

Declaration

The dissertation is my original work and has not been submitted previously for a degree at this or any other university/institute. To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Students name: Thisara Rupasinghe Date: 22/01/2012

Signature:

This is to certify that this dissertation is based on the work of Mr. Thisara Rupasinghe under my supervision. The dissertation has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor name: Dr. Prasad Wimalaratne Date: 22/01/2012

Signature:

Abstract

Over the last couple of years wireless communication has become of such fundamental importance that a world without it is no longer imaginable for many of us. Beyond the established technologies such as mobile devices and WLAN, new approaches to wireless communication are emerging; one of them are called ad hoc sensor networks.

Ad hoc and sensor networks are formed by autonomous devices called nodes communicating via radio without any additional backbone infrastructure. If two nodes are not within mutual transmission range, they communicate through intermediate nodes relaying their message, that is the communication infrastructure is provided by the devices themselves. In view of the great potential of ad hoc and sensor networks in a variety of application scenarios such as disaster relief, monitoring and surveillance, or data gathering, it is not surprising that there has recently been a flurry of research activity in the field.

Because of high initial cost, nodes power issues and deployment overhead traditional sensor networks are not practical on some situations. And since today's mobile devices are equipped with different kind of sensors and expensive hardware they can be used in place of sensor network. On the other hand most of the mobile devices are underutilized and idle on most of the time. Therefore using this application, the mobile device can be converted into a sensor node without disturbing user activities. Google Android becomes the target platform for this application, since it is the most popular and widely used mobile device platform today.

This application is currently used only the magnetometer sensor. But it is expandable to work with all other sensors on mobile devices. Once a user of this sensor network request magnetometer reading of specific location this application send the request to the centralized server. Then it finds the most suitable mobile nodes to do this job and forward that request to those selected nodes. Mobile node collects that requested data and uploads them periodically to the centralized server where user have access to them. Since mobile devices are spread all over the world it resolves the deployment difficulties. And usually mobile devices come with enough power sources which also has an Internet connection which is used to connect to centralized server. This resolves most of the common problems inherited with traditional sensor networks.

Table of Contents

1 Chapter : Introduction.....	1
1.1 Motivation.....	2
1.2 Problem Definition.....	3
1.3 Aims and Objectives.....	3
1.4 Scope.....	4
1.5 Overview.....	5
2 Chapter : Background.....	6
2.1 Literature Review.....	8
2.1.1 Open wireless sensor network telemetry platform for mobile phones.....	9
2.1.2 Sensor network to Android platform to control Rovio Robot.....	10
2.1.3 Location and Mobility in a Sensor Network of Mobile Phones.....	10
3 Chapter : Requirement Analysis.....	11
3.1 Functional Requirements.....	12
3.1.1 User Login.....	14
3.1.2 Add Job.....	15
3.1.3 View/Edit Job.....	16
3.1.4 Synchronize.....	16
3.1.5 Record Service.....	17
3.1.6 View Data.....	18
3.1.7 Email Data.....	18
3.1.8 User Ranking Process.....	18
3.1.9 Server Side Implementation.....	19
3.1.10 Exception Handling.....	19
3.1.11 Service Level Authentication.....	19
3.2 Non-Functional Requirements.....	19
4 Chapter : Design and Implementation.....	21
4.1 Proposed Solution.....	21
4.2 Design Overview.....	22
4.3 Front End Technologies.....	23
4.3.1 Suitable mobile platform.....	23
4.3.2 Development Environment.....	24
4.4 Back End Technologies.....	25
4.5 Implementation Details.....	26
4.5.1 Recorder Service.....	26
4.5.2 Add Job.....	28
4.5.3 Select Jobs.....	28
4.5.4 Synchronize.....	30
4.5.5 Upload Data.....	31
4.5.6 Database Design.....	31
5 Chapter : Evaluation.....	33
5.1 Testing.....	33
5.1.1 System Testing.....	33
5.2 User Evaluation.....	34
5.3 Application performance on a Device.....	36
6 Chapter : Conclusion.....	38
6.1 Achievements.....	38
6.2 Future work.....	38
6.3 Conclusion.....	39

7 References.....	40
Appendix.....	42
Appendix I – User Evaluation Questionnaire.....	42
Appendix II – User Feedbacks.....	46
Appendix III – Test Cases.....	47
Appendix IV – User Manual.....	52
Launch and Login to Application.....	52
Add New Job.....	53
View Jobs.....	54
Edit Job.....	55
View Data.....	55
Appendix V – Project Information.....	56

List of Figures

Figure 2.1: Structure of a typical sensor network.....	7
Figure 2.2: Wireless sensor network gateway for transmitting data to nearby mobile phones over a short-range radio link [15].....	9
Figure 3.1: Use case diagram for overall system.....	13
Figure 3.2: Entity Relationship diagram of the system.....	14
Figure 3.3: Login screen wireframe.....	15
Figure 3.4: Add Job wireframe.....	16
Figure 4.1: System Overview.....	22
Figure 4.2: Recorder Service Sequence diagram.....	26
Figure 4.3: Add Job Sequence diagram.....	28
Figure 4.4: Get Job Sequence diagram.....	29
Figure 4.5: Synchronize Sequence diagram.....	30
Figure 4.6: Upload Data Sequence diagram.....	31
Figure 4.7: Database structure.....	32
Figure 5.1: Sample user distribution for evaluation by computer literacy and age range.....	34
Figure 5.2: Questing vs User Rating.....	35
Figure 5.3: Avg. Rating vs User.....	36

List of Tables

Table 5.1: User Rating Catalog.....	35
Table 1: Test case 1.....	46
Table 2: Test case 2.....	46
Table 3: Test Case 3.....	47
Table 4: Test Case 4.....	47
Table 5: Test Case 5.....	48
Table 6: Test Case 6.....	48
Table 7: Test Case 7.....	49
Table 8: Test Case 8.....	49
Table 9: Test Case 9.....	50

List of Abbreviations

WSN	Wireless Sensor Network
GPS	Global Positioning System
WSN	Wireless Sensor Network
GUI	Graphical User Interface
NFC	Near Field Communication
IDE	Integrated Development Environment
SDK	Standard Development Kit
ADT	Android Development Tools
SOAP	Simple Object Access Protocol
REST	Representational state transfer

Acknowledgement

First and foremost I owe my deep admiration to the Director and all academic and non-academic staff of University of Colombo School of Computing, for facilitating us to follow a great Masters program.

I would like to thank my supervisor, Dr. Prasad Wimalaratne, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the project, especially android mailing list and online tutorials for their enormous support.

1 Chapter : Introduction

An ad-hoc or short-live network is the network of two or more mobile devices connected to each other without the help of intervening infrastructure. In contrast to a fixed wireless network, an ad-hoc network can be deployed in remote geographical locations and requires minimum setup and administration costs. Moreover, the integration of an ad-hoc network with a bigger network, such as the Internet or a wireless infrastructure network increases the coverage area and application domain of the ad-hoc network.

An ad-hoc network can be classified into two main types: mobile ad-hoc network and mobile ad-hoc sensors network [7], [1]. Unlike typical sensor networks, which communicate directly with the centralized controller, a mobile ad-hoc sensor network follows a broader sequence of operational scenarios, thus demanding a less complex setup procedure. A mobile ad-hoc sensor or hybrid ad-hoc network consists of a number of sensors spread in a geographical area. Each sensor is capable of mobile communication and has some level of intelligence to process signals and to transmit data. In order to support routed communications between two mobile nodes, the routing protocol determines the node connectivity and routes packets accordingly. It also helps them to save its limited battery power on the individual nodes as well. This makes a mobile ad-hoc sensor network highly adaptable so that it can be deployed in most of the environments.

The mobile ad-hoc sensor network is a new invention with a long-term potential for transforming our daily lives. In mobile ad-hoc sensor networks, each host may be equipped with a variety of sensors that can be organized to detect different local events. Moreover, an ad-hoc sensor network offers less overhead on laying infrastructure and deploying. But in terms of initial cost it will be quite high, because those nodes are comparatively expensive. Therefore the initial cost depends on the number of nodes needed for network.

1.1 Motivation

Ad-Hoc wireless sensor networks are becoming an important part of the technology. It already been used in different scenarios to save lives. And the basic goals of using a wireless ad hoc sensor network generally depend upon the application, but the following facts are the most common motivators to use sensor networks [8].

Determine the value of some parameter at a given location: In an environmental network, it might be necessary to know the temperature, atmospheric pressure, amount of sunlight, or the relative humidity at a number of locations. According to this example it is clear that a given sensor node can be connected to different types of sensors, each with a different sampling rate and range of allowed values.

Detect the occurrences of 'events of interest' and estimate parameters on the detected event: In the traffic sensor network, one would like to detect a vehicle moving through an intersection and estimate the speed and direction of the vehicle.

With the rise of powerful mobile devices and mobile platforms like Google Android, Blackberry OS or Apple iOS, it has become feasible to find alternatives to avoid these problems that are bundled with traditional sensor networks. Mobile device platforms tend to be widely available, as economies of scale drive their prices down, because of that powerful mobile device are everywhere these days. Their hardware and software systems are much more homogeneous in nature than most of sensor network architectures as a whole. Moreover they have built-in support for a number of sensor needs, such as cameras, cellular radios, magnetometers, gyroscope, near field communication sensors etc. And because of these devices are broadly used for personal consumption, they provide a sense of immediate availability which is not seen in traditional embedded systems. That is most of the time those devices are idle and can be used for some effective work.

There are lots of sensor networks deployed all around the world for different purposes, even though there are many drawbacks. And if there is an alternative to avoid them, it can gain the maximum use from these sensor network concepts. After this has been successful anyone can use this network to collect the relevant magnetic fields data from any geological location as they expected. Hence this can save lot of funds and effort by avoiding the implementation of a new sensor network from the scratch. And going forward, if it can create a trend to use mobile devices as sensor nodes which will then lead to introduce more sensors into upcoming mobile devices. Therefore it will make this concept more useful and expandable in future.

1.2 Problem Definition

Even though sensor networks are so important and useful there are some drawbacks inherited with it, which will avoid them from being used. One of them is, it needs to deploy hundreds of nodes in order to collect accurate and efficient data on selected geographical locations. This is not an easy task to deploy them safely. And it needs considerable amount of initial cost, which is single sensor node with generic sensors cost about hundreds of dollars. Therefore it will cost a huge amount to deploy the network. Once deployed then, even losing one of them is an added cost.

Those nodes are battery powered. And they can only be active for a limited number of hours before its battery gets drained. After that it is necessary to recharge those nodes in order to continue further. Since those are low powered devices communication need to be done very carefully without wasting more power. But anyway it needs to send necessary information over to the base station.

And traditional sensor nodes have some inherited security issues. That is once they are deployed, they are on their own and no one is able to look after them. If it is necessary to make those nodes mobile, then there needs to be a proper tracking mechanism to track down each and every node. Even though there are GPS sensors it's really difficult to track them due to power constraints. And also someone in the middle can feed some false information into the sensor network.

Therefore it is clear that, even though is very helpful and flexible for researchers, there are many drawbacks that needs to be taken into consideration before going for it. And if there are any other alternative to overcome these weaknesses, sensor networks will become part of human life.

1.3 Aims and Objectives

The main objective of this is to find a way to avoid the drawbacks of the ad-hoc sensor networks by creating sensor network using mobile devices. And following are the other objectives of this,

- Find an appropriate media to communicate with other nodes. There are many alternatives like bluetooth, infared, wifi, and mobile data connection. This can use

different mediums depending on the situation.

- Model communication hierarchy of the sensor network that is how single node shares information with its base station and how that base station can send all the collected data to the single centralized location. And how frequently it should upload those data. When deciding those factors it is necessary to consider low power, low memory. Low processing and other constraints on those sensor nodes.
- Elaborate the importance of this by using an innovative application to one of the major mobile device platform. After taking all the mobile device platforms in to consideration it is decided that android is the best suitable platform for this prototype implementation.
- User evaluation to measure the importance of this solution. This will ensure that findings of this effort are useful to the society.

1.4 Scope

This is an application which will collect data on mobile devices instead of a complex ad-hoc sensor network. Therefore the most important output of this effort is to implement an innovative application on one of the mobile device platforms, to demonstrate the importance cost effectiveness and usefulness of such application. Using implemented application user can specify the type of information he needs, the frequency and around which location, based on GPS locations. Furthermore the device's GPS system can be use to find accurate locations and need to construct a method to find the closest nodes to given GPS location.

There are many medium of communications on mobile devices. Like Wifi, Bluetooth, Mobile data connection. Therefore it is necessary to find what the best medium to serve the purpose. As well as there should be a communication hierarchy. This will be the baseline for communication between node-node and node-base station. This should align with the communication medium selected above. Once best communication medium and communication hierarchy defined, a solid architecture design needed upon implementing the application.

Another important thing is to implement a policy that a user will receive data as a proportion to their contribution to others to gather information. That will keep everyone contributing to the community.

Since this will be an mobile application this should not disturb any of user activities on the device and user should not feel any difference of using the application or not. In order to achieve it, this application must use the device resources in most effective way.

Therefore in this approach a researcher or any other with the need of this kind of data do not need any more expensive sensor nodes. And also deploying such a sensor network is a huge overhead which will take a lot of effort, time and money. Therefore using this approach they can build a sensor network with lesser effort but with closely equivalent capabilities as traditional sensor network.

1.5 Overview

This is the basic introduction to the problem, aims and objectives of the project. The next, second chapter will describe the background on ad-hoc sensor networks, Android platform and literature review of the problem. And it describes the other solutions proposed, and developed for this problem and how far they are successful in solving this problem. It also briefly describes the functional and non-functional requirements that need to satisfy in order to address this problem, the communication methodology and architecture of the proposed solution.

Third Chapter is the Methodology. It describes design and architecture of the system. It includes design diagrams, ER diagrams of the database, deployment diagrams, pseudo codes for complex logics and test plan. Simply this chapter includes each and every implementation details where others can understand the gravity of this project.

