onReceive()* method receives an event which contains device details. This callback method is the place where the Bluetooth device-specific contextual logic is implemented.

2.6 Production System Implementation Guidelines

2.6.1 Central Server

The system's server side can be implemented by a central-server or a distributed system design. The later design is recommended only when the usage of the system becomes large enough to justify the scalability-complexity trade-off. If the system is expected to be deployed across a broad geographical region or even worldwide, a cloud-based server is recommended.

The server can be implemented using any off-the-shelf server engine technology such as NGINX [14] or Node.js [8] to name a few. Security mechanisms, database I/O optimization, encryption and other system enhancements can be implemented as per the system administrator's requirements. The data format for transfer between the doctor's portal and the mobile application can be either XML or JSON. JSON is recommended as both the Android and iOS mobile operating systems have stable, ready-to-use JSON parsing libraries. A possible server-side stack can look like this -

- Amazon AWS
- Node.js
- Express.js
- MongoDB or CouchDB

Both Node.js and NGINX server-side technologies have robust Javascript callback mechanisms that are well-suited to handle multiple, asynchronous requests over the Internet. Express.js can be used to power the doctor's portal and other web-based features of the system should the requirements change in that direction.

2.6.2 Databases

The database implementation remains largely open-ended. Both relational and non-relational databases can be used. One of the advantages of using a non-relational database would be

the relatively simple integration with a new server technology such as Node.js. One of the key design issues is indexing as in the deployed system, the number of retrievals for the Medical-Policy database will be significantly greater than the number of insertions. Traditional relational databases have tried-and-tested indexing but newer non-relational databases such as MongoDB also support indexing. Schema design considerations should focus on how to efficiently map medical contexts to possibly multiple mobile contexts.

2.6.3 Mobile Application

The mobile application should have multiple views, regardless of the operating system it is built on. This is to provide a clear distinction between the context-based and the policy-based alerts in the system. Typically, both Android and iOS mobile operating systems advocate and implement multi-threaded applications. The main thread of the application should handle the user interface elements of the application only.

While using the Bluetooth APIs, care must be taken to actively manage the device scanning process as this process tends to affect battery life significantly. Similarly, event listeners corresponding to different sensors in the device should be handled in all the life-cycle methods of the application. This enables or prevents the sensors from being active and reporting data when the application is 'minimized' or interrupted by other device functions. Care must be taken to not store any contextual information, obtained through sensors, on the device. The only contextual information already present in the application should be the information obtained from the central server. Sensor data logging, while appealing as it can allow for creative expansion of the system, is a serious privacy issue.

Since the mobile application mainly focuses on alerts, it is very important to utilize the notification system of the mobile operating system it is deployed on. Android and iOS have notification systems that allow the application running in the background to deliver 'pop-up' style notifications. In fact, the notification systems typically work with a web server as well. This allows certain notifications to be sent directly to the mobile device even when the application is not running on the device.

To store the information obtained from the central server during the setup phase, the application can use the operating system's internal storage. Android's internal storage can be accessed by the *openFileOutput()* method and iOS has an *NSFileManager* class for the same. The information storage is persistent across multiple application runs but not across multiple re-installations. Such persistence can be achieved by implementing additional caching logic on the server-side.

