Letterkenny Institute of Technology

A thesis submitted in partial fulfilment of
the requirements for the Master of Science in Computing in
Enterprise Applications Development Letterkenny Institute of Technology

Investigation of Indoor Positioning based on WLAN 802.11

Author:
Eugene Ferry

Supervisor:
John O Raw

Submitted to the Higher Education and Training Awards Council (HETAC)

August 2013

Declaration

I hereby certify that the material, which I now submit for assessment on the programmes of study leading to the award of Master of Science in Computing in Enterprise Application Development, is entirely my own work and has not been taken form the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution.

Signature of candidate: _____ Date: _____

I hereby certify that all the unreferenced work described in this thesis and submitted for the award of Master of Science in Computing in Enterprise Application Development, is entirely the work of Eugene Ferry. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution.

Signature of Supervisors _____ Date _____

Acknowledgements

This thesis would not have been possible without the support of many people and I would like to thank all those who assisted me in its completion.

I would like to thank my family and friends who have provided me with a constant source of support and encouragement throughout this process. I would like to thank all who assisted me, especially my supervisor Mr John O'Raw whose expertise and guidance has been invaluable. Finally I would like to thank all the academic staff at Letterkenny Institute of Technology for all their help and guidance over the years.

Abstract

The need for location based services has dramatically increased within the past few years, especially with the popularity and capability of mobile device such as smart phones and tablets. The limitation of GPS for indoor positioning has seen an increase of indoor positioning based on Wireless Local Area Network 802.11.

This thesis reviews the various different techniques used by applications to determine one's location through the measurement of Wi-Fi signals. It particularly focuses on the Cisco Context-Aware Mobility which provides a Real Time Location System solution based on Wi-Fi. It details the implementation of an Android application, developed to communicate with the Cisco Context-Aware Mobility to visually display the location of the mobile device. The application was tested in a production environment. Limitations in the production environment along with the diagnostic capabilities of the Context-Aware Mobility were identified.

Abbreviations

AoA	Angle of Arrival
AP	Access Point
API	Application Programming Interface
COO	Cell of Origin
DIP	Density Independent Pixel
DoA	Direction of Arrival
DPI	Dots per Inch
GPS	Global Position System
HTTP/S	Hypertext Transfer Protocol/Secure
K-NN	K-Nearest Neighbour
LWAPP	Lightweight Access Point Protocol
MSE	Mobility Service Engine
NBC	Naive Bayes Classifier
NLOS	Non Line of Sight
NMSP	Network Mobility Service Protocol
RF	Radio Frequency
RMI	Remote Method Invocation
RSS	Received Signal Strength
RSSI	Received Signal Strength Indication
RTLS	Real Time Location System
RTT	Round Trip Time
SAML	Security Assertion Markup Language
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol
SSID	Service Set Identifier
SVM	Support Vector Machine
TDoA	Time Difference of Arrival
ToA	Time of Arrival
WCS	Wireless Controller System

WLAN	Wireless Local Area Network
WLC	WLAN Controllers
WSDL	Web Services Description Language
XML	Extensible Markup Language

Table of Contents

Table of Figures

Code Listings

1 Introduction

1.1 Background and Objective

Over the last few years the popularity of Real Time Location Systems (RTLS) has increased. This increase has created a demand for indoor positioning systems (Hossain et al., 2007). The availability of indoor positioning systems not only provides the location of the user and its surrounding environment but it also provides businesses with the opportunity to advertise or promote their business, their latest deals and much more.

The most popular positioning system used today is GPS, however GPS does not work well indoors as a clear line of sight to the satellites is not always attainable due to obstructions (Ching et al., 2010). This has led to the introduction of indoor positioning based on WLAN 802.11. According to (Hossain et al., 2007) indoor location systems should be cost-effective, involve a short training phase and provide a high level of accuracy. The ever increasing popularity of smartphones and tablets has also made these RTLS more accessible to the general public.

1.2 Problem

Over the last few years there have been many different types of RTLS developed using Wi-Fi. The majority of enterprises, universities and shopping centres deploy WLAN as a means of communications and services, hence the reason Wi-Fi is used in RTLS. The fact that most mobile devices now contain Wi-Fi radios also has a major factor in choosing Wi-Fi.

Wi-Fi based RTLS can be developed using a number of different techniques to measure Wi-Fi signals to determine ones location. The level of accuracy provided by these techniques can vary depending on the chosen environment. This thesis will discuss some of these techniques and determine which would be most suitable for a university environment.

1.3 Preliminary Analysis

There have been many proposed solutions to better improve the accuracies of some of these techniques. One proposed solution involved applying different formula in order to account for multipath interference and so forth. Another allowed the user to inform the RTLS of the accuracy of its location positioning by stating if the proposed location is in fact correct or not. One of these solutions is Cisco's Context-Aware Mobility. The Context-Aware Mobility provides real time positioning by measuring Wi-Fi signal and incorporating location calculations to provide a high level of accuracy.

1.4 Solution Proposal

Each solution has advantages and disadvantages. This thesis looks to ascertain if the use of the Cisco Context-Aware Mobility is a viable solution. Deploying it on an Android device, this thesis will determine if the Cisco Context-Aware Mobility is capable of providing a high accuracy RTLS suitable for distribution and that will complement the needs for SMEs, large enterprises, universities and shopping centres.