Fourth and final chapter is the Evaluation. This includes the critical evaluation of the system. That is how far it is successfully achieved aims and objectives. The importance of this system to the society, strengths & weakness and the future improvements of the system will be discussed. All those points need to be justified in this chapter.

2 Chapter : Background

Although wireless sensor nodes have existed for decades and used for applications as diverse as earthquake measurements to warfare, the modern development of small sensor nodes dates back to the 1998 “Smartdust” project and the NASA Sensor Webs Project. One of the objectives of the “Smartdust” project was to create autonomous sensing and communication within a cubic millimeter of space. Though this project ended early on, it led to many more research projects. The researchers involved in these projects coined the term mote to refer to a sensor node. The equivalent term in the NASA Sensor Webs Project for a physical sensor node is pod, although the sensor node in a Sensor Web can be another Sensor Web itself. Physical sensor nodes have been able to increase their capability in conjunction with Moore's Law. The chip footprint contains more complex and lower powered micro controllers. Thus, for the same node footprint, more silicon capability can be packed into it. Nowadays, motes focus on providing the longest wireless range, the lowest energy consumption and the easiest development process for the user. [10]

The WSN is built of "nodes", from a few to several hundreds or even thousands, where each node is connected to one or more sensors. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a micro controller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting. A sensor node might vary in size from that of a shoe box down to the size of a grain of dust, although functioning "motes" of genuine microscopic dimensions have yet to be created. The cost of sensor nodes is similarly variable, ranging from hundreds of dollars to a few pennies, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth. The topology of the WSNs can vary from a simple star network to an advanced multi-hop wireless mesh network. The propagation technique between the hops of the network can be routing or flooding. Following fig2.1 illustrate the ordinary ad-hoc sensor network.

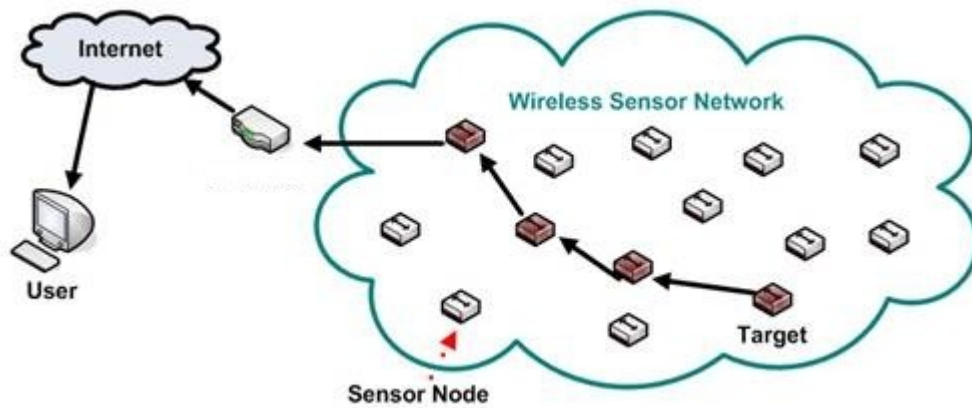


Figure 2.1: Structure of a typical sensor network

Some of the Wireless ad hoc sensor network characteristics are as follows,

Large number of sensors: Aside from the deployment of sensors on the ocean surface or the use of mobile, unmanned, robotic sensors in military operations, most nodes in a smart sensor network are stationary. Networks of 10,000 or even 100,000 nodes are envisioned, so scalability is a major issue.

Low energy use: Since in many applications the sensor nodes will be placed in a remote area, service of a node may not be possible. In this case, the lifetime of a node may be determined by the battery life, thereby requiring the minimization of energy expenditure.

Network self-organization: Given the large number of nodes and their potential placement in hostile locations, it is essential that the network be able to self-organize; manual configuration is not feasible. Moreover, nodes may fail, and new nodes may join the network. Therefore, the network must be able to periodically reconfigure itself so that it can continue to function. Individual nodes may become disconnected from the rest of the network, but a high degree of connectivity must be maintained.

Collaborative signal processing: Yet another factor that distinguishes these networks from MANETs is that the end goal is detection/estimation of some events of interest, and not just communications. To improve the detection/estimation performance, it is often quite useful to fuse data from multiple sensors. This data fusion requires the transmission of data and control messages, and so it may put constraints on the network architecture.

Querying ability: A user may want to query an individual node or a group of nodes for information collected in the region. Depending on the amount of data fusion performed, it may not be feasible to transmit a large amount of the data across the network. Instead, various local sink nodes will collect the data from a given area and create summary messages. A query may be directed to the sink node nearest to the desired location.

Wireless ad-hoc sensor networks are very beneficial in different scenarios in different manner. These networks advance operational efficiency of certain civilian applications [8]. Some of them are as follows,

- i. Military sensor networks to detect and gain as much information as possible about enemy movements, explosions, and other phenomena of interest.
- ii. Sensor networks to detect and characterize Chemical, Biological, Radiological, Nuclear, and Explosive attacks and material.
- iii. Sensor networks to detect and monitor environmental changes in plains, forests, oceans, etc. One environmental application is tsunami alert system.
- iv. Wireless traffic sensor networks to monitor vehicle traffic on highways or in congested parts of a city.
- v. Wireless surveillance sensor networks for providing security in shopping malls, parking garages, and other facilities.
- vi. Wireless parking lot sensor networks to determine which spots are occupied and which are free.

2.1 Literature Review

Sensor network have been very helpful to the society throughout the last decade. Therefore many people try to avoid those drawbacks to make maximum use of them. And make it less complicated for people who want to build up new sensor networks. In order to do so many people try to use public mobile devices to enhance the capabilities of sensor networks. Some of those efforts are described below.

2.1.1 Open wireless sensor network telemetry platform for mobile phones

Mobile phones are underutilized resources for connecting low-power wireless sensor networks to the Internet. WSNs typically expend most of their battery power on data transmission. Mobile phones carried by the public could enable a hybrid approach where data makes a low-power short distance hop to phones in the vicinity using Bluetooth or a similar short range protocol, then uses the phones' long distance connectivity to upload to the Internet. Because a large fraction of mobile phones have Bluetooth short-distance radio, this effort is to find low-cost hardware for generic WSN-to-Bluetooth gateway and open-source software that allows a wide subset of mobile phones to download and save WSN data.

This concept was proposed by C. K. Harnett and his effort is to make available an open-source “opportunistic telemetry” system that creates the long-distance data link using mobile phones carried near the sensor network, reducing the power burden on the WSN [15]. The overall system is illustrated in Fig 2.2 below.

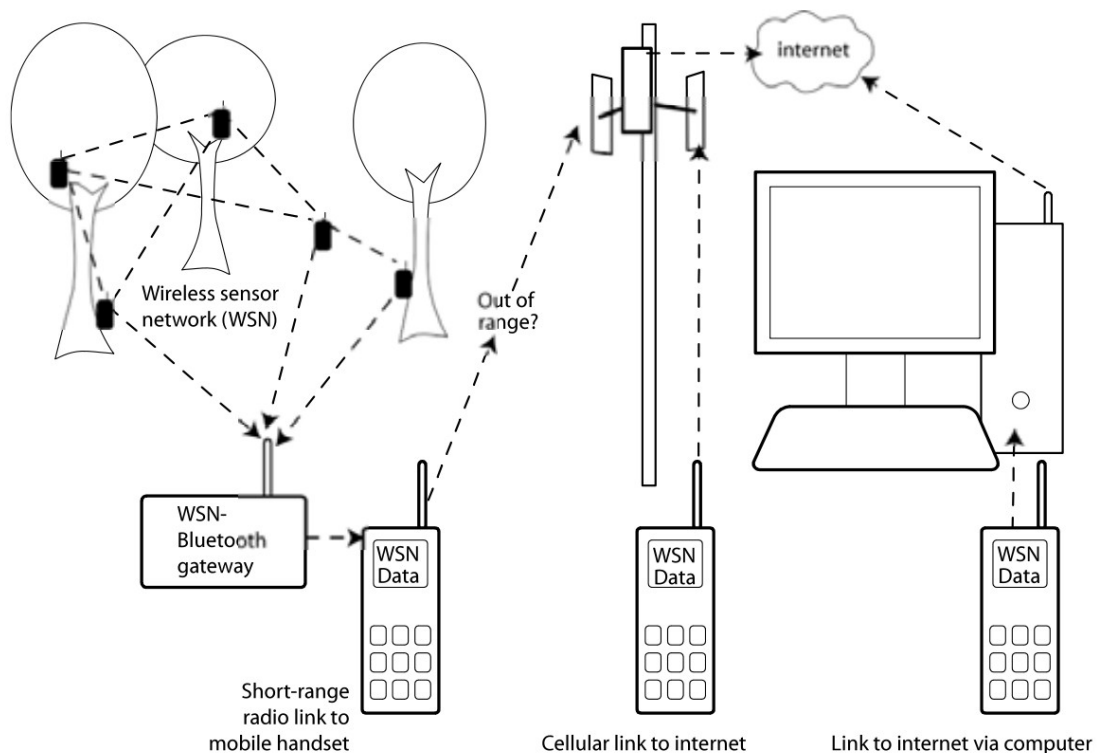


Figure 2.2: Wireless sensor network gateway for transmitting data to nearby mobile phones over a short-range radio link [15]

2.1.2 Sensor network to Android platform to control Rovio Robot

This effort was done by Yusuf Simonson, Robert Fowler at Renaissance Computing Institute and Edgar Lobatón at The University of North Carolina [14].

They believe that new smart phone architectures like Android and iOS will play increasingly important roles in sensor network design. Because of this, they wish to investigate the state of development for sensor network-based applications on the Android framework. They introduce an application for Android that allows for the semi-autonomous remote control of Rovio robots. It coordinates with a sensor network of cameras to provide a live stream of camera images and predicted robot locations. Furthermore, it provides functionality for moving the robot to user-selected destinations without the need for manual control. The application acts as a straw man for working with traditional sensor network architectures, and provides important insight on some of the challenges related to sensor network development, especially on the Android platform. They outline these challenges and provide some suggestions on semantics that could alleviate development effort.

2.1.3 Location and Mobility in a Sensor Network of Mobile Phones

This research was done by Aman Kansal and Feng Zhao at Microsoft Research center.

Mobile phones have two sensors: a camera and a microphone. Their goal in this position paper is to explore the use of these sensors for building an audio-visual sensor network that exploits the deployed base of millions of mobile phones worldwide. Among the several salient features of such a sensor network, this focus on mobility. Mobility is advantageous since it yields significant advantage in spatial coverage. However, due to the uncontrolled nature of device motion, it is difficult to sample a required region with a given device. They propose a data centric abstraction to deal with this difficulty. Rather than treating the physical devices as their sensor nodes, they introduce a layer of static virtual sensor nodes corresponding to the sampled data locations. The virtual nodes corresponding to the required region to be sensed can be queried directly to obtain data samples for that region. They discuss how the locations of the virtual sensor nodes can be enhanced, and sometimes derived, using the visual data content itself. Experiments with real data are presented to expose some of the practical considerations for their design approach.

3 Chapter : Requirement Analysis

Detail requirements need to be identified and analyzed before going into detail design of the solution. Therefore following segment describes the requirement analysis and the design to the problem described above.

Even though sensor networks are there for quite a sometime, still there are lots of areas that need to be improved to make better use of it. This effort is also to reduce the infrastructural overhead of sensor networks by using public mobile devices as sensor motes. That will avoid many difficulties that were there with traditional sensor motes. In the meantime this should not affect users mobile device usage.

At present, mobile devices are equipped with many important sensors like magnetometers, NFCs, gyroscopes, accelerometers, GPS modules, Bluetooth, cameras, radio links and there are lot more sensors waiting to be introduced into them. And most of the time those sensors are idle on those devices. Therefore in this context those idle sensor can be used to get some important work. That is we can build expensive sensor network out of those idle mobile devices by avoiding initial overhead and cost.

If we can build an application that can convert a mobile device into sensor node, then it can collect data and at the end it can upload all the data to the centralized location using long distance data connection. This will be the data connection of that mobile device. The other important fact is once user starts the application this should be able run on the background without affecting usual user operations until he intentionally stops it. Until that it will collect, communicate and upload data as requested. Anyone who is running the application will be a user as well as a contributor to the sensor network. And it will always send data as per others user requirements. But if a user needs to collect data then he should be able to make a request to the system with his data requirement criteria. Then as for that request others should upload data to the system.

In order to make this system a reality, there should be a mobile device application which runs on user devices and a server side back end. Mobile application will become front side which interacts with the user and it always communicates with the back end to cater user requests.

3.1 Functional Requirements

Any coherent and reasonable project must have requirements that define what the project is ultimately supposed to do. A requirement is an objective that must be met. Analysts cast most requirements in functional terms, leaving design and implementation details to the developers. Functional requirements describe what the application is supposed to do by defining functions and high-level logic. Following diagram figure: 3.1 illustrate overall functionalities of the mobile application and users' interaction with the system. Moreover this use case diagram only shows user interaction with the mobile application. There is service back end, implemented on Axis2 web service platform deployed on Apache tomcat web server.

There are three major type of users and there are,

- i. New Users
- ii. Users who add new jobs
- iii. Users who executes jobs

New user can only create a user account which is necessary to login to the system. And other two types of users can do almost same tasks. But depending on their role, there are some tasks they can perform and cannot. As an example, only the user who adds a particular job is capable of viewing or retrieving the data from server.