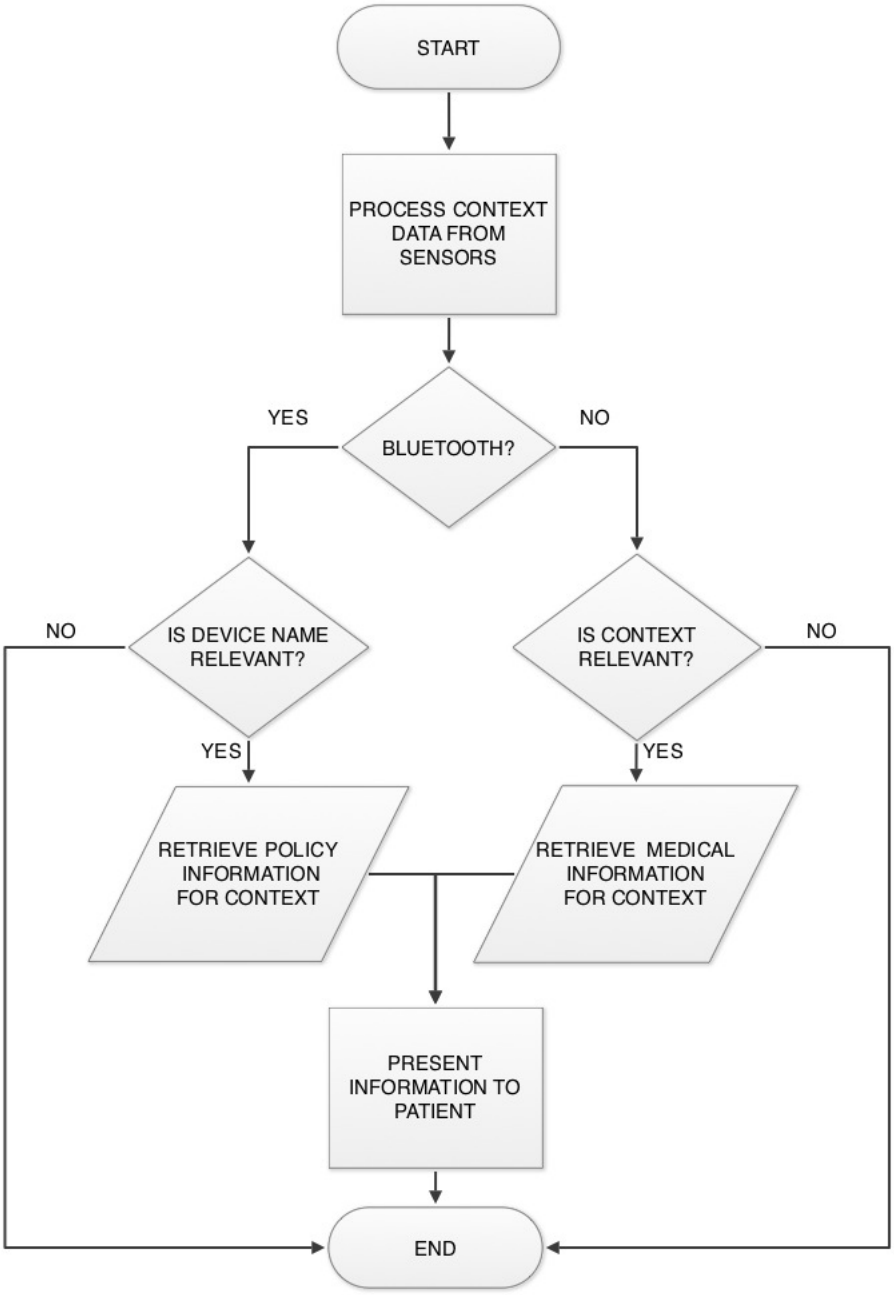

Figure 2.5: Patient Alert System Flow Chart

Chapter 3

Analysis

3.1 Approach

Since the hypothesis concerns the feasibility of building a context-based information system and a mobile application to improve user experience and awareness, we first understand the extensive research that has been done by Human Computer Interaction and user experience experts to understand the approaches that are recommended by them. Since user experience is a human interaction metric we cannot obtain specific numerical metrics for analyzing the success or failure of the proposed system. Analyzing the user experience aspect of systems is not a straightforward process and many approaches have been proposed. Moreover, user experience studies typically require carrying out interaction tests on specific focus groups. In this case, the analysis would require forming focus groups of doctors and patients using a common medical information system and its mobile application, which was not feasible given the scope of the work. Instead, we look at the research done by experts in the fields of mobile user experience and context-based systems that has carried out human interaction experiments and proposed solutions by analyzing complex user experience metrics.

3.2 Analysis

User experience studies and context-based system research has focused on evaluating systems that could potentially incorporate context-awareness to improve user experience of the system. But as Figure 2.1 illustrates, the context-based information system works alongside an existing information system. However, to the end user, the features and the benefits of the context-based system appear to be a part of the same underlying information system.

Table 3.1 summarizes the design features of the proposed system based on experimental and theoretical research. Explanations of the recommendations and their impact on system design follow.

Table 3.1: Research recommendations and corresponding system design features

Research Recommendations	**System Design**
Users have much to gain from implicitly sensed contexts in mobile applications [1].	The mobile application senses contexts continuously.
Providing right information to the user at the right time [10].	The system generates and displays alerts to the user only when the mobile context matches the pre-defined medical context. Only that information which corresponds to the medical context is shown to the user.
Context-awareness reduces the requirement for user input [13].	The system automatically takes into account the location, time and other mobile contexts. This process is continuous and requires no user input or control.
Context-awareness reduces the complexity of interactive systems [6].	The system prompts the user to input information into the underlying medical information system.