Following initial research, an Android application was developed using the Cisco Context-Aware API which made use of the MSE located within the LYIT. This application investigated the accuracy and usability of the Cisco Context-Aware Mobility and assessed its capability as a RTLS.

A hypothesis of this research was deemed to be: Cisco's Context-Aware Mobility is not capable of providing a high accuracy RTLS.
To evaluate this hypothesis a number of tests were carried out on the Android application within the LYIT.

1.5 Outline of Report

- Chapter 2: Discusses Wi-Fi measuring techniques and the Cisco Context-Aware Mobility
- Chapter 3: Discusses application requirements and software design, including design diagrams

- Chapter 4: Discusses the implementation of the application
- Chapter 5: Analyses the results of the testing and discusses its findings
- Chapter 6: Concludes on overall findings and suggests further work

2. Survey

2.1 Measuring Wi-Fi Signals

With the popularity of Wi-Fi increasing dramatically over the last few years, so has the need and use of Real Time Location Systems (RTLS). The availability of WLAN Access Points (APs) deployed almost everywhere including universities, hotels and shopping centres to name but a few, allows RTLS to calculate ones location using a number of different techniques and approaches. These approaches differ in terms of the measurement techniques used to sense and measure the position of the device. There are also a number of factors that influence the technique that RTLS choose to implement such as cost, accuracy, performance, and environment. The following section expands on some of the techniques used.

2.1.1 Cell of Origin

Cell of Origin (COO) or Nearest Cell is one of the simplest forms of location tracking. Although simple to implement, this techniques makes no attempt to resolve the exact position of the mobile device other than the cell that it has been located in and therefore lacks accuracy (Cisco, 2010d). This is mainly due to the large size of the cells (Retscher and Fu, 2010). The most basic of cell networks contain transmitters that transmit evenly in all directions creating a circle. Due to the fact that circles do not tessellate well, the network cells are usually calculated as hexagons (Jagoe, 2002) as shown in Figure 2.1.1.

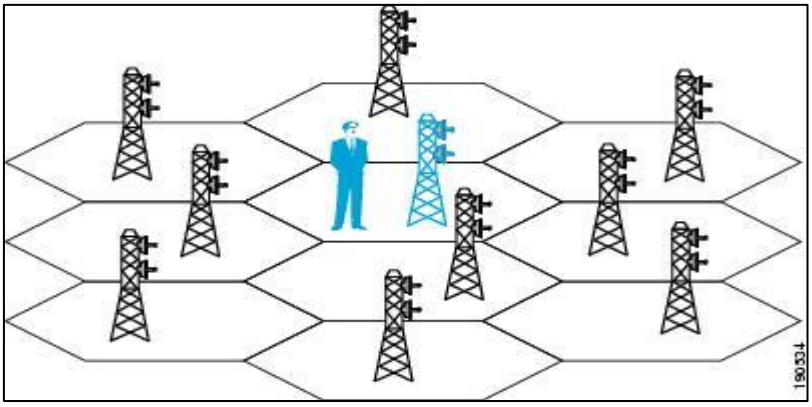

Figure 2.1.1: Cell of Origin (Cisco, 2008b)

This technique is used due to its relative ease of implementation which negates the need for complex algorithms thus providing fast positioning performance. The accuracy of this

technique is based on the density of the tower, with a higher density proving better accuracy due to the coverage area being smaller (Ching et al., 2010). However most towers have low densities which mean large coverage areas that only provide accuracies of hundreds of meters and over. Like all positioning techniques it does have its disadvantages. One of its main downfalls is its lack of accuracy and due to various reasons mobile devices can be associated to far away cells, even though there are closer cells at hand. This incorrect association to cells can really affect the accuracy in multi-story buildings where floor to floor cell overlap occur (Cisco, 2008b).

There are procedures that can be taken to increase the accuracy of the nearest cell technique, such as using the Received Signal Strength Indication (RSSI) provided by the cells and noting the highest signal strength. The mobile device is then associated with the cell that records the highest signal strength (Cisco, 2008b) as shown in Figure 2.1.2 thus increasing the possibility of selecting the correct nearest cell. This technique is good for low cost, good performance and non-critical tracking, but it would not be suitable for users requiring finer accuracy.

Figure 2.1.2: Highest Signal Strength (Cisco, 2008b)

COO can provide accuracies of 100m up to 1km for urban areas and 35km for rural areas providing a response time between 2 to 5 seconds (Jagoe, 2002), (Kos, Grgic and Kitarovic, 2007). Indoor APs can provide a far greater density than that of a mobile cell tower, hence, the COO can provide greater accuracy than indoor GPS (Ching et al., 2010). According to (Jagoe, 2002) COO was the most common method used by positioning systems in 2001.

2.1.2 Lateration

2.1.2.1 Time of Arrival

Time of Arrival (ToA) is based on the time of arrival of a signal sent from a mobile device to two or more sensors (Kos, Grgic and Kitarovic, 2007), (Llombart, Ciurana and Barcelo-Arroyo, 2008).