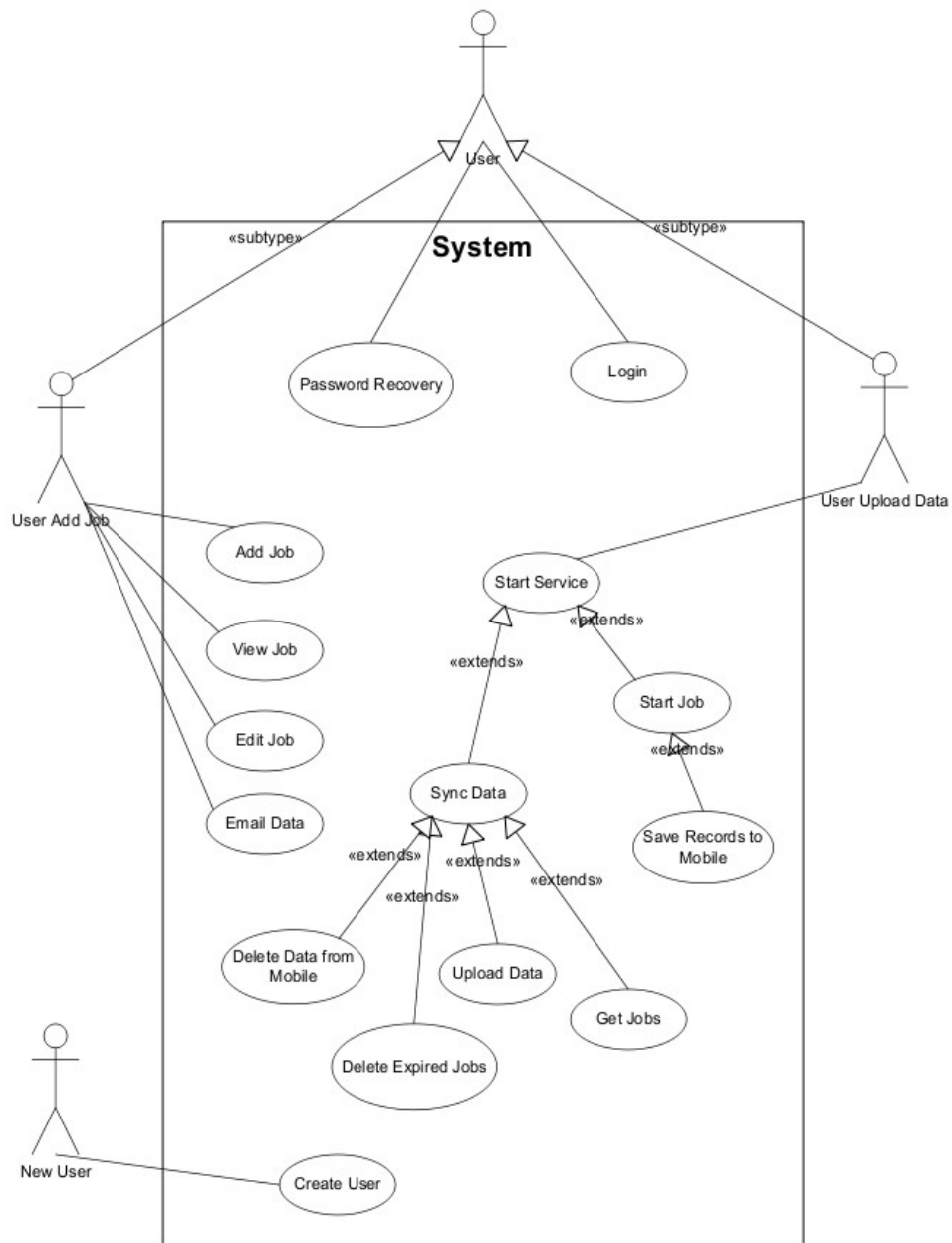


Figure 3.1: Use case diagram for overall system

Below figure: 3.2 illustrates the identified entities of this system and the relationship between those entities. Cardinality of those relationships indicated with relationship.

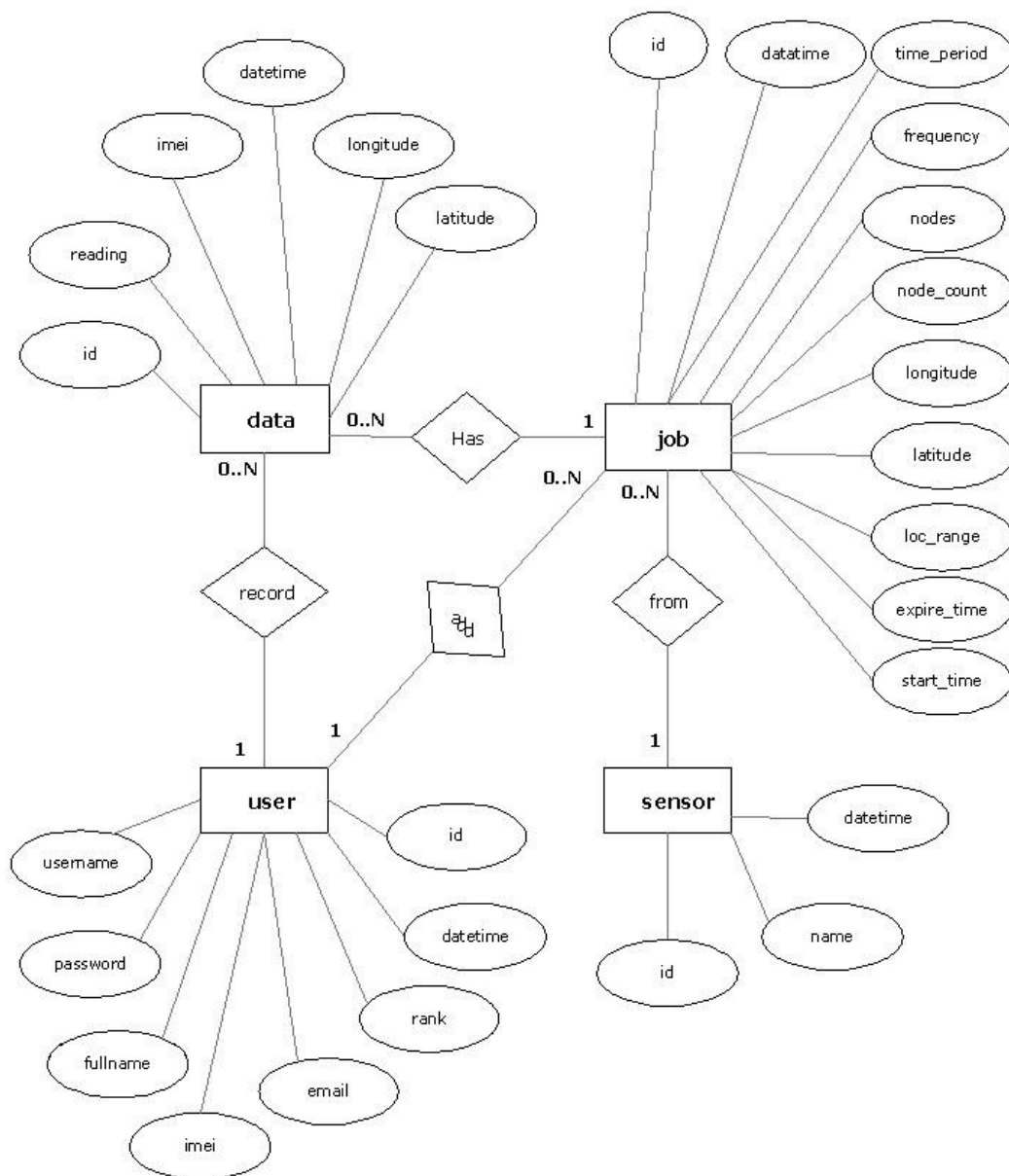


Figure 3.2: Entity Relationship diagram of the system

Following are the high level functional requirements of the application.

3.1.1 User Login

This is the first screen of the mobile application. Every user need to have an user account to login to the system. If not they should be able to create a new user account using Register new user screen. And if user forget or lost his password and user name there should be a password recovery methodology in order to gain access to his account again. The entire password authentication and create new user account

function should be implemented and the application will invoke the back end service for all of them. This does not use its own local database. This is only for mobile applications security procedure, and there should be separate layer of web service security at back end.

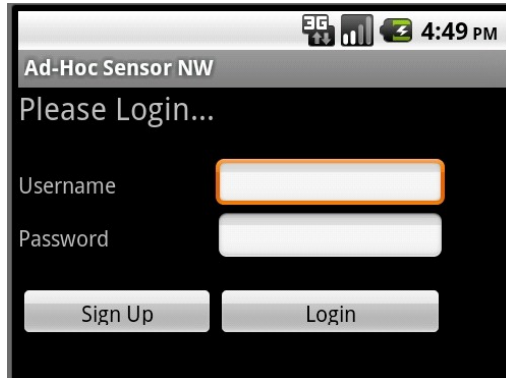


Figure 3.3: Login screen wire frame

3.1.2 Add Job

This will enable user to add job into queue. This will invoke the back end web service to add this job into the server database. The UI screen will consist of following fields to be added to the DB. Figure 3.4 illustrate the add job screen.

- Sensor Type – Which sensor data need to be recorded
- Latitude – Base location latitude of the data to be recorded
- Longitude – Base location longitude of the data to be recorded
- Location range – Radius of the area that use to collect data
- Start time – When to start recording data
- End time – When this job going to expire
- Frequency – In what frequency data need to be recorded in minutes
- Time period – How many hours does it need to record data
- Nodes – Number of nodes to be participating in collecting data
- Description – Additional information

Figure 3.4: Add Job wire frame

3.1.3 View/Edit Job

This enables users to view and edit the job that he added. First user can view the jobs. Then he is able to edit them before they get expired. User can edit all information except the sensor type. Because a job totally depend on the sensor type and changing the sensor type is more or like adding new job into the queue. For both view and edit operation it should invoke the server implementation to get those data.

3.1.4 Synchronize

This is one of the most important functionality on this system. Because this will synchronize the local system, that is mobile application with the centralized server which is the back end web service. User should be able to explicitly invoke this function to synchronize his device with server. And also other operations may be also invoking this function to keep the device updated. This operation need to do following tasks in order to complete synchronization.

i. Upload data

First it should check whether there are any records in the local 'data' table. If there are any, those data should be uploaded into the server and once server updated successfully all the record on the local 'data' table should be deleted. Because otherwise local database will gradually fill up the mobile devices memory space.

ii. Delete job records

If there are any invalid jobs (that is expired jobs) in the local 'job' table delete them all. Because those are invalid jobs, they are not going to execute in the mobile device.

iii. Get new jobs

Here first it checks whether there is a valid job on the local 'job' table. And if there is, it should continue working on that job and persist data on that job. And if there isn't any job on the local 'job' table then it should invoke the server to get new jobs from the server. And continue work on them.

3.1.5 Record Service

This is the main functionality of this application. That is to read and persist the sensor data according to the job. This is a quite complex task and following points will illustrate the functionality.

i. Once user touch to start recorder service this should start separate new worker thread to start work on reading and uploading data. It should run independently and record data according to the specified job.

ii. First thing it should do once user command to start recorder service is to synchronize the mobile device with the server. Because may be the device will be at a stale state.

iii. Now there should be new valid jobs on the local 'job' table. Therefore it will grab that job and start working on it. It should persist the sensor reading and other relevant information in to the local 'data' table with the specified job frequency. Each time, before it persist data into the local database it must make sure that this job is valid with in specified time range and specified location range. If it does not

satisfy that criteria it will make the job as expired or on-hold according to the situation.

- iv. There must be specified upload timeout, and once it reach upload timeout it will upload all the records on the local 'data' table into the server and clear the local table to avoid unnecessary space problem on mobile device.
- v. If there is valid job, but is supposed to start at a time in future then it should go into sleep mode until it reach the time to start that job.
- vi. Also if there isn't any job found according to the location and the capabilities then the service will not need to be alive and it should go into sleep mode for specified time period and check back again after it wakes up.

3.1.6 View Data

This will enable user to view the data recorded by the different users for that particular job. But since there are small space available at mobile device screen, its should not displayed the information on this view data screen.

3.1.7 Email Data

Usually mobile devices have small screens compared to computers. Therefore its little difficult to display large amount data on those small screen. Therefore user and request the system to send his data to the email address specified by him. Then it becomes convenient for the user to do whatever he wants with those data. Email send operation should be done at the server side

3.1.8 User Ranking Process

There are many users who add jobs and contribute for data collecttion. Therefore it is necessary to have some kind of process to identify the users who contributes a lot and make their jobs prioritized when they need to collect some data. This will encourage others to contribute and grow this community. This can be done as and every time user uploads data into the server it should increment integer associated with the user. This integer can be called as the rank of the user. This will maintain the stability and fairness of the system.

3.1.9 Server Side Implementation

There should be a server side back end to cater user requests. All the data should be finally uploaded into that server. Then users can look into their data from the server or request it to email. Users should submit their data requirement, which is a job into the system. This server should manage all those jobs from different users and distribute those jobs to the most suitable contributors to start work on them. And contributors have to upload readings back to the server. All the communication with node should be happen through that back end service.

3.1.10 Exception Handling

There should be proper exception handling mechanism. Because this mobile application interacts with a separate server and therefore it need to properly propagate exception from server side to mobile application which is to front end of the system. Otherwise front end will not know what happened at the back end and it will freeze the application till get response from the back end.

3.1.11 Service Level Authentication

All the mobile devices are accessing a single server. And there should be way to authenticate the application that invokes the services. Otherwise anyone from outside can invoke those service and they can misuse the system very easily.

3.2 Non-Functional Requirements

- i. Application need to use the sensor and the battery optimum way. Otherwise it will lead to drain the devices battery very quickly. Which make it really hard time for user to charge his device frequently.
- ii. Because of the mobile application is running on mobile device, it consist of limited battery, limited processing power and limited memory. Therefore it should be really careful when implementation to make it 100% optimized. Otherwise it will eat up all the resource of the device and it will ruin the device.
- iii. Since this is running on the background this should not affect the normal usage of the mobile phone. Simply this should not disturb any of the users' activities.