3.2.1 Implicit Context Sensing

The mobile application in the context-based information system keeps monitoring device sensors for contextual information. This happens even when the application is not active on the mobile device. Once the user installs the application and provides appropriate details, the application does not require any user input.

3.2.2 Providing Right Information at the Right Time

The mobile application alerts are triggered only when the mobile context of the user and the medical context match. Also, policy based alerts are triggered only when the user is in the proximity of a Bluetooth node. The use of multiple contexts is required to determine the right time for the alert.

3.2.3 Context-awareness Reduces User Input Requirement

Since the system is designed to work on top of an existing medical information system, the contextual information generated by the system can be used in the medical information system. This, however, may depend upon the design and restrictions of the underlying medical information system. Certain cases are easily possible. For example, if a medical information system mobile application requires the patient to track the approximate distance they walked each day, an API access to the context-based information system can update or log this information into the medical application directly.

3.2.4 Context-awareness Reduces the Complexity of Interactive Systems

The context-aware information system keeps a track of the contexts and processes the contextual information to generate relevant information. This information can then be directly used by the user. Also, the alerts in the system are only triggered when there is an appropriate context match. This allows the user to rely on the mobile application for time or location based interaction with the medical information system's application.

3.3 Hypothesis Evaluation

Since the design of the proposed context-based information system follows research recommendations on improving the user experience of information systems, the results of the analysis are promising. The mobile application is a vital part of the underlying information system and the proposed system uses mobile context to enhance its user experience. The nature of the context-based information is such that with tight integration with the underlying information system, the systems will effectively appear as one to the end user.

In theory, it can be said that the system clearly enhances the user experience by using context. However, the scope of this thesis did not allow for the building of a complete prototype system to conduct user experience tests. For a better and comprehensive analysis of the hypothesis, a prototype system needs to be deployed on top of a participating information system. Also, user experience tests need to be carried out on the information system on its own to obtain the user experience metric; this metric will be used for comparison with the metric obtained from the usability tests carried out on the system with the addition of the context-based information system.

A user experience test for the system would essentially focus on a group of patients who use the same underlying Medical Information System's mobile application. This would typically require the patients to be a part of the same or the same group of hospitals. A key aspect of system evaluation would be to analyze the data collected by the underlying Medical Information System. This is because if the context-based system worked as expected, the patients would provide better data due to the context-based alerts. The backend database of the context-based system would also need to track the number of times different contexts were triggered in the system. Proper privacy disclosures would be required for such context logging. At the end of the test period, a survey of the participants, paper-based and in the form of interviews would be required to determine the overall impact of the context-based system.

Chapter 4

Conclusions

4.1 Current Status

The context-aware patient information system described can be implemented on any current medical information system. The system proposed is flexible as specific implementation details and the addition of security features is left to the system administrator. The current implementation is for the Android mobile operating system only.

4.1.1 Limitations

The proposed system has been designed to primarily enhance the user experience and awareness of an underlying information system. While doing so, many aspects of software design like security and portability were not the prime area of focus. Some of the key limitations of the system in its current state are -

Limited Security Mechanisms

The system has not been specifically designed to hold against network or database attacks. Components of the system rely on standard, 'off-the-shelf' security mechanisms such as HTTPS and the Android operating system's built-in application security and privacy mechanisms.

Platform Dependence

The mobile application has been developed for the Android mobile operating system only. While a similar application can be developed for the other popular operating systems such as iOS and Blackberry OS 10, it would still require the availability of equivalent mobile device sensor APIs for true cross-platform homogeneity of the mobile application. Also, the platform's vendor platform must be able to classify the system's application as an enterprise application for controlled deployment. Due to the application's requirement of access to as many sensors as possible, the application is not likely to be approved on the commercial 'app stores'.

Dependence on Sensor Availability

Since the system is primarily context-based, lack of context availability effectively disables the system. While the system has been designed to use multiple sensors (wherever possible) to determine contexts such as location, the possibility of multiple sensors being unavailable is not an unlikely one. The lack of a particular sensor can have variable impact on the system depending on what other sensors are available. One way of preventing the application from working in a semi-correct state is to specify mobile device requirements beforehand and perform the appropriate sensor checks from within the application on the first instance of the user running it.

4.1.2 System Design to Prototype

Since the proposed system is based on user experience research literature recommendations and analysis, it needs to be deployed and tested. One of the major challenges of deploying even a prototype version of the system is the availability of an underlying medical information system. Since most medical information systems (or their mobile applications) are proprietary, achieving system-specific or tight coupling of context-based features is a big challenge. However, dummy information systems can be used to test the context-based

information system. A prototype-level or open mobile application of the underlying information system is necessary for thorough testing.