"*The ToA technique requires very precise knowledge of the transmission start time(s), and must ensure that all receiving sensors as well as the mobile device are accurately synchronized with a precise time source*" (Cisco, 2008b). ToA determines the distance between the mobile device and the receiving sensors by calculating the time it takes for the signal to travel between them (Kos, Grgic and Kitarovic, 2007). The calculated distance is used as a radius to create a circular plot around each respective receiving sensor, with the mobile device believed to be placed somewhere along each plot (Llombart, Ciurana and Barcelo-Arroyo, 2008). The location of the mobile device is pinpointed where the plots intersect as shown in Figure 2.1.3 or the intersecting area.

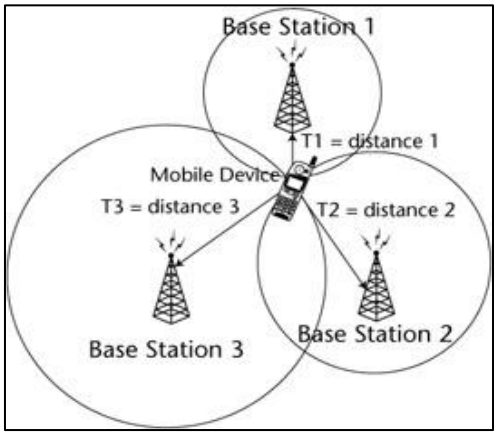

Figure 2.1.3: Time of Arrival (Etutorials, 2013)

Cases in which three sensors are used are referred as ToA tri-lateration and cases in which four or more sensors are used are referred to as ToA multi-lateration. ToA can also be constructed in 3D using spheres instead of circular plots, which is the technique incorporated by Global Positioning Systems (GPS).

(Liu and Yang, 2011), (Llombart, Ciurana and Barcelo-Arroyo, 2008) State that RF trilateration requires at least three APs to calculate distances and determine location.

However, they also state that high positioning accuracy using trilateration can be difficult to achieve due to multipath interference and the difficulty of obtaining detailed coordinates of all the APs. The difficulty in obtaining accurate coordinates of APs in buildings can be due to building structures, network services and mobile services including attenuation, multipath, occlusion and reflection (Liu and Yang, 2011) (Brown and Dunn, 2011).

ToA systems are flexible, non-complex and are highly adaptive to changing environments (Llombart, Ciurana and Barcelo-Arroyo, 2008). According to (Yu et al., 2012) ToA is more accurate than Received Signal Strength (RSS) techniques and more practical than Time Difference of Arrival (TDoA) and Angle of Arrival (AoA) which are discussed later in this chapter.

However, due to the need for precise time synchronization very small errors in time can result in very large positioning errors. According to (Llombart, Ciurana and Barcelo-Arroyo, 2008) a time inaccuracy of 1 microsecond can lead to a distance error of up to 300m. ToA based methods suffer when utilised by a large number of people due to the traffic generated by the system. This is due to the ToA- based system generating traffic when calculating distances (Llombart, Ciurana and Barcelo-Arroyo, 2008).

Using the ToA technique accuracies of up to 1 to 2 meters can be achieved on a WLAN network when an investment in dedicated hardware (synchronized clocks) is made (Ciurana et al., 2010). However (Ciurana et al., 2010) suggests a software only approach by using the CPU clock of the mobile device to measure the Round Trip Time (RTT) that is required to calculate distances.

2.1.2.2 Time Difference of Arrival

Time Difference of Arrival (TDoA) is based on the time of arrival of a signal sent from a mobile device to three or more sensors (Morgan, 2009), (Abdul-Latif, Sheperd and Pennock, 2007). However, unlike ToA, TDoA only requires the receiving sensors to have synchronized time sources and not the mobile device. This means that TDoA does not require knowledge of the transmission start time (Cisco, 2010c).

TDoA is calculated using hyperbolic tri-lateration which requires at least three synchronized receiving sensors. As shown in Figure 2.1.4 the time taken for a mobile device to transmit a signal to receiving sensor A and receiving sensor B is recorded. The time difference of the transmission is calculated by subtracting the two times. Therefore, if sensor A receives the transmission before sensor B then sensor A is closer. Estimated TDoAs are converted into distance measurements which result in a set of nonlinear hyperbolic curves; where the curves intersect indicate the position of the mobile device (Elkamchouchi and Mofeed, 2005).

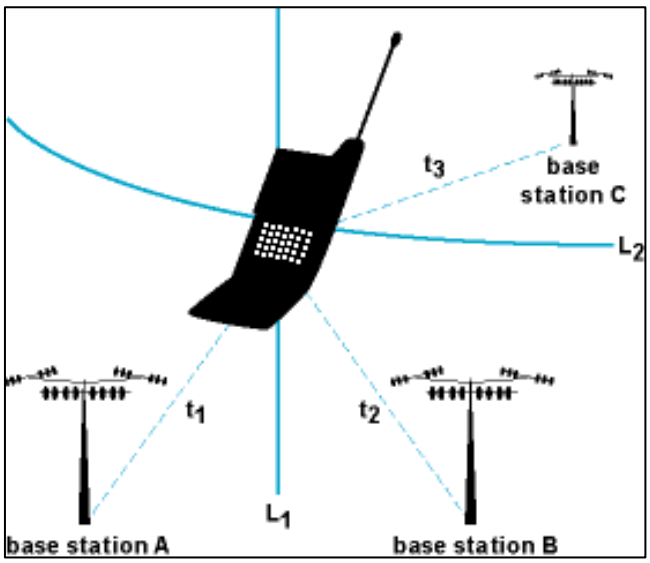

Figure 2.1.4: Time Difference of Arrival (Driscoll, 2013)

According to (Cisco, 2008c) TDoA systems can obtain greater accuracy with greater bandwidth and transmit power. It also states that indoor TDoA and ToA systems are best suited to large, open buildings that contain high ceilings and little obstructions allowing adequate clearance between the ceiling and building contents.