- iv. Sensor network integration need to be done seamlessly that the mobile device user should not feel that some other application is running on his device.
- v. The UI layer should follow the proper standards. This will make it more familiar to the users. Therefore users must be able to adapt to the system very easily.
- vi. Server side needs to be alive all the time. Because without it, mobile application cannot do anything. Therefore it is very important to make sure that service is up and running all the time.
- vii. When hundreds of uses are connecting the service, there may be high capacity transferring from server to devices, which will make the service slow down at the mobile end. Therefore server side must have enough bandwidth to cater large number of users connecting at once.

4 Chapter : Design and Implementation

Ad-hoc sensor network are very much important to the research who are involved in different kind of researches. It does not restrict to particular research interests, but it is used and still using in total different subject areas. Therefore it will be important to create some system out of those requirements.

4.1 Proposed Solution

The main objective of this is to build a sensor network using mobile devices by considering above requirements. In this context a mobile device will be a sensor mote which can play the traditional sensor motes role. Therefore once the prototype application implemented with the above specified requirements, it can be used to collect relevant sensor readings as per users criteria. Once user specified the location and the time frequency of the data needed, application will upload this job to the server. Then server will try to find relevant devices to do that job. For this it will use the device's in built GPS sensor. When it finds a device it will send that job and device will take over the job and start recording data. First it will persist data into its local database. Then it will upload that data into the centralized server in batches. This will make use of devices data connection or the wifi network to upload data. But since it's not using throughout while time, it saves most of the devices battery life.

Therefore this effort here is to use public mobile devices as sensor network. That is today's most of the mobile devices like phones, tablets, game consoles can collect data, persist to databases, find location information and communicate over Internet. But it cannot do this task using factory mobile itself. It needs some sort of application to manage device communication, data transfers user jobs and everything. Because of that there is a need of a simple application and that is where android becomes the best platform to implement the application on.

4.2 Design Overview

As for the requirements this system need to be an client server system, which will allow all the client devices to connect into the server to get user requests ans upload data into the system. The client will be a mobile device application and the service will be a web service. Mobile application needs to be deployed on mobile device and the web service needs to be deployed on high available web server. This web service is the heart of the system. It manages everything and mobile application executes the task assigned by the service. Then all mobile devices frequently communicate with the server and feed data and relevant information.

Since all the mobile devices are independently communicating with the server it will avoid the issue of single point of failure. That is one device failure would not effect to fail the whole system. And mobile device can use its own data connection to connect into Internet and communicate with the server. It does not need any infrastructure. As well as mobile device can user its GPS modules and other sensor when it needs them.

Following Figure 4.1 illustrates overall layout and the architecture of the system.

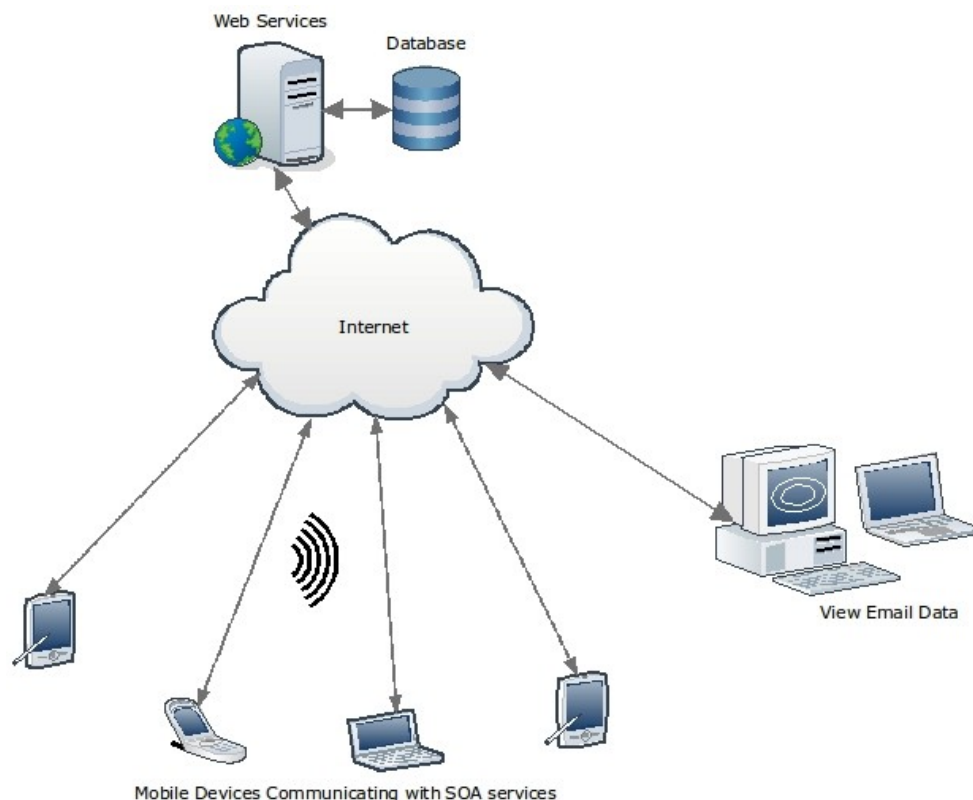


Figure 4.1: System Overview

4.3 Front End Technologies

Front end of this system is the mobile application which runs on the user devices. This application executes user jobs. And this totally depends on the centralized server and it always interacts with the server.

4.3.1 Suitable mobile platform

Today there are many popular and sophisticated mobile device platforms based on different hardware vendors. Therefore finding the best suited platform for this purpose has become important to the success of the project. Following are the most popular mobile device platforms today,

- Apple iOS
- RIM BlackBerry OS
- HP WebOS
- Google Android
- Samsung Bada
- Microsoft Windows Mobile
- Nokia Symbian Belle

After considering most of the mobile device platforms, Android has become the best suitable platform for this purpose. There were many reasons behind to choose android over others. They are,

- i. Free and open source platform by Google.
- ii. Android comes with most of the mobile device manufacturers and service providers. Which means it will not make any restrictions.
- iii. Now Android holds close to 50% of the Smartphone market share all around the world. Therefore it will help to reach most of the Smartphone users in the world.
- iv. Wide range of devices from low end mobile device to high end tablet. It will help to find different audiences.
- v. It does not need any specific hardware or software to work on the platform or to develop application on top of it.

- vi. Android basically uses java for application development. Therefore it makes easy to adopt to work on it.
- vii. Eclipse integrated SDK is freely available for application developers on android. It makes the implementation a quite simple.
- viii. There is a strong development community for Android who are willing to help beginners. And also there are lot of on-line resources and books written for android development.

Based on the above mentioned reasons Android became the most suitable platform to implement the prototype application to convert mobile device into sensor node. And it is decided to use Android 2.2 Froyo as the targeted version for the application.

4.3.2 Development Environment

Eclipse is the best development environment for the android. Its free and open source IDE and one of the most widely used IDE in Java community. There is fully featured SDK for the Google android and it also includes emulator which can be used to test most of the android applications. Also there is android plug-in to eclipse which is called Android Development Tools (ADT). With those tools, android SDK, ADT and eclipse make it really comfortable to start writing application on android platform.

Android comes with a light weight database called 'SQLite'. Therefore this can be used to data persisting operation to keep data on mobile device. But this cannot be used to store data for permanent usage, since mobile device have limited storage capacity it is necessary to upload data into the centralized server frequently and clear the local SQLite database to save space.

It is not enough to test android application only on emulator. Since this application involved on hardware sensors it's hard to completely test it on emulator. It is necessary to have android mobile device to test sensor reading, location services etc. Not only that, it is very important to test an application on a real mobile device to test aspects like performance and battery life. Therefore android mobile phone called 'Samsung T959 Galaxy S' is used as the primary test device to test this application. And some other random devices were used to test the compatibility of the application.

As the back end of this application is a Axis2 web service there need to be a way to invoke web services from the mobile application. There is library to support in that case, which is

called KSOAP2. The ksoap2-android project provides a lightweight and efficient SOAP library for the Android platform. Therefore it avoids all unnecessary overhead of invoking back end.

4.4 Back End Technologies

In this system server side back end is the heart of it. All the processing and decision making happens here. Android application or the front end just executing the orders from the server. Therefore server side implementation also needs to be more efficient and flexible. Following technology stack has becomes more appropriate after considering the requirement which it suppose to cater.

i. Servlet Container : Apache Tomcat 6.0

Apache Tomcat is the most popular and widely used open source servlet container. And it is a simple & light weight server which can be use with Axis web service engine.

ii. Web Service Engine : Apache Axis2 1.4

Axis2 is also an open source and light wight web service engine. This is a complete re-design and re-write of the widely used Apache Axis SOAP stack. Apache Axis2 supports SOAP 1.1 and SOAP 1.2, and it also has integrated support for the widely popular REST style of Web services.

iii. Database : MySQL 5.0

As a free and open source project, MySQL is the first choice as the database of this system. Even though it has few paid versions for commercial use, it provides full-featured database management system which is more than enough for this kind of system. And there is really nice front end for MySQL called 'MySQL administrator' and 'MySQL query browser'. This make its even easy to manage this database.

iv. Eclipse with Web Tools plug-in (WTP)

Eclipse IDE is the obvious choice for Java developers. And with WTP it has full support to developing web service and other j2ee application. It is a hassle free environment to work with Java application.

4.5 Implementation Details

Further implementation details on important components of the system are described below. Those diagrams and pseudo codes will illustrate the logic behind the functionalities.

4.5.1 Recorder Service

This service is running on the mobile device. It is a background service which will run by itself upon user command to start service. This is the basically do record all the sensor reading into the local sqlite database with all other information, according to job specified. Figure 4.2 is the sequence diagrams for Recorder Service.

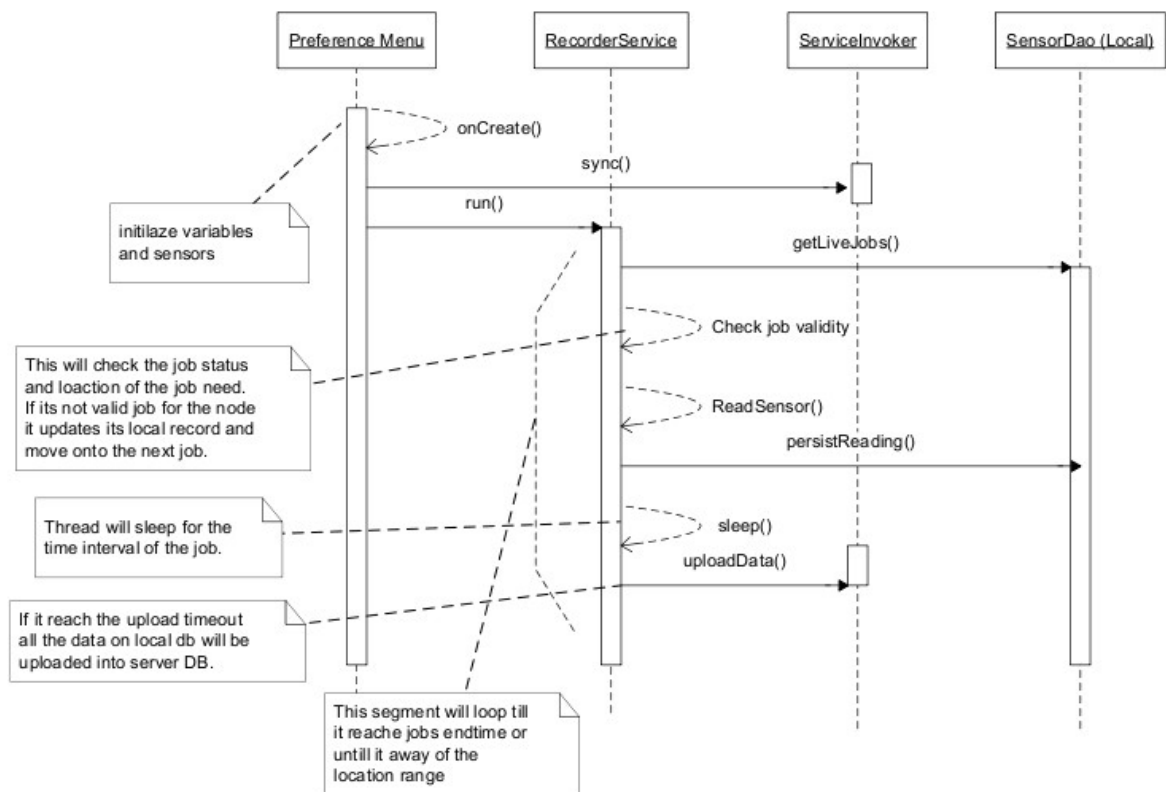


Figure 4.2: Recorder Service Sequence diagram

Pseudo code

```
Initialize sensors and variables
Initialize available location service (GPS provider or network provider)
Sync local database with back end
While service is active
  Get live jobs from local database
  If job found choose first record
    While job status is alive
      If upload timeout reached upload data
      If job is in valid location range and time range
        persist sensor reading data into local database
        thread sleep for the job frequency till next reading needed
      else
        if job expired or out of location range
          update the correct job status
          sync local db with back end
          get live jobs from local db
          if there is any live job select the first job
        else if job is yet to start
          sleep the thread till it start it
          update the job status
    else
      sleep thread for constant amount of time
      sync local db with back end
```