4.2 Future Work

The information system described in this work is based on the context-aware nature of today's most popular mobile operating systems and the devices they power. In the current system, even if a patient is reminded to make a log entry into an EMR system, they still have to do it themselves. A tight coupling between medical information systems and a context-based Information System can lead to features like auto-logging of relevant data.

4.2.1 Additional Sensors

Wearable medical sensors worn by the patients can be designed to work with an information system. Such a system has great potential in remote patient healthcare. The medical sensors can provide specific data to the mobile device directly or to a dedicated analog signal receiver. The data can then be interpreted as another context dimension. For example, a patient who is recovering from a recent heart attack may be advised to do light cardiovascular exercises as part of the recovery process. The current system will watch for contexts like location in a park to display the doctors advice and information. If the patient has a pacemaker installed and it is able to communicate with the information system, then the mobile application will be able to enhance its patient-warning mechanism by keeping a track of the heartbeat rate over a time interval or monitor it for a certain doctor-specified threshold. Any type of sensor or information system that can connect to or work with this system will effectively provide additional contexts which can make the information and alerts even more relevant and accurate than they are in the current system.

4.2.2 Cross-platform Service

The mobile application can be implemented on other mobile operating systems such as iOS and Blackberry OS 10. Mobile operating system vendors actively make new context-based sensor APIs available for developers to encourage the development of context-based applications. New technologies such as Google Glass, Near Field Communication (NFC) can be used to provide mobile contexts. Also, the context-based alert system could be implemented as an Android service instead of an application. This would allow existing application to use the context-based alerts. However, the user interface for the contexts and alerts will have to be provided by the application.

4.2.3 Generic Application of the Context-based System

Apart from the medical domain, such a context based information system can be implemented for commercial information systems such as news alert or tourist information systems. The users can themselves provide relevant contexts that will be enabled on the mobile application allowing alerts to be triggered for relevant contexts and Point-Of-Interests (POIs). The important point to note here is that the underlying system must have a mobile application for the system to work.

If the underlying information system is not bound by stringent privacy-protection standards and other relevant legal policies, the system can natively implement one or more features of the context-based information system.

4.2.4 Context Management Mobile Library

As we can see in figure 2.5, the Android operating system has libraries that provide abstraction and APIs to mobile applications for using and controlling the underlying hardware elements. During the deployment of the context-based information system, the developer will certainly observe some specific patterns in which the sensors are used to obtain mobile

context. This pattern can be compiled into an API call accessible directly to the application. Not that context here refers to mobile context and not the Context class in the android.content package in Android.

For example - A mobile application wants to check all its neighboring Bluetooth devices to determine if there are any known ones. This is to guage the familiarity of the environment the user is in. Currently, this task can be easily accomplished using the Android Bluetooth API. But if such functionality could be implemented by a single API method such as "isBluetoothEnvironmentKnown()", it would certainly make it easier and encourage developers to incorporate mobile context into their applications. A collection of such methods for different sensors to obtain the user's mobile context can be the crux of a context management mobile library.

4.3 Conclusion

Due to the legal and technical constraints faced in modifying existing medical information systems, the approach of deploying a context-based information on top of the system to enhance user experience is an appropriate one. Along with user experience, the system also improves the awareness aspect of the underlying medical information system by providing relevant, context-based policy alerts. The context-based information system can be modified to appropriately support the underlying information system; a tourist information system is a good example of such an information system. Due to the constant development in mobile devices and telecommunication infrastructure, a mobile context-based system is powerful and flexible enough to improve the user experience of another information system.

Bibliography

[1] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, HUC '99, pages 304–307, London, UK, UK, 1999. Springer-Verlag.

[2] Leon Barnard, Ji Soo Yi, Julie A. Jacko, and Andrew Sears. Capturing the effects of context on human performance in mobile computing systems. *Personal Ubiquitous Comput.*, 11(2):81–96, January 2007.

[3] Howard L. Bleich and Warner V. Slack. Reflections on electronic medical records: When doctors will use them and when they will not. *International Journal of Medical Informatics*, 79(1):1 – 4, 2010.

[4] Kelly Caine and Rima Hanania. Patients want granular privacy control over health information in electronic medical records. *Journal of the American Medical Informatics Association*, 20(1):7–15, January 2013.