TDoA techniques have poor accuracy due to technical limitations and propagation environments, but its poor accuracy is mainly due to Non Line of Sight (NLOS). NLOS is when a direct line of sight between the mobile device and the base station (AP) is obstructed by obstacles (Kim, Lee and Park, 2008). NLOS propagation error results in the transmitted signal

taking longer to arrive at the receiver as the signal is reflected and diffracted which results in a greater travelling distance (Kim, Lee and Park, 2008).

A solution to the NLOS propagation error is to apply TDOA error modelling at each base station (Kim, Lee and Park, 2008). It concluded that using the error modelling algorithm the NLOS error can be mitigated by 45 to 77%.

2.1.2.3 Received Signal Strength Indication

Received Signal Strength Indication (RSSI) is another technique which uses either a mobile device or a receiving sensor to measure the signal strength of the transmission in order to determine location. Typically in an RSSI environment, a location server will know the locations of all the receiving sensors (measurements devices). The receiving sensors measure the signal strength from a transmission sent from a mobile device and forward it on to a calculation engine. The calculation engine then locates the mobile device using triangulation or other known techniques as shown in Figure 2.1.5 taking into account the transmitter output power, cable losses and antenna gains (Cisco, 2010c).

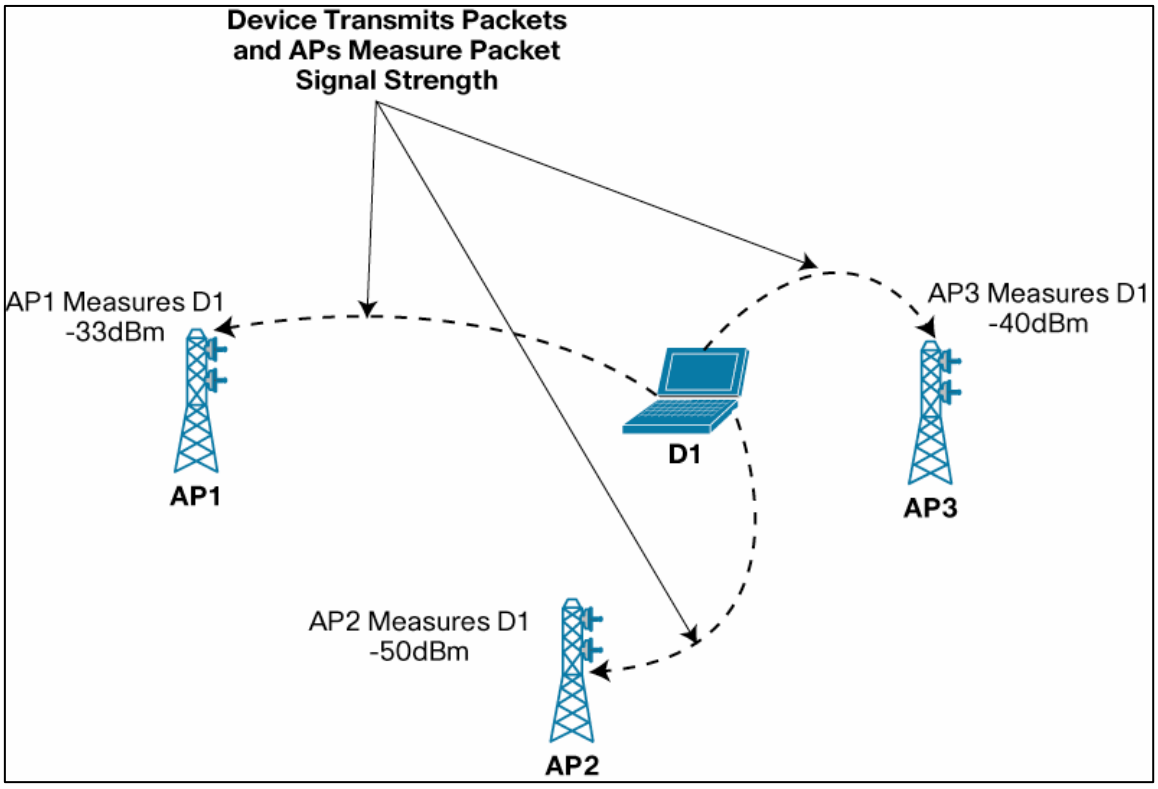

Figure 2.1.5: RSSI (Cisco, 2008c)

It is also assumed in direct line of sight situations, that if two signals are received by the receiving sensor from two mobile devices, then the mobile device with the lowest signal strength is the furthest away. In most situations a direct line of sight is not obtainable as environments such as warehouses and offices contain many obstructing objects and reflective surfaces (Cisco, 2010c). This means that the direction of the signal sent from the mobile device to the receiver is not direct. Instead it is reflected off a surface such as a wall or ceiling, meaning that the signal has to travel further thus producing lower signal strength when detected by the receiver (Cisco, 2010c). This multipath interference can impact on the accuracy of the positioning system as it believes that the mobile device is further away than it actually is.