4.5.2 Add Job

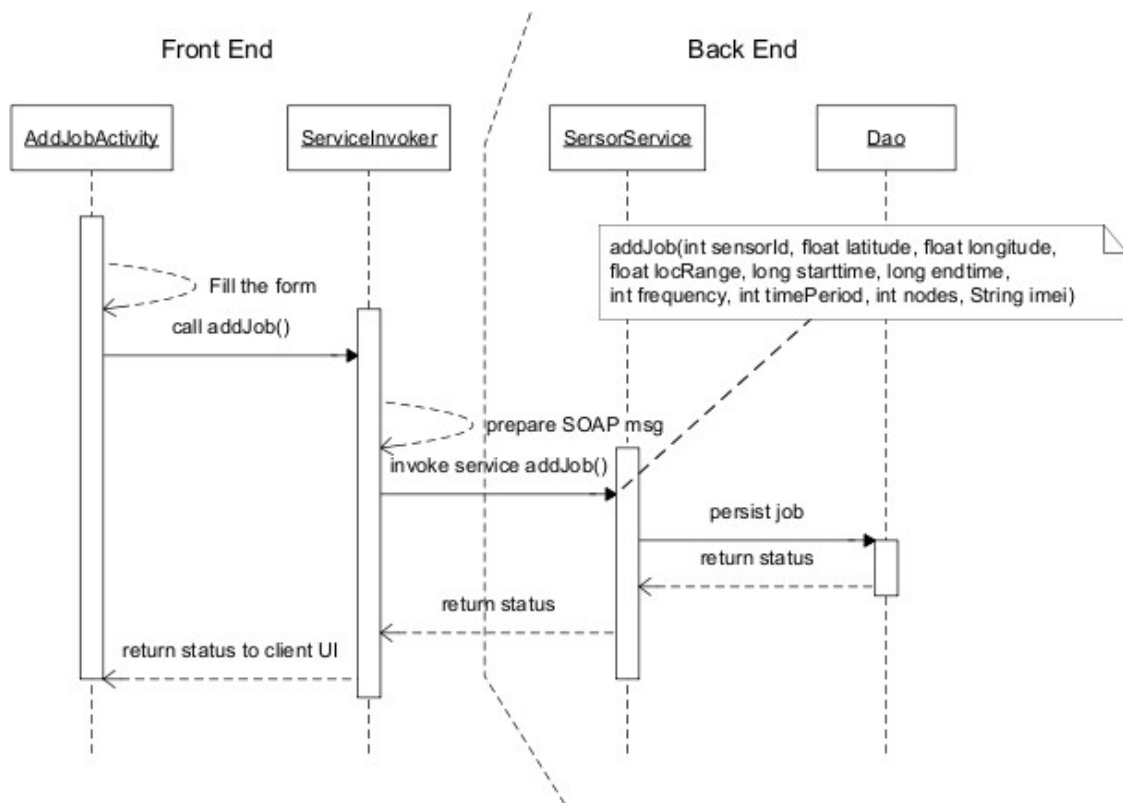


Figure 4.3: Add Job Sequence diagram

This will add an entry to the job table as users' data requirement. Mobile application directly invokes the back end web service and persist the job record. It will not persist anything to local database. Above figure 4.3 depicts the add job functionality.

4.5.3 Select Jobs

Get jobs is one of the important operations on the back end. Because it is the one who select the most appropriate mobile node for a particular job. Following sequence diagram, pseudo code illustrates the basic logic behind this functionality. Fig3.4 illustrates the sequence diagram on select job functions,

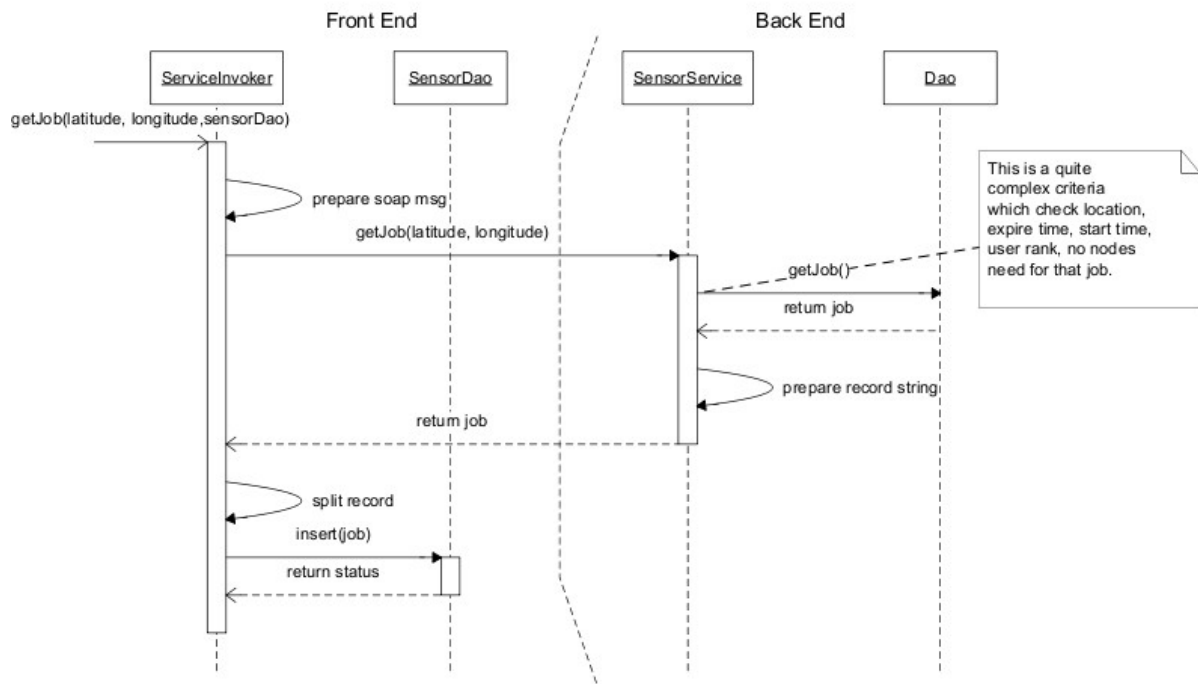


Figure 4.4: Get Job Sequence diagram

- i. When selecting the appropriate job it is very important to check the location range of the mobile node with job location. Therefore to find distance between two GPS location is calculated using following formula. All these longitude, latitude values need to be from radians. R is the radius of the planet.

location1 = (lat1, lon1)

location2 = (lat2, lon2)

$$\text{dist} = \arccos(\sin(\text{lat1}) * \sin(\text{lat2}) + \cos(\text{lat1}) * \cos(\text{lat2}) * \cos(\text{lon1} - \text{lon2})) * R$$

- ii. Number of nodes that needed to complete the job, should be greater than the node_count which is the number nodes currently do that job. That is this job still needs more nodes to do its work.
- iii. Job expire date must be satisfied. It should fall in future.
- iv. Job start time should satisfied following rules

```

if job start_time is null
    start_time = now
if job start_time is not null and its already in past
    start_time = now
if still start_time is in future
    it will be the start_time
  
```

- v. Job expire time should satisfied following rules

```

If expire_time is null
    expire_time will be (start_time + time_period)
if expire_time is not null and (start_time + time_period) < expire_time
    expire_time = start_time + time_period
else
    expire_time will be itself

```

- vi. First order the record set by start time ascending order to find the closest jobs to the current time. And then order records by user rank descending, to get the highest rank users job with priority within the closest jobs.
- vii. DateTime on sql can not assign as null it will always be "0000-00-00 00:00:00" therefore we need to check for "UNIX_TIMESTAMP(field)=0". Because UNIX timestamp of a time will 0 when it has 0 values.

4.5.4 Synchronize

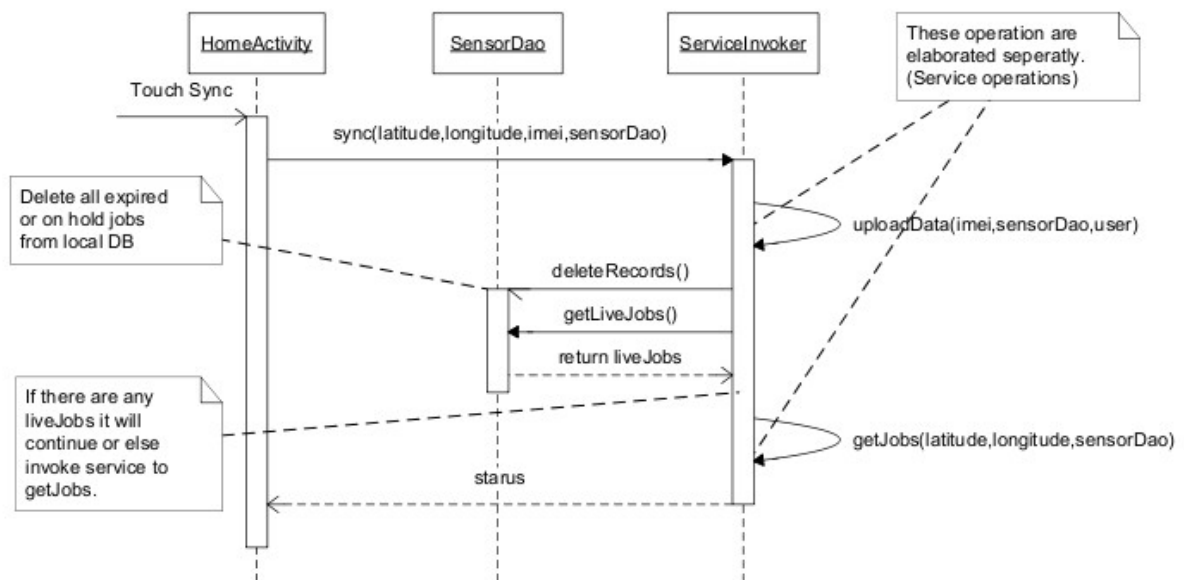


Figure 4.5: Synchronize Sequence diagram

This is used to synchronize the local database with the centralized database. It uploads all records on the data table and deletes the invalid jobs from the local database. Then get valid job from the server and insert into local database. Figure 4.5 shows sync functionality.

4.5.5 Upload Data

This will upload the data from local device to the centralized database. Each time application uploads data into the server it will increment the user rank to mark him as a valued contributor. Following sequence diagram explains its activities from mobile device to the back end server. Following figure 4.6 illustrates upload data functionality.

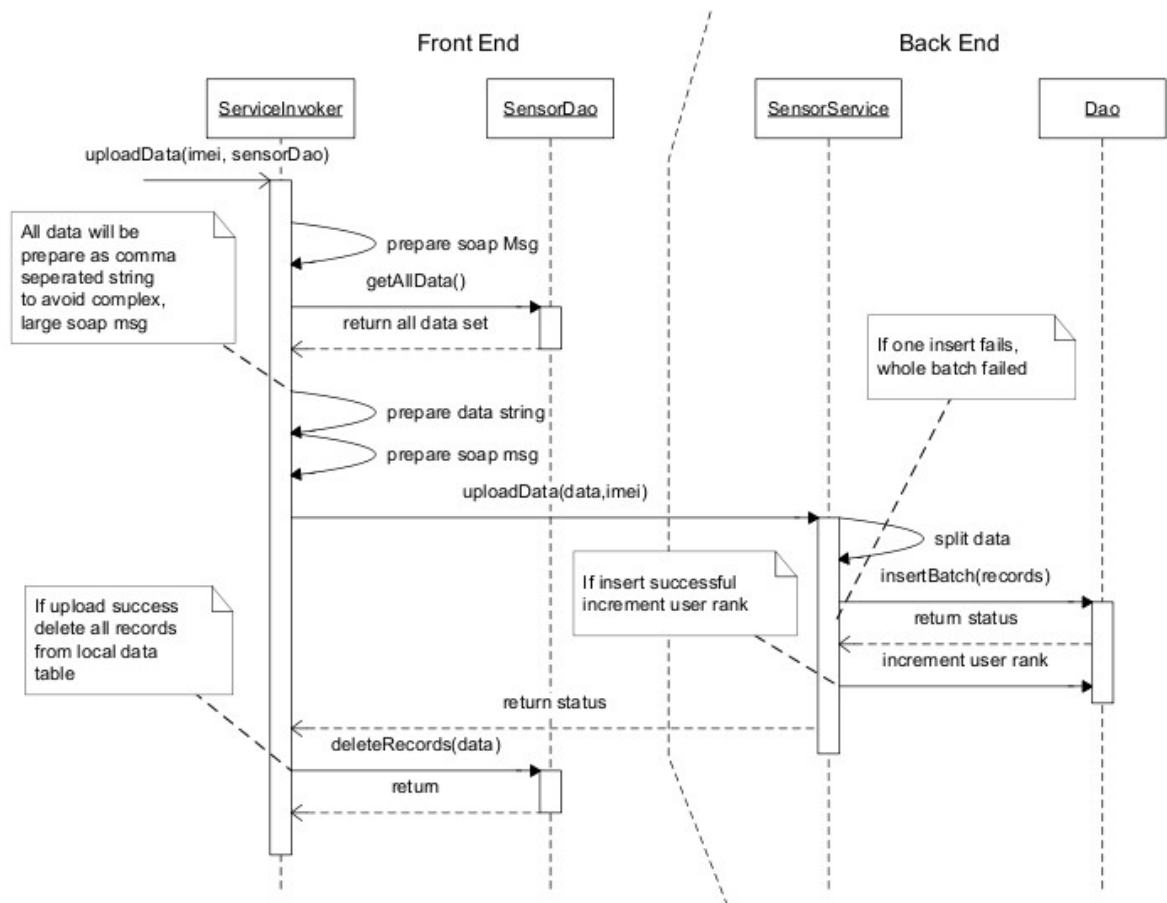


Figure 4.6: Upload Data Sequence diagram

4.5.6 Database Design

Figure 4.7 is the database table structure for the system. Both front end and back end duplicate this database schema. But local application is not using all the tables and the field of the schema. And it is important to have database on local device, because otherwise it will have to connect to server all the time. This will lead to eat up memory, CPU time and battery of the mobile device.