[5] Alvin T. S. Chan, Hong Va Leong, Joseph Chan, Alan Hon, Larry Lau, and Leo Li. Bluepoint: a bluetooth-based architecture for location-positioning services. In *Proceedings of the 2003 ACM symposium on Applied computing*, SAC '03, pages 990–995, New York, NY, USA, 2003. ACM.

[6] Keith Cheverst, Keith Mitchell, and Nigel Davies. Exploring context-aware information push. *Personal Ubiquitous Comput.*, 6(4):276–281, January 2002.

[7] Constantinos K. Coursaris and Dan J. Kim. A meta-analytical review of empirical mobile usability studies. *J. Usability Studies*, 6(3):11:117–11:171, May 2011.

[8] Ryan Dahl. Node.js, 2009. [Online; accessed 23-May-2013].

[9] G. Fenza, D. Furno, and V. Loia. Hybrid approach for context-aware service discovery in healthcare domain. *J. Comput. Syst. Sci.*, 78(4):1232–1247, July 2012.

[10] Gerhard Fischer. Context-aware systems: the 'right' information, at the 'right' time, in the 'right' place, in the 'right' way, to the 'right' person. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 287–294, New York, NY, USA, 2012. ACM.

[11] Hyrax Inc. My Medical for iPhone/iPad, 2009. [Online; accessed 20-April-2013].

[12] S. Johnson. A framework for mobile context-aware applications. *BT Technology Journal*, 25(2):106–111, 2007. Copyright - Springer Science+Business Media, Inc. 2007; Document feature - ; Diagrams; Maps; Photographs; Last updated - 2012-07-24; DOI - 1295553401; 36213271; 53283; BTTJ; SPVLBTTJ105502533.

[13] S. Kurkovsky. Using principles of pervasive computing to design m-commerce applications. In *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on*, volume 2, pages 59–64 Vol. 2, 2005.

[14] NGINX. NGINX, 2011. [Online; accessed 23-May-2013].

[15] Albrecht Schmidt, Michael Beigl, and Hans w. Gellersen. There is more to context than location. *Computers and Graphics*, 23:893–901, 1998.

[16] B. Skov and Th. Hoegh. Supporting information access in a hospital ward by a context-aware mobile electronic patient record. *Personal Ubiquitous Comput.*, 10(4):205–214, March 2006.

[17] Rui Zhang and Ling Liu. Security models and requirements for healthcare application clouds. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, CLOUD '10, pages 268–275, Washington, DC, USA, 2010. IEEE Computer Society.

Appendix A

Prototype Application

A.1 Source Code

The complete Android application (.apk file) and it's source code can be found in the attached disc.