The RSSI technique is an attractive one as it does not require any specialised hardware at either the mobile device or the receiver, making it a cost effective choice especially for those of 802.11 WLAN systems wanting to offer location systems. The pure RSSI technique does not implement measures to account for attenuation and multipath, hence resulting in loss of accuracy which leads to unacceptable results. Therefore it is best suited to controlled environments where a direct line of sight is attainable (Cisco, 2010c).

2.1.3 Angulation

2.1.3.1 Angle of Arrival

Angle of Arrival (AoA) or Direction of Arrival (DoA) is a technique in which the location of the mobile device is located by calculating the angle of incidence of the signals received by the receiving sensors. In order to measure the angles an antenna array is required (Niculescu and Nath, 2003), (Abdul-Latif, Sheperd and Pennock, 2007), (Drane, Mac Naughton and Scott, 1998).

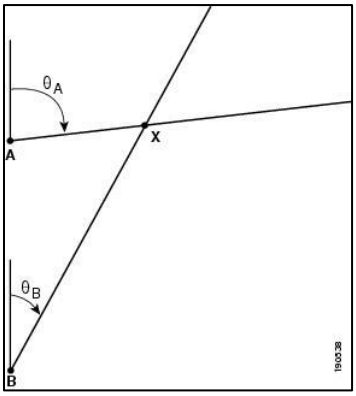

Figure 2.1.6: Angle of Arrival (Cisco, 2008b)

The location of the mobile device is determined by the intersecting lines in which the signal is received by at least two receiving sensors (Abdul-Latif, Sheperd and Pennock, 2007) as shown in in Figure 2.1.6. Improved accuracy can be achieved from three or more receiving sensors.

In situations where a clear line of sight is attainable between the mobile device and the receiving sensors, mechanical antennas at the receivers adjust to point to the highest signal strength received. These antennas can be used to calculate the direction in which the signal is received (angle of incidence) determining the lines of bearing (Cisco, 2008b). It is where these lines intersect that the position of the mobile device is located.

AoA techniques have previously been used in the cellular industry in order to determine the location of mobile phone users in cases of emergency. This was achieved using numerous towers to calculate the angle of arrival of the signal and by converting this information to longitude and latitude coordinates. However the AoA technique struggles for accuracy when a clear line of sight is not attainable, for instance, when challenged by multipath interference such as reflection from surrounding objects (Wong, Klukas and Messier, 2008). According to (Abdul-Latif, Sheperd and Pennock, 2007) a small error in angle measurement can result in a large error in positioning.

Increasing distances between the mobile device and the AP results in accuracy degradation. However, within an indoor WLAN environment the APs are close to the mobile device so accuracy is not affected (Wong, Klukas and Messier, 2008). (Wong, Klukas and Messier,

2008) tested two algorithms Maximum likelihood (ML) and Space-Alternating Expectation-Maximization (SAGE) to determine the performance of their system in which it concluded that SAGE was not suitable for determining the AoA of a mobile device using indoor APs. However it states that using the ML algorithm has the potential of determining position with less than 2m accuracy.

2.1.4 Location Patterning

Location Patterning or Fingerprinting is a technique where no specialised hardware is required by either the mobile device or the receiving sensors allowing it to be fully implemented in software (Chang, Rashidzadeh and Ahmadi, 2010), (Lui et al., 2011). The Location Patterning technique makes assumptions that each mobile device location will contain a unique Radio Frequency (RF) signature and also that each building or floor will contain unique propagation characteristics meaning no two buildings or floors should be the same in terms of RF pattern recognition (Cisco, 2006). Although most commonly Location Patterning is based on RSSI it can be implemented to include ToA, TDoA and AoA techniques.

The Location Patterning technique consists of two main phases: Calibration phase and Operation phase.

2.1.4.1 Calibration Phase

The calibration phase involves walking around the chosen environment with a mobile device and using the receiving sensors (APs) to measure the signal strength of the mobile device to accumulate the required data (Del Mundo et al., 2011), (Chang, Rashidzadeh and Ahmadi, 2009).

In order to acquire this data, a set of reference points are placed on a map of the environment to precisely mark where the data should be recorded. At each marker an array of RSS values associated with the mobile device are recorded in a database along with the coordinates of the reference point (Shin et al., 2010). This is usually referred to as a radio map. The array size of the RSS values is determined by the number of receiving sensors

detected by the mobile device. Figure 2.1.7 shows a simple representation of the calibration phase.

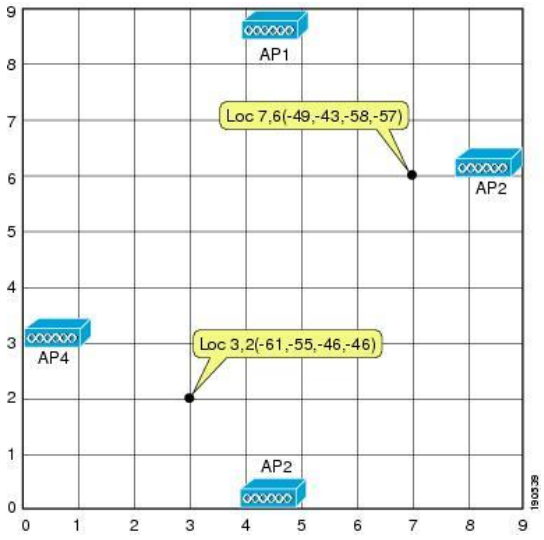

Figure 2.1.7: Calibration Phase (Cisco, 2006)

According to (Cisco, 2006) the recorded signal strength of a mobile device will vary and due to this, multiple samples are required at each reference point during the calibration phase. A method used to simplify this procedure is to record a mean value of all the measurements taken at each specific location for a specific mobile device, meaning that the array of RSS values becomes an array of mean RSS values.