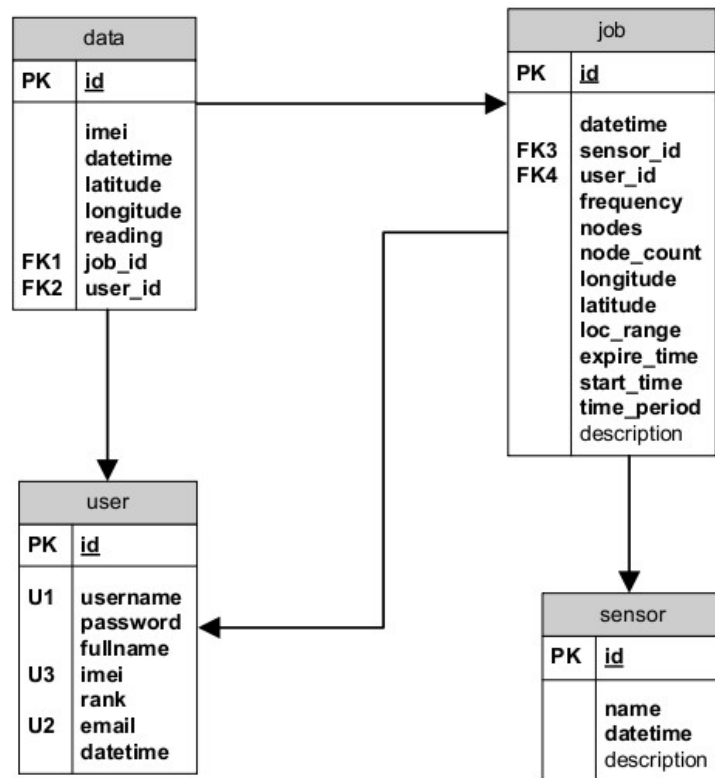


Figure 4.7: Database structure

5 Chapter : Evaluation

5.1 Testing

Testing is the most important to make sure this system adhere the functional and non functional requirement identified at the analysis phase. This happens on two stages that is,

i. Developer unit testing

This is done after implementing each and every functionality. This will verify that the functionality work as developer expect it to be. Therefore this test is done from the developers' angle.

ii. System testing

This is full functional test. It is done to make sure as system this application meet all the requirements successfully. Complete set of test cases were developed using the requirements identified at analysis phase and system was tested upon those test cases.

5.1.1 System Testing

All of the test cases found under Appendix III cover the basic functionality of the system. After implementation, while running those test cases it was successful for most of the test case. But there were some defects on couple of test cases. One of the severe defects that found was in the uploading operation as illustrated on table 7. Because of some coding mistake the uploading never happened, but it should be uploading data periodically as specified on the job. As well as the service didn't stop once it is commanded to stop. That was there because of some logical error and it didn't take much time to fix it.

Another issue was found on the exception flow of the system. That is once service failed the application fails. Application didn't handle any exceptions coming from server. This was due to a design issue. There should be a proper design to propagate exception and handle them at the proper stage. This took some more time to fix the issue.

All of these testing are done from the developer's angle and it will only make sure this system satisfies the requirements. This is where user evaluation comes, it will test the application from users perspective.

5.2 User Evaluation

This becomes important because all other testing is done from the developers perspective. This is totally depending on user feedbacks and this is kind of beta testing of application life cycle. After complete implementation of the system application distributed among selected set of users along with the prepared questionnaire (Appendix I). Their feedbacks after using the system are really useful to find out defects, usability, weaknesses, improvements and future enhancements of the system. 15 users have been selected for the user evaluation from different user categories. And following figure 5.1 show the user distribution with respect to the users age range and the computer literacy.

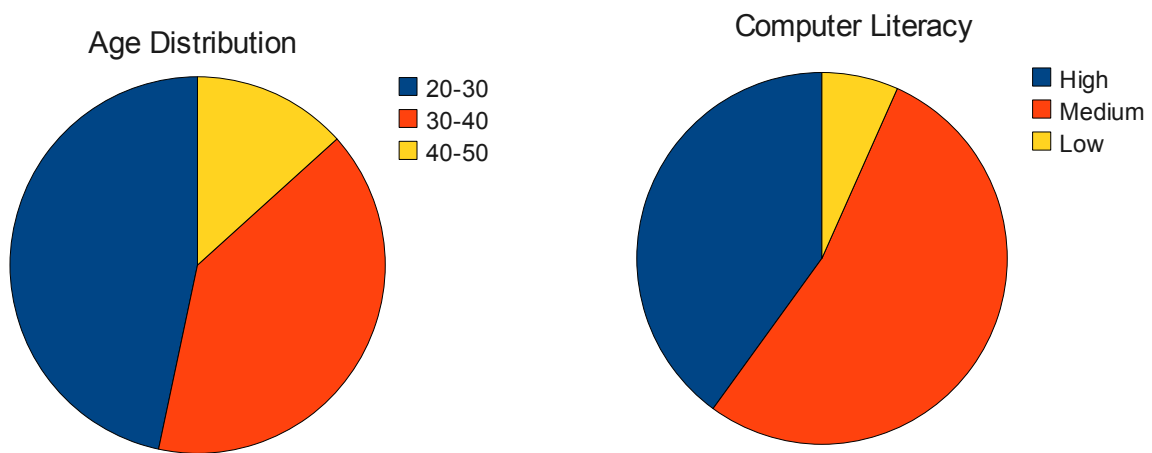


Figure 5.1: Sample user distribution for evaluation by computer literacy and age range

All the feedbacks from the selected users are illustrated in Appendix II. According to those feedbacks following information was extracted. And below table 5.1 shows the meaning of the user rating denoted by a number.

	Description
1	Strongly Disagree
2	Disagree
3	No Comments
4	Agree
5	Strongly Agree

Table 5.1: User Rating Catalog

Below figure 5.2 show the average user rating to the all 27 question on the questionnaire distributed with the application. According to this diagram it is clear that all most all questions got an acceptable level of user rating and comments to improve them. Out of those 27 about 7 questions got user rating below 3. Which implies that about 75% of the system features and performance are up to the expected level. And it was necessary to dig in and analyze the other 25% and find out what went wrong and how to improve those. With that analysis, few defects were found that were not able to identify at the developer testing stage. Furthermore it helped to identify and fix few issues associated with the back end service and authentication framework.

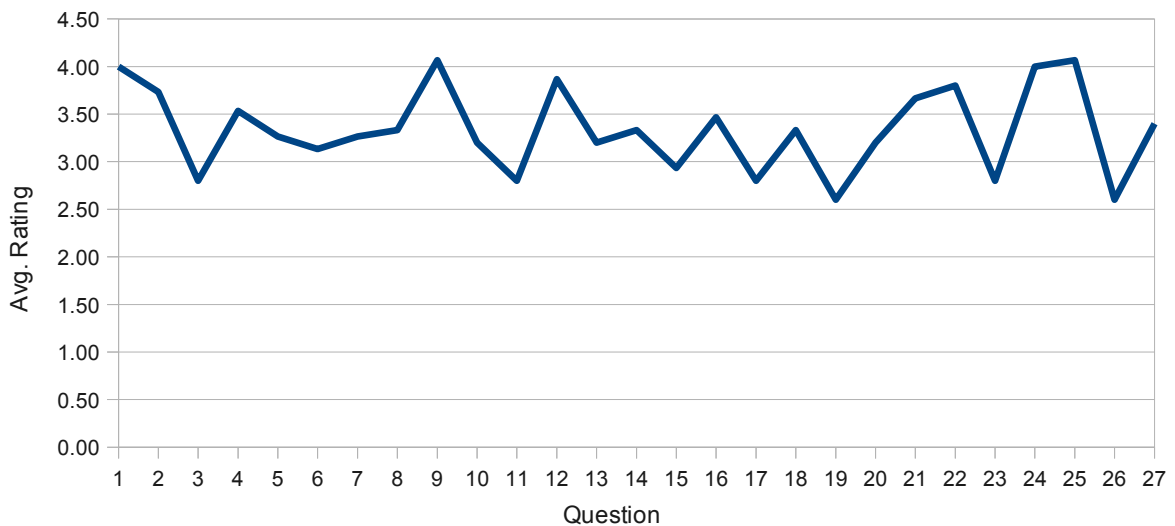


Figure 5.2: Questioning vs User Rating

Below figure 5.2 show the user rating against the whole system. It illustrates that most of the users that were used to test the application rated positively. From those 15 users about 13 people rated above 3. Which is about 85% of the users accepts the system. 2 users and about

15% of all users rated less than 3 and they have left some important feedback to improve this system. The major concern that they had was to improve the stability of the system. That includes the battery life of the device, resource usage and data usage. These facts are really valid and even though there is a limit where it is possible to optimize those factors.

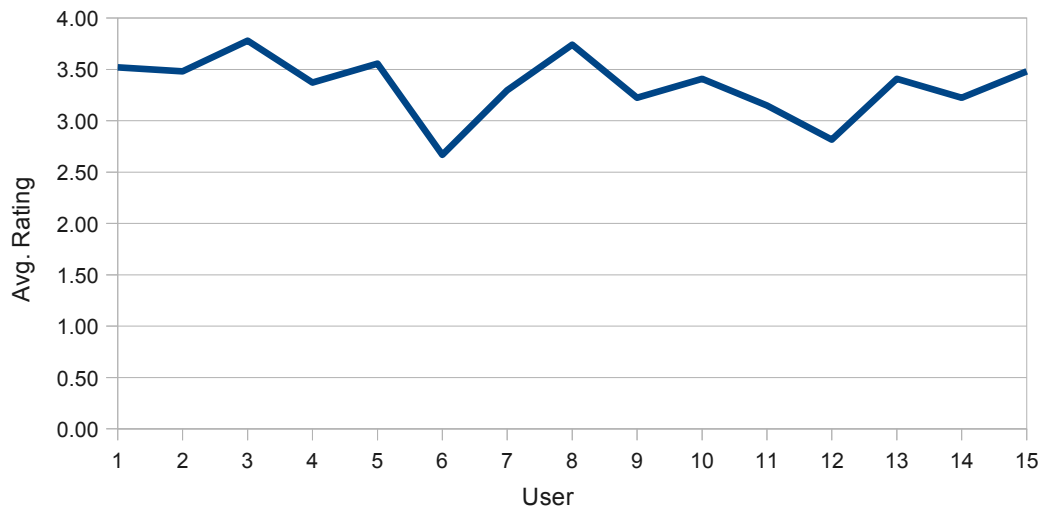


Figure 5.3: Avg. Rating vs User

5.3 Application performance on a Device

One of the main requirement of this application is to use minimum resource on the running mobile device. Because this functionality is not the main purpose the running mobile device. Therefore user should not feel any difference, a lag or any abnormal behaviors while using the application. And if not, user may not accept to use this since it interfere with usual activities.

In order to verify that it is necessary to check the resource usage on the device. Therefore some of the tools are available on Android market which will calculate the CPU and memory usage on each and every process which is running on the device. Following figure 5.4 and figure 5.5 shows resource usage on this application by using two resource monitoring tools.

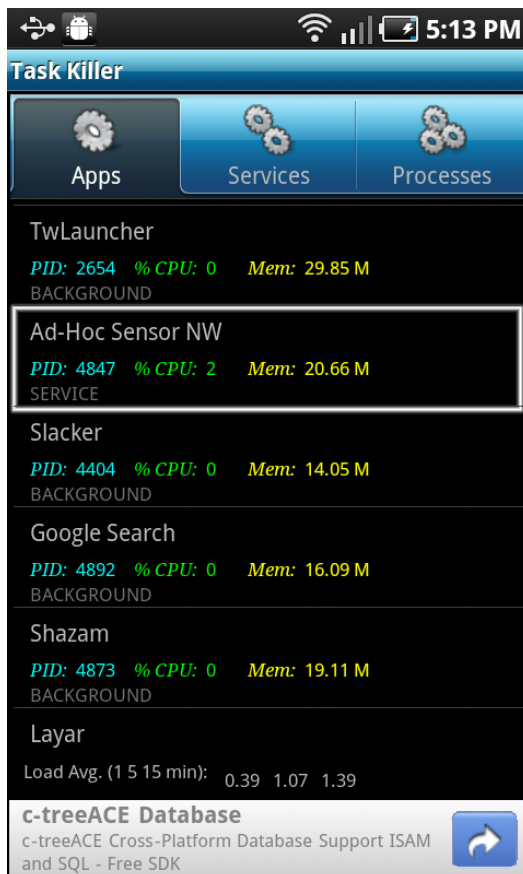


Figure 5.4: Resource usage on Astro Task Killer

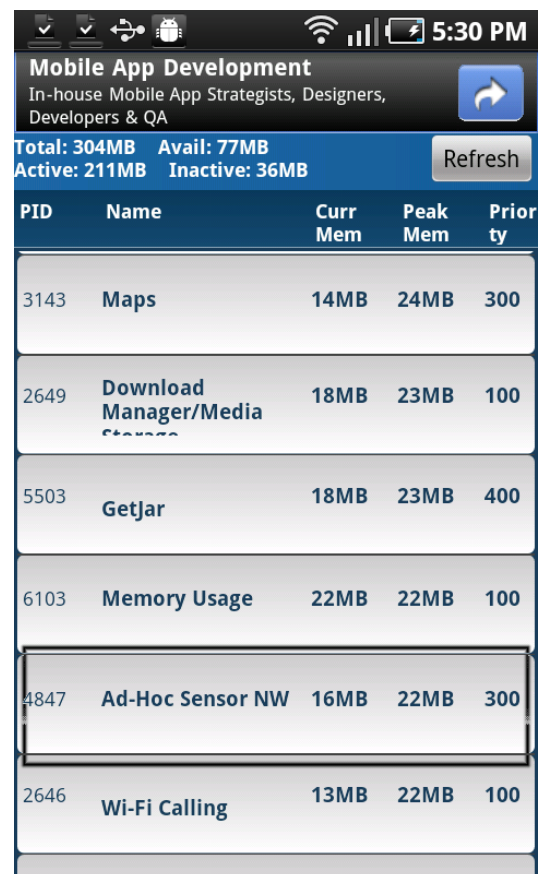


Figure 5.5: Resource usage on Memory Usage Tool

6 Chapter : Conclusion

6.1 Achievements

Finally there is a system up and running which everyone can freely use. Back end service is hosted on some third party hosting environment. And android application is freely available for everyone through the Google codes. All this project related information can be found under Appendix V.