A.2 Screenshots

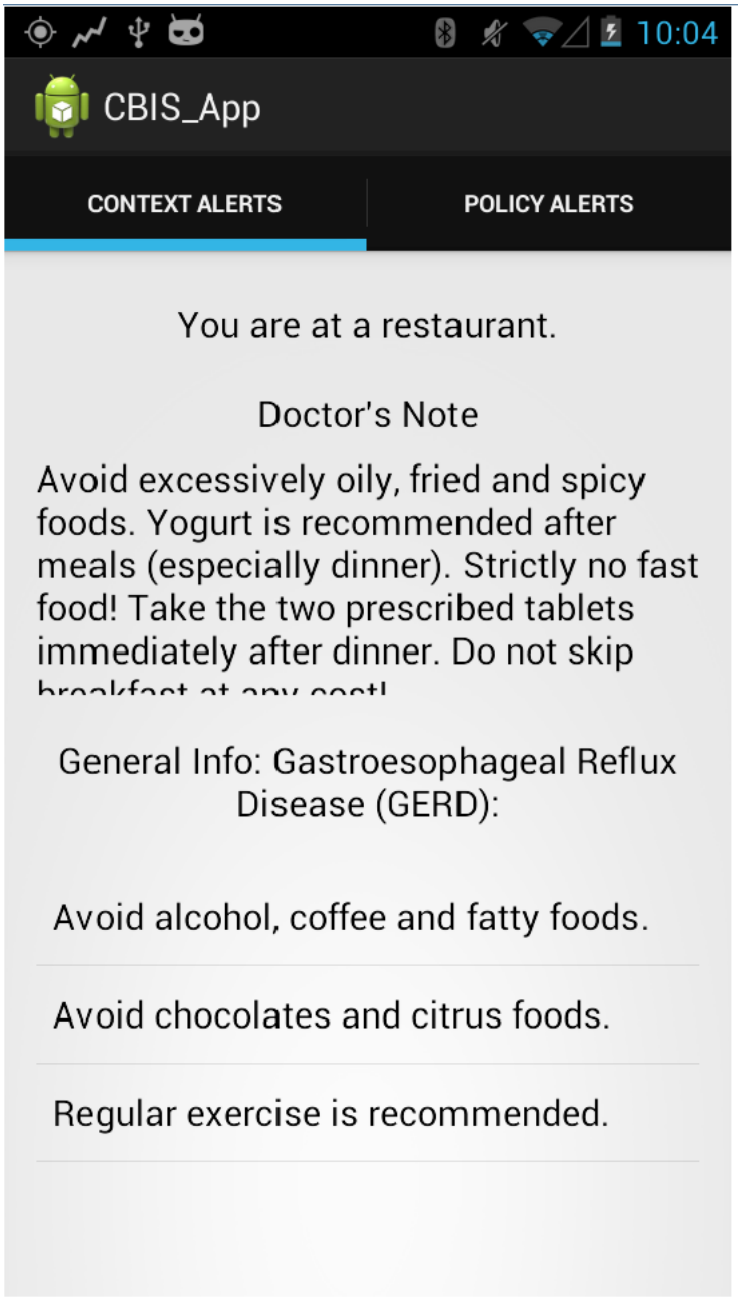

Figure A.1: Health Alerts

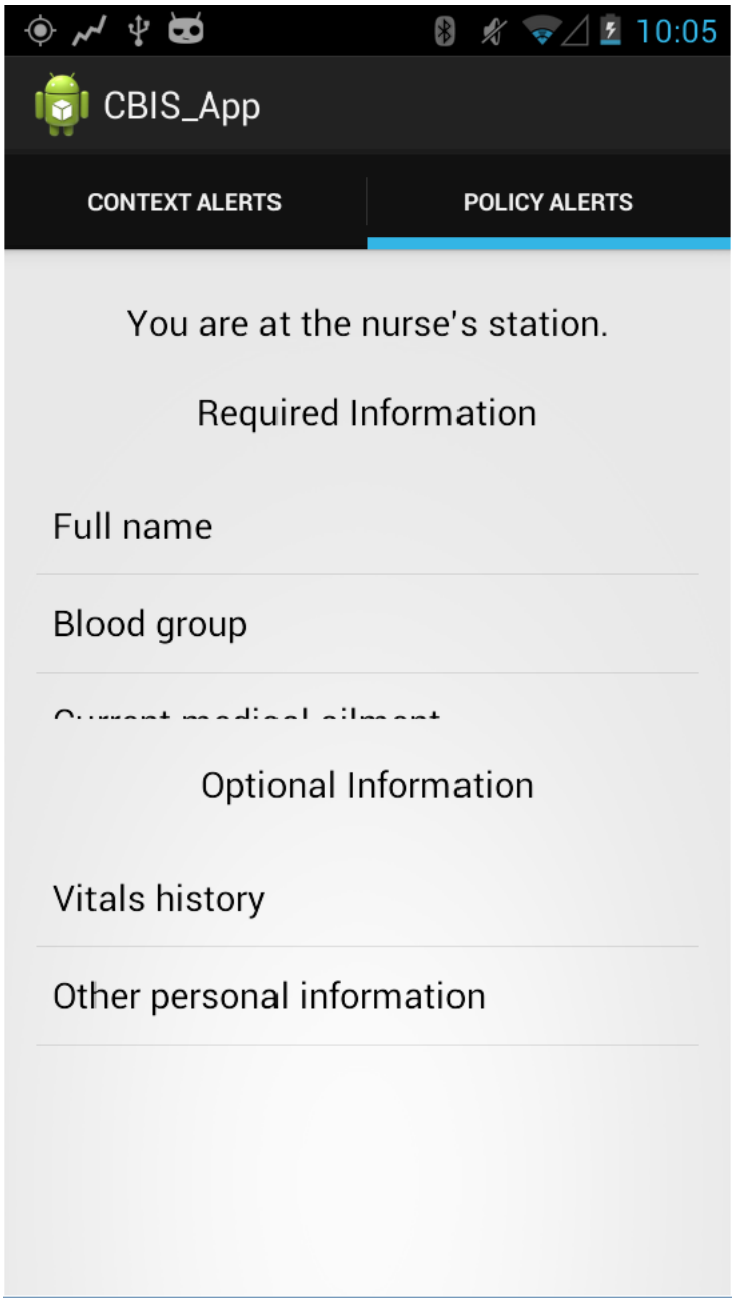

Figure A.2: Policy Alerts