2.1.4.2 Operation Phase

The operation phase consists of the receiving sensors forwarding the signal strength received from the mobile device to a calculation engine. The calculation engine then performs complex algorithms with the use of the radio map, allowing it to retrieve the location of the mobile device (Koo and Cha, 2012).

There are two ways to compare fingerprints: deterministic and probabilistic.

- Deterministic: Calculates the difference between the current reference fingerprint and all those recorded during the calibration phase. The fingerprint in the radio map with the smallest difference from the reference fingerprint is selected as the mobile devices position (Ching et al., 2010).

- Probabilistic: Calculates the probability that the user is at a specific reference point. This method requires more sampling at each reference point during the calibration phase (Ching et al., 2010). During the operation phase, the system calculates the probability of the mobile device being at each reference point in the radio map. The reference point with the highest probability is selected as the mobile device's position (Meng et al., 2011).

The most common algorithms used include K-Nearest Neighbour (k-NN), Naive Bayes Classifier (NBC) or Support Vector Machine (SVM). A comparison carried out by (Del Mundo et al., 2011) found that k-NN had an accuracy of 0.7294 within 1.96 meters, NBC achieved an accuracy of 0.8173 within 1.96 meters and SVM achieved an accuracy of 0.8267 within 1.96 meters. It concluded that SVM outperformed NBC in terms of accuracy and found that k-NN was the least accurate of the three.

Location Patterning techniques perform well in indoor environments and provide good accuracy when the mobile device is detected by at least three receiving sensors (Cisco, 2006). In order to increase the accuracy, the mobile device must be detected by six to ten receiving sensors which in turn provides an accuracy of up to five meters. A benefit of using Location Patterning techniques is that it can be used with existing infrastructure such as 802.11 WLANs, and it also accounts for attenuation and multipath interference during the calibration phase. According to (Del Mundo et al., 2011) Fingerprinting techniques perform better than that of cell based (Cell of Origin), lateration (ToA, TDoA) and angulation (AoA) techniques in terms of positioning accuracies.

However, a disadvantage to this technique is the need for a high number of APs in tight spacing. Another drawback to this technique is the calibration phase. The radio maps used tend to be very specific to the area in which they were created for regardless of the structure of the buildings. The chances of identical buildings having the same radio maps are very low which in turn provide little re-use of the radio maps. As mentioned earlier the results of the calibration phase are likely to vary, especially overtime as changes occur that affect RF propagation such as changing environments and so forth. (Ching et al., 2010). This requires re-calibration in order to achieve a consistence level of accuracy, with complete re-

calibration required twice annually in some environments (Ching et al., 2010), (Meng et al., 2011).

There have been many suggestions to improve the accuracy of the Location Patterning technique. Some of these look at improving the calibration phase by allowing the end user to update the radio map by correcting the actual position (Gallagher et al., 2010), (Koo and Cha, 2012) where (Kim, Chon and Cha, 2012) uses an autonomous calibration phase. Others look to combine a number of techniques such as GPS, RSS, accelerometers and digital compasses to help improve accuracies (Chun et al., 2011), (Chon and Cha, 2011), (Kim, Shin and Cha, 2012).

2.1.5 Cisco RF Fingerprinting

RF Fingerprinting uses the Received RSSI approach to provide indoor location performance which only Location Patterning techniques were capable of. It works in a very similar way to Location Patterning, but with more speed, efficiency and improved accuracy.
"*RF Fingerprinting significantly enhances RSS lateration by using RF propagation models developed from radio propagation data gathered directly from the target environment or environments very similar to it*" (Cisco, 2008a).

The calibration phase for RF Fingerprinting is the same as that in Location Patterning in which the coordinates of the reference point is stored along with the mobile devices RSSI from three or more APs. The accumulated calibration data is processed and used to build an RF propagation model which calculates path loss and shadow fading standard deviation to account for propagation discrepancies in the current environment. RF Fingerprinting also allows the use of models created during the calibration phase or pre-packaged RF propagation models.

According to Cisco, the RF Fingerprinting technique performs better than other techniques such as triangulation or RSS lateration as these techniques make no attempt to account for attenuation or any environmental considerations. It also states that RF Fingerprinting

applies 'statistical analysis techniques' on the calibration data allowing it to rule out unconvincing possibilities and further improve location accuracy (Cisco, 2008a).

RF Fingerprinting does not require the same calibration effort as Location Patterning nor does it require re-calibrating as often as Location Patterning. Unlike Location Patterning, RF Fingerprinting can use the same RF model in similar environments or it can use pre-packaged RF models which allow for swift deployment.