Devices communicate with the back end service using SOAP messages which uses devices' data connection. All the communication of device and service happen via mobile data connection. Application can only operate with availability of that data connection. Implement a proper mechanism to prioritize the user request upon their contribution to the system, that will keep this system live from community contribution. Even though this system can not replace traditional sensor networks this can be useful to everyone who cannot afford complex and expensive traditional sensor network.

6.2 Future work

Currently all nodes communicate with the server and wasting all their resources to upload data. Even though they are doing it periodically it is a waste of resources. It will be an improvement, if it is possible to find nearby nodes using bluetooth and transfer all data into one randomly selected node. And that node will upload all those data for its nearby nodes. Therefore all others do not need to use data network to upload data. And it saves all others battery power as well as the data usage, but there is an inherent problem with that. But usually users do not keep their bluetooth transmitters on while they are not using them. In that case system does not have the authority to turn on the transmitter. Therefore if it is possible to find some work around that it will improve this system a lot.

In user evaluation some important improvements to the system came up. And most of the critical improvements were managed to add up to the system. And one of them is to enhance the service side authentication. Back end web service and client application is communicating with simply constructing SOAP message it is difficult to implement tough authentication mechanism. But request level user name password verification. And still there is a need of proper authentication protocol.

6.3 Conclusion

This mobile sensor network will be useful for people who needs to collect data on magnetic field sensor and it will avoid lot of cost and infrastructural over head. But system is not a replacement to the traditional sensor networks. Because those mobile devices does not include all the different kind of sensor that are available on traditional sensor nodes. And since this system is just came out, there will be many improvements and defects to be fixed to make this much more stable and usable product.

This client application is only available for android devices. But there are other popular mobile device platforms as well. Therefore make this client available for those platforms will enable most people to use this.

Finally this system is not standalone system which anyone can use. But this is user driven system which everyone depend on other users of this system. Every user needs to contribute to the system and that will feed data to other users who are in need of that data. Therefore to make this system work there need to be large community contributing to the system. That will make this system a reality.

7 References

- [1] Xiang-Yang Li, Wireless Ad Hoc and Sensor Networks, 2nd. ed., London: Cambridge University Press, 2008.
- [2] C.D.M. Cordeiro & D. P. Agrawal, Ad Hoc & Sensor Networks- Theory and Applications, New York: World Scientific Publishing Company, 2006
- [3] F. Ableson, C. Collins & R. Sen, Unlocking Android- A Developer's Guide, New York: Manning Publications, 2009
- [4] E. Burnette, Hello, Android- Introducing Google's Mobile Development Platform, 2nd. ed., Florida: Pragmatic Bookshelf, 2009
- [5] S.Y. Hashimi, S. Komatineni & D. MacLean, Pro Android 2, New York: Apress, 2010
- [6] (2011, Feb.). Wikipedia: Smartphone [Online].
Available: <http://en.wikipedia.org/wiki/Smartphone>
- [7] (2011, Feb.). Wikipedia: Wireless Sensor Network [Online].
Available: http://en.wikipedia.org/wiki/Wireless_sensor_network
- [8] (2011, Mar.). Humayun Bakht: Sensor networks and ad-hoc networking [Online].
Available: <http://www.computingunplugged.com/issues/issue200410/00001398001.html>
- [9] (2011, Mar.). WSN Research Group: Sensor Networks [Online].
Available: <http://www.sensor-networks.org/>
- [10] (2011, Mar.). Wikipedia: Sensor Node [Online].
Available: http://en.wikipedia.org/wiki/Sensor_node
- [11] (2011, May). SQLite: Information [Online].
Available: <http://www.sqlite.org/>
- [12] (2011, May). Android Developer: Eclipse ADT [Online].
Available: <http://developer.android.com/sdk/eclipse-adt.html>
- [13] (2011, May). Eclipse IDE: Web Tools [Online].
Available: <http://www.eclipse.org/webtools/>
- [14] E. Lobaton, R. Alterovitz , Y. Simonson and R. Fowler , “Designing Smart Camera

Networks using Smartphone Platforms: A Case Study ,” The Department of Computer Science, Chapel Hill, North Carolina, USA, 2010.

[15] C.K. Harnett, “Open Wireless Sensor Network Telemetry Platform for Mobile Phones,” Sensors Journal, vol. 10, (6), pp. 1083-1084, 2010.

Appendix

Appendix I – User Evaluation Questionnaire

User Evaluation for Sensor Network over Mobile Devices

Introduction

I'm Thisara Rupasinghe and I'm following MCS program at University of Colombo School of Computing. Therefore as my masters' project I'm trying to use mobile device network as ad-hoc sensor network. Since now a day's mobile phones are much advance and equipped with lot of sensor this has become reality. As a result I have implemented the infrastructure that is SOA bases central server and then a prototype mobile application for Android platform. Target android version for this application is Android 2.2 Froyo.

Therefore this is to collect your feedback on that Android mobile application named “Mobile Ad-hoc Sensor Network”. Basically this application is for collect magnetometer reading around all over the world on a user specific location and a user specific time interval. It users the android mobile device network to collect data and uploaded into a centralized server where user has access to view those information. At this time, this application is only available for android mobile phones and only for magnetometer sensor. But this has been implemented in a way that this can be used for other sensors on the mobile devices currently and to be introduced in future.

Therefore your feedback will be really helpful to improve the usability and the usefulness of this application furthermore.

Please note that your personal information and your feedback will safe with us and will not be used for any other purpose or will not be disclosed to any other person/organization.

.....
Thisara Rupasinghe

Date: 08:10:2011

Personal Information

1. Name: _____
2. Age: _____
3. Gender: M | F
4. Computer Literacy: _____
5. Area of education: _____
6. Occupation: _____
7. Computer Literacy: _____
8. Mobile Device: _____
9. Android version: _____

	Description
1	Strongly Disagree
2	Disagree
3	No Comments
4	Agree
5	Strongly Agree

Table: Rating description

	Description	Rate					Comments
UI Related							
1	It is common familiar login screen. Layout and the password recovery function very helpful.	1	2	3	4	5	
2	Register New User screen is a well structured. Layout is quite usual. No sensitive information needed.	1	2	3	4	5	
3	View data screen layout is a simple table. Since there is limited space available only critical information displayed.	1	2	3	4	5	
4	It interactively display all reading data once touch on a record at the view data screen.	1	2	3	4	5	
5	Add Job screen includes all data fields at one screen without scroll. It is a simple	1	2	3	4	5	

	layout design.						
6	Add job screen field validations and error messages are displayed using toasts.	1	2	3	4	5	
7	View job screen has perfect scrollable screen with touch to edit functionality.	1	2	3	4	5	
9	Home screen has good navigation to all functions and space utilized wisely.	1	2	3	4	5	
10	All screens are auto rotating with accelerometer input.	1	2	3	4	5	
11	Scroll bar is available everywhere it is necessary.	1	2	3	4	5	
12	Error messages are given each time there is system exception or users mistakes.	1	2	3	4	5	
Functional							
12	Cannot step into the system without user login. All authenticated users allow to login to the system.	1	2	3	4	5	
13	Sync functionality work properly without any errors or conflicts.	1	2	3	4	5	
14	Email Data function is useful feature to have. User receives nicely formatted data set.	1	2	3	4	5	
15	System does not crash unexpectedly.	1	2	3	4	5	
16	Data collected are exactly align with the specific job added.	1	2	3	4	5	
17	Application response time is quite impressive.	1	2	3	4	5	
18	Most contributed users get priority over others.	1	2	3	4	5	
19	Saved user information used when he returned next time.	1	2	3	4	5	
20	Email functionalities work fine in the system.	1	2	3	4	5	
Hardware Related							
21	When GPS disabled its working with network locations service.	1	2	3	4	5	
22	If all location services are down, it does not try to enable them, but only ask user to enable it to continue.	1	2	3	4	5	

23	After install and run the application, system is stable as it was before.	1	2	3	4	5	
24	Data network is not connected all the time. But it is using periodically.	1	2	3	4	5	
25	Data usage has increased.	1	2	3	4	5	
26	Battery life does not have any noticeable effect. It is as more or less the same as before.	1	2	3	4	5	
27	Storage is not used a lot, since it periodically clear local data after upload into server.	1	2	3	4	5	

User Comments

What are the weaknesses or bugs on the current application?

What are the improvements or features need to be added into the applications?

What do you think about the application? Is it useful for the people who are interested on collecting magnetic field data?

Anything else you would like you to add to make this application a success?

.....
Date

Appendix II – User Feedbacks

Question	User1	User2	User3	User4	User5	User6	User7	User8	User9	User10	User11	User12	User13	User14	User15	Avg Rating
1	5	5	5	4	3	3	3	4	4	4	4	3	5	4	4	4.00
2	4	4	4	3	5	3	2	4	5	3	3	4	4	5	3	3.73
3	2	3	4	2	4	1	5	1	4	3	3	1	2	4	3	2.80
4	3	4	3	3	5	4	4	3	3	4	2	4	2	4	5	3.53
5	5	3	3	4	3	1	4	3	2	4	3	3	5	2	4	3.27
6	4	4	5	3	3	2	3	2	3	5	2	2	3	4	2	3.13
7	4	3	3	4	5	3	2	5	2	3	4	2	4	3	2	3.27
8	2	4	4	3	5	2	4	4	3	2	3	1	4	5	4	3.33
9	5	4	4	5	3	3	4	5	4	4	4	3	5	4	4	4.07
10	4	3	3	2	4	4	2	4	4	3	1	4	4	3	3	3.20
11	3	2	5	2	4	2	2	3	1	3	4	3	3	2	3	2.80
12	4	4	4	5	3	2	4	4	4	5	3	4	5	3	4	3.87
13	3	4	4	3	3	3	2	5	4	4	2	3	3	2	3	3.20
14	4	3	2	5	3	3	5	4	4	2	3	3	1	5	3	3.33
15	5	3	3	2	5	1	4	2	3	3	1	4	2	3	3	2.93
16	1	4	4	4	2	3	5	3	3	4	5	3	3	3	5	3.47
17	3	3	4	1	3	4	1	4	3	3	2	3	3	2	3	2.80
18	4	4	5	2	3	3	4	4	3	1	5	2	4	4	2	3.33
19	2	5	3	1	3	2	3	4	3	2	1	1	2	2	5	2.60
20	3	2	5	4	3	4	2	3	3	4	3	4	3	3	2	3.20
21	4	4	4	2	3	3	3	5	4	4	4	5	3	3	4	3.67
22	5	4	3	4	4	2	4	4	4	3	4	4	5	2	5	3.80
23	3	1	2	5	4	2	4	4	2	3	3	1	3	2	3	2.80
24	5	4	5	4	4	3	4	4	3	5	5	3	3	4	4	4.00
25	4	4	5	5	3	5	3	4	5	4	4	3	5	3	4	4.07
26	3	2	2	4	3	1	4	5	1	3	2	1	3	2	3	2.60
27	1	4	4	5	3	3	2	4	3	4	5	2	3	4	4	3.40
	3.52	3.48	3.78	3.37	3.56	2.67	3.30	3.74	3.22	3.41	3.15	2.81	3.41	3.22	3.48	3.34

Appendix III – Test Cases

Following test cases cover the main functionality of the system.

Test Case ID	1
Tested Component	Login
Tested Date	10/08/11
Description	1. Enter invalid user name & password 2. Try password recovery using email address 3. Check save user name password on first login
Expected Output	1. Display error message. 2. Send email with reset password to matched address. 3. Auto fill user name password from system
Actual Output	1. Proper error message displayed. 2. Reset password received on email. 3. Saved password work after first login.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 1: Test case 1

Test Case ID	2
Tested Component	Subscribe
Tested Date	10/08/11
Description	1. Keep mandatory fields blank and press subscribe. 2. Enter correct information and subscribe.
Expected Output	1. Display error message to enter missing fields. 2. User added successfully.
Actual Output	1. Error message displayed. 2. Subscription successful.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 2: Test case 2

Test Case ID	3
Tested Component	Add Job
Tested Date	12/08/11
Description	1. Leave mandatory fields empty and submit 2. Enter all fields and add job.
Expected Output	1. Display error message to enter missing fields. 2. Job added successfully.
Actual Output	1. Error message displayed. 2. New job added to table.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 3: Test Case 3

Test Case ID	4
Tested Component	View/Edit Job
Tested Date	12/08/11
Description	1. Press on view job button and check whether all user jobs are available. 2. Press on a job and edit all editable fields and update job. 3. Keep mandatory filed empty and update job. 4. Select expired job to edit.
Expected Output	1. All jobs displayed, added by this user. 2. Job updated successfully. 3. Display error message to enter missing fields. 4. Display error message, expired jobs cannot update.
Actual Output	1. Relevant jobs are on the display 2. Job updated 3. Particulate error message displayed. 4. Particulate error message displayed.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 4: Test Case 4

Test Case ID	5
Tested Component	View Data
Tested Date	12/08/11
Description	1. Select the job to view data
Expected Output	1. Job data displayed with limited fields.
Actual Output	1. Data displayed.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 5: Test Case 5

Test Case ID	6
Tested Component	Send Data
Tested Date	15/08/2011
Description	1. Select job and press send data.
Expected Output	1. Data on the select job will be sent to the users email address specified at subscription.
Actual Output	1. Nicely formatted data received on the specified email address.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 6: Test Case 6

Test Case ID	7
Tested Component	Recorder Service
Tested Date	15/08/2011
Description	<ol style="list-style-type: none"> 1. Start service exit application while service running. 2. Stop service. 3. Check data upload on upload timeout. 4. Synchronization on end of job.
Expected Output	<ol style="list-style-type: none"> 1. Service should run on background and record sensor reading. 2. Service should stop recording reading. 3. Data upload should happen on job specified time interval. 4. Once job finished system should synchronize automatically.
Actual Output	<ol style="list-style-type: none"> 1. Start service as expected. 2. Service didn't stop. 3. Data upload never happened. 4. System synchronize at job expired.
Variance	<ol style="list-style-type: none"> 2. Service needed to stopped, but it didn't. 3. Data should be uploaded periodically, but it never happened.
Reason for variance	<ol style="list-style-type: none"> 2. Stop didn't happed due to fault on logic. 3. Coding mistake
Remedial Actions	<ol style="list-style-type: none"> 2. Logic fixed. 3. Code corrected.