2.2 Cisco Context Aware Mobility

Most Context-Aware Mobility solutions used today are Wi-Fi based due to the deployment of WLAN by the majority of enterprises and also that most mobile devices contain Wi-Fi radios. Context-Aware Mobility solutions enable enterprises to retrieve and use contextual information regarding mobile assets to help optimise and change business processes. Information can be collected on any mobile assets involved in the business process, including that of devices, people or products. Cisco's Context-Aware Mobility Service is used to capture such information and more, such as location, temperature, availability and applications used, whilst also providing the ability to integrate this information with other systems to better enhance business functionality. Cisco's Context-Aware API allows users to create customised context-aware programs that interact with Mobility Service. The following section expands on the architecture and functionality of Cisco's Context-Aware Mobility API.

2.2.1 Architecture

The Context-Aware Mobility Service can be used by either the 3300 Series Mobility Service Engine (MSE) or the 2700 Series Wireless Location Appliance which in turn communicate with the Wireless Control System (WCS).

2.2.1.1 Wireless Control System

The WCS is a WLAN management tool that provides the ability to manage network controllers through a web based user interface. It allows users to configure and monitor controllers and access points. It also provides the ability to edit building floor plans such as

changing the types of walls, locations of APs and also provide heat maps of the selected floors (Cisco, 2011).

2.2.1.2 Mobility Service Engine

The MSE communicates with WLAN Controllers (WLC), the WCS and the Location Server. "*MSE is the integration point for applications through well defined APIs that help in Asset tracking for location.*" (Cisco, 2012b). It allows integration between Cisco technologies and their partners and it is used to gather statistics and data of clients of WLC and WCS. Applications such as the WCS or partner application wanting to use MSE services can do so by communicating through APIs (Context-Aware API) using XML/SOAP. The Mobility Service Engine provides real time location tracking and historical trends of rogue devices, Wi-Fi clients and RFID tags allowing business to gather a better understanding of client movement and habits.

The Context-Aware Mobility Service runs on the MSE and can be combined with other mobility services to provide even more functionality. It can also be deployed across multiple MSE providing a scalable network design (Cisco, 2010a). Figure 2.2.1 shows the architecture of the MSE.

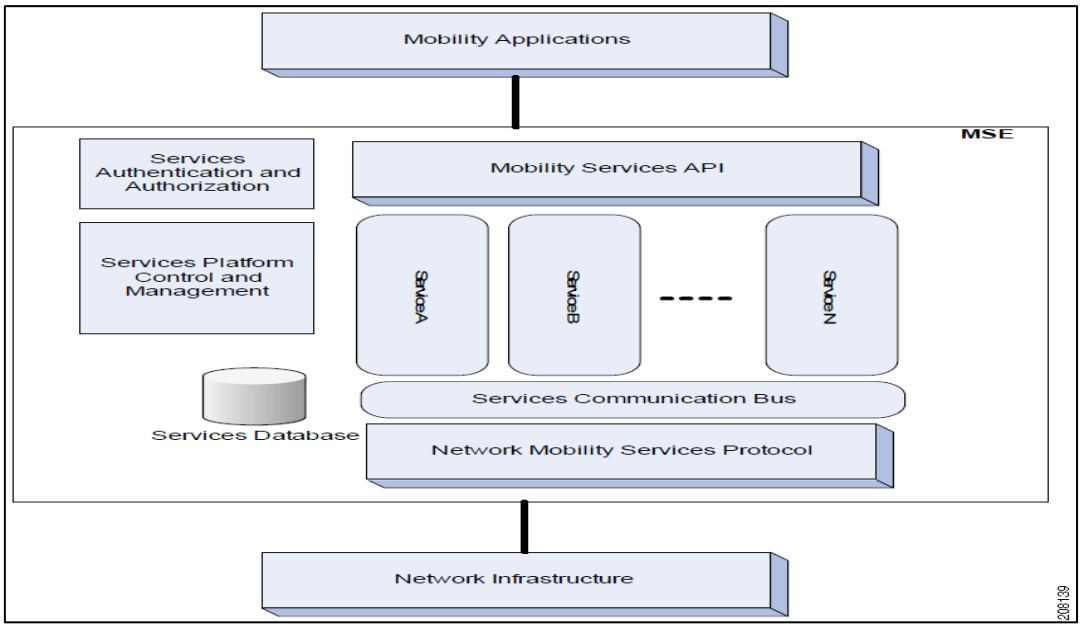

Figure 2.2.1: MSE Architecture (Cisco, 2010a)

In order to track the location of a mobile device, the MSE running the Context-Aware Mobility Service gathers information from all of its controllers and their associated APs in the chosen environment. As the Context-Aware Mobility Service runs on the MSE instead of a single controller, it can aggregate all of the AP measurements from numerous controllers (Cisco, 2010a). This also ensures that the location calculations can be performed at rapid speeds (a few seconds), which is necessary for active location tracking information to be of any use to its consumers.

According to (Cisco, 2012c) the Cisco 3310 Mobility Service Engine can cater for up to 2,000 clients and tags combined and the Cisco 3350 Mobility Service Engine can cater for up to 25,000 clients and tags combined. It supports Wi-Fi 802.11 a/b/g/n networks.

2.2.2 Context-Aware Mobility API

The Context-Aware Mobility API provides customers and partners the ability to build and use customised applications that interface with the MSE. The Context-Aware Mobility API communicates with the MSE through SOAP (Simple Object Access Protocol)/XML (Extensible Markup Language) over HTTP/S (Hypertext Transfer Protocol/Secure) as shown in Figure 2.2.2.