Table 7: Test Case 7

Test Case ID	8
Tested Component	Synchronize
Tested Date	15/08/2011
Description	Press sync and check whether, <ol style="list-style-type: none"> 1. Upload all data to server. 2. Get new jobs from server once there is not active job.
Expected Output	<ol style="list-style-type: none"> 1. Data should clear from device and upload to server. 2. Get new active job from server.
Actual Output	<ol style="list-style-type: none"> 1. Data uploaded to server properly. 2. Retrieve new job from server.
Variance	none
Reason for variance	none
Remedial Actions	none

Table 8: Test Case 8

Test Case ID	9
Tested Component	Exception flow
Tested Date	15/08/2011
Description	<ol style="list-style-type: none"> 1. Shutdown back end service and use application. 2. Shutdown back end database and use application. 3. Check business exception for random operations.
Expected Output	<ol style="list-style-type: none"> 1. Application displays error, service not available. 2. Application displays system error. 3. System propagates system exceptions and displays at application.
Actual Output	<ol style="list-style-type: none"> 1. Application fails. 2. Application fails. 3. Business exception display at application level.
Variance	<ol style="list-style-type: none"> 1. Application didn't handle service level system exceptions. 2. Application didn't handle service level system exceptions.
Reason for variance	<ol style="list-style-type: none"> 1. Design issues. 2. Design issues.
Remedial Actions	<ol style="list-style-type: none"> 1. Redesign the exception flow. 2. Redesign the exception flow.

Table 9: Test Case 9

Appendix IV – User Manual

The main functionalities of the system are illustrated below. UI wire-frames are created with most common layouts and therefore it is very comfortable with regular users.

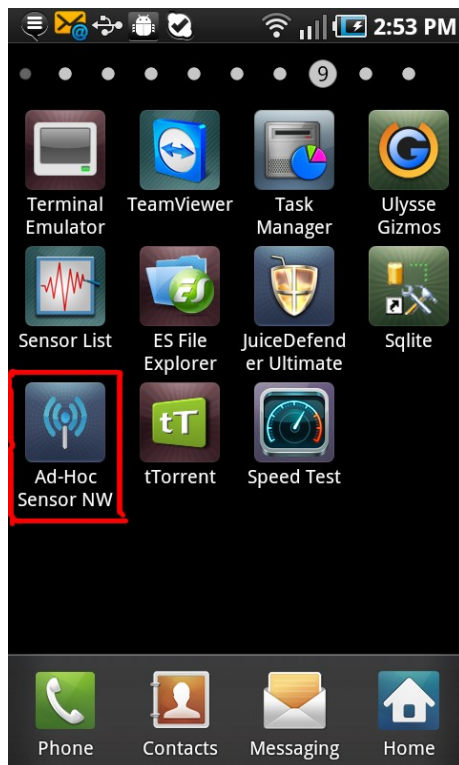


Figure 2: Device Menu

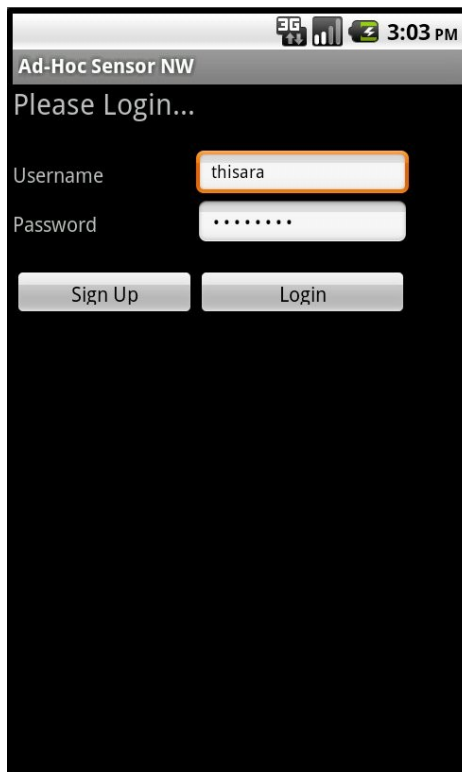


Figure 1: Login Screen

Launch and Login to Application

Once it has been successfully installed on your device there should be a icon with name “Ad-Hoc Sensor NW” as highlighted on the Figure 1 above. Press that icon will launch the application on the device. And it directs user to the login screen where user has to enter user name and the password as shown in figure 2. If you are not an already registered user it allows to subscribing as new user after input user information. As illustrated on figure 3, use name, password, full name and email are necessary to create new account.

If you are registered user you can use your user name and password to login the application. It will direct to the home screen of the application which is look like figure 4

Ad-Hoc Sensor NW

Sign up

Username

Password

Confirm PW

Full Name

E-mail

Figure 3: Sign up Screen

Preferences

Ad-hoc Sensor NW on Magnetometer

Magnetometer Reading : 0.0

X : 0.0 Y : 0.0 Z : 0.0

GPS Location : 0.0, 0.0

Available Sensors

Goldfish 3-axis Accelerometer

TextView

Figure 4: Home Screen

Add New Job

Ad-Hoc Sensor NW

Add Job

Sensor Type

Latitude

Longitude

Dist Range (km)

Start Time ☒

End Time ☒

Frequency (min)

Time Period (hr)

Nodes

Description

Figure 5: Add new job

Ad-Hoc Sensor NW

1	2011-09-10 18:36:19.0	40.6892	-74.0444
2	2011-09-10 18:36:19.0	0.0	0.0
13	2011-11-02 19:50:04.0	6.933079	80.1898

Figure 6: View Jobs

User can add new job to collect data. On home screen there is a button called add button which direct you to the screen at Figure 5. All the fields are mandatory on the add job screen except for the Start time, end time and the description. Description field can leave empty but if you not entering any start time or end time you have to untick the check box next to them. Longitude and latitude fields will automatically filled with the current location, but it can be change as user desired. Distance is the maximum range of a node to collect data. Data recording frequency is captured with minutes. Total time needed to collect data is the “Time Period”. “Nodes” is the number of nodes needs to collect data at a time. By pressing save it will add new job to the queue.

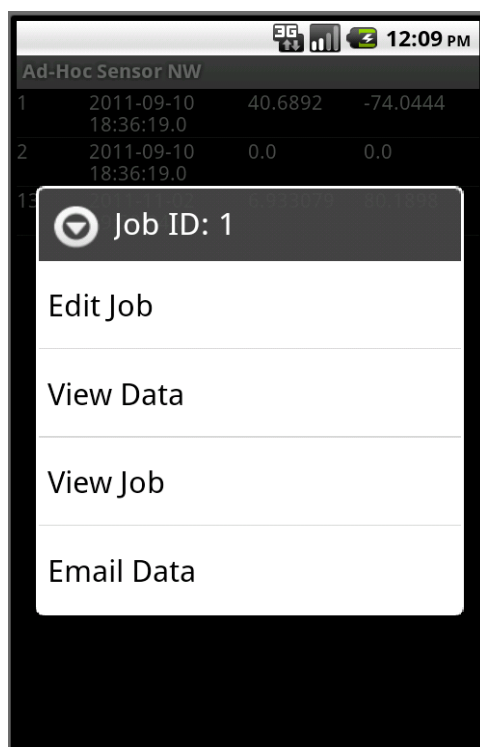


Figure 8: View job Long Press Menu

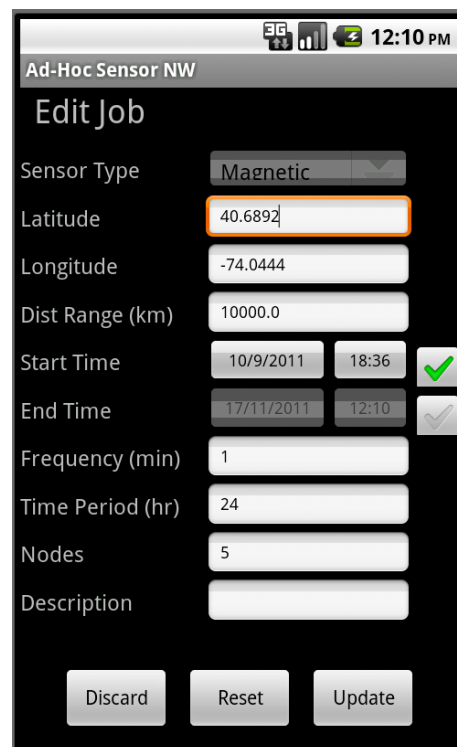


Figure 7: Edit Job

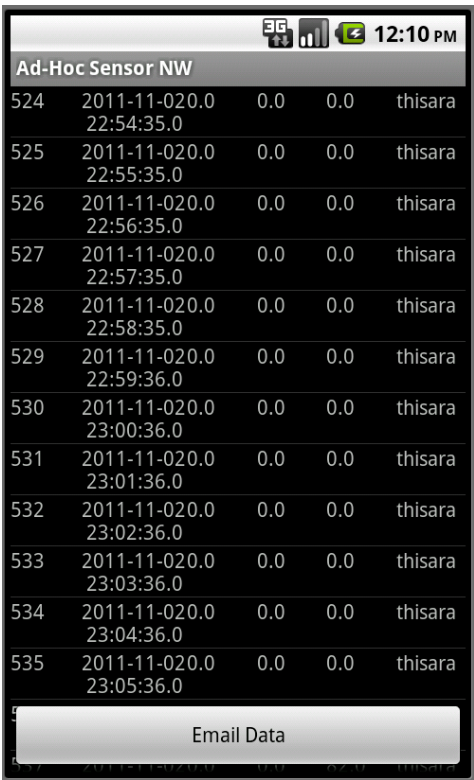
View Jobs

There is button called “View Jobs” on home screen which direct user to the screen as Figure-6. This will list all the jobs added by the user. But it will only display the job id, data, longitude and latitude respectively in a row as per limit permits. And once user presses on a row it will display the all details of the selected.

And once long press on a job it will pop up a menu as shown on Figure-8. There user have 4 options like view job, edit job, view data and email data.

Edit Job

User can edit any job created by him, but which is not expired. Long press on a particular job will pop up a menu and user can select the “Edit Job” to make amendments to the job. Edit job screen looks like the Figure7. User can change all most all fields except the sensor type and base location longitude and latitude. That is changing the sensor type means it will be totally new job and changing the base location means all the data collected so far will be useless after that. Therefore it will not allow to changing those fields.



Ad-Hoc Sensor NW				
524	2011-11-020.0 22:54:35.0	0.0	0.0	thisara
525	2011-11-020.0 22:55:35.0	0.0	0.0	thisara
526	2011-11-020.0 22:56:35.0	0.0	0.0	thisara
527	2011-11-020.0 22:57:35.0	0.0	0.0	thisara
528	2011-11-020.0 22:58:35.0	0.0	0.0	thisara
529	2011-11-020.0 22:59:36.0	0.0	0.0	thisara
530	2011-11-020.0 23:00:36.0	0.0	0.0	thisara
531	2011-11-020.0 23:01:36.0	0.0	0.0	thisara
532	2011-11-020.0 23:02:36.0	0.0	0.0	thisara
533	2011-11-020.0 23:03:36.0	0.0	0.0	thisara
534	2011-11-020.0 23:04:36.0	0.0	0.0	thisara
535	2011-11-020.0 23:05:36.0	0.0	0.0	thisara

Email Data

Figure 10: View Data Screen

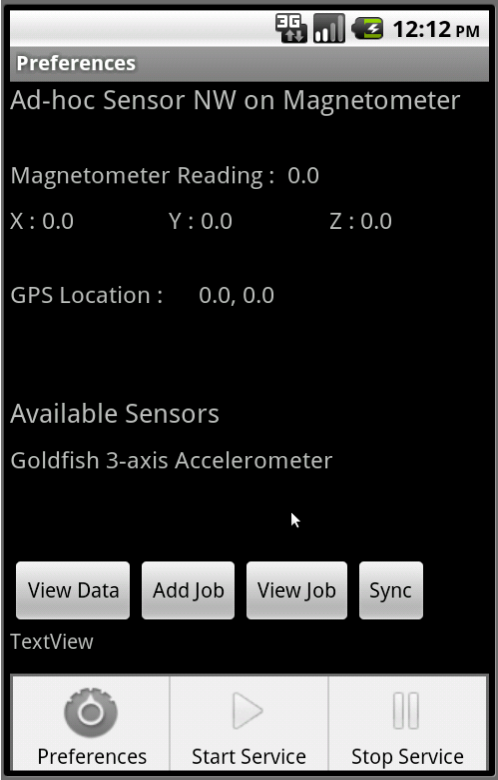


Figure 9: Application Menu

View Data

Long press on “View Jobs” user can select the view data option which direct user to figure9. It will display data id, data time, longitude, latitude, reading and the uploaded user respectively. “Email Data” button on the bottom of the screen will email those data on a nicely formatted CSV file to him email address specified on his user account.

Appendix V – Project Information

1. This project is uploaded into Google Code and all the information about the project can be found on the project home using following URL.

<http://code.google.com/p/mobile-ad-hoc-sensor-nw/>

2. This source code of the android application and the back end service is shared using Google Code project hosting repository. Therefore anyone who is interested, can check out the source code from following URL.

<http://mobile-ad-hoc-sensor-nw.googlecode.com/svn/trunk/>

3. Application .apk installation file for android 2.2 based mobile devices can be downloaded from below location. But this is beta release and all issues you face can be recorded with in Google code project. That will really impotent to improve this system furthermore.

<http://code.google.com/p/mobile-ad-hoc-sensor-nw/downloads/list>



4. Back end service is hosted on third party hosting environment. It can be access using following URL.

<http://mobilesensornw.hostjava.net/Axis2WS1.4/>