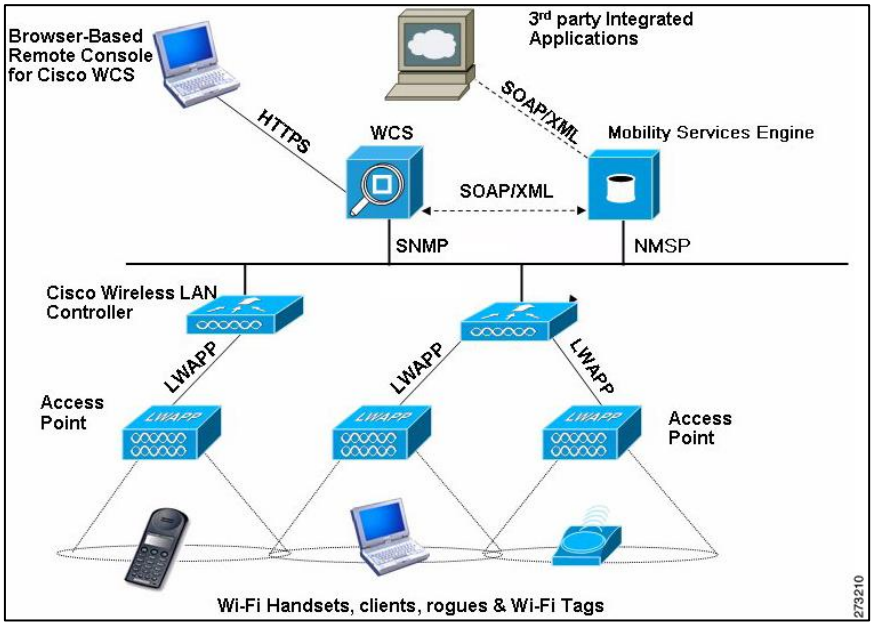

Figure 2.2.2: Context-Aware Mobility (Cisco, 2010d)

This involves a request being sent from the client to the server to get or set information and a response being sent from the server to the client in response to the request. It can also involve notifications being sent from the server to the client asynchronously when a specified event is detected by the server (Cisco, 2010d).

2.2.2.1 Context-Aware Mobility API Modules

The Context-Aware API consists of a number of modules (Cisco, 2010d) which include server connection, Server Administration, Network Design Interface, Tag, Rogue Clients, Rogue Access Point, Presence, Asynchronous Notifications and the Mobile Station module.

The server connection API is used to authenticate clients against the server database and to establish a business session. The business session identifies the user's privileges and enables them to login and logout from the server. The server administration API is used to retrieve the context-aware server information such as the server details and objects that have changed within a certain date. The network design interface is used to obtain a specific network design object or a list of network design objects contained in the location server. The tag API is used to monitor and control up to 2500 active RFID tags. This includes the ability to add, edit and delete tags, retrieve the location of specific tags or a list of tags, and also retrieve tag history and statistics.

The rogue client API is used to identify and control rogue clients located on the WLAN. This provides the ability to retrieve rogue client information, location history and to delete a rogue client from the server. The rogue access point API is used to identify and control rogue access points. It provides the same functionality as that for the rogue clients. The presence API provides the ability to configure parameters such as location resolution type, location format and response encoding, allowing users to retrieve client locations in encoded format from the MSE.

Another module is the asynchronous notification API, which is used to register and set asynchronous events within the server. Notifications are created based on a set of triggers that are set against a tracked asset within the system. Triggers can be set to notify users on

a number of events. If an asset is within a certain area, building or floor, or the status of the devices battery life. It can also be used to notify if the tracked asset has moved a certain distance greater than, or less than, the specified amount or if the asset is not located on the server. Finally the mobile station API is used to identify and control 802.11 clients located on the network. This includes laptops, phones, tablets etc. all of which are referred to as stations. The API provides a number of methods to monitor and control stations. These include methods to add, edit and delete stations, retrieve station information, station statistics, station location and station history for the specified device. There are also methods available to retrieve the same information on an entire list of stations.

2.2.2.2 Context-Aware Mobility API Benefits

The MSE and Context-Aware Mobility API can provide its consumers with many benefits (Cisco, 2010b) such as flexibility, allowing the user to determine whether to use one or multiple modules depending on their needs and the data can be retrieved using a pull (query) or push (subscription) model. The API can also be used with any programming model. It provides network usability in which the API provides contextual network information without the need for the user to understand how this information is retrieved or how it is represented by the different devices in which the MSE gathers data from. This allows the user to apply more focus on delivering the service which uses the contextual data.

It also provides location usability as the API uses the physical environment to provide location information, removing the underlying location techniques used to calculate the location. This allows users to use the API without the need to understand these location techniques.

Another benefit is accessibility, where the API communicates with the MSE using SOAP/XML it allows web based services such as .Net, C++, Java, etc. to access this information using web technologies such as WSDL. It also provides the benefit of locating multiple devices. The API caters for many devices such as stations, tags, and rogue devices etc. providing contextual data through a common form. This allows the user to become familiar with any

of the API modules once they are familiar with one of them. It also provides contextual combination based on statistical information and network access information. This allows users to build products that use this information without the need of having to retrieve it from multiple sources.

3 Design

This chapter outlines the function requirements and design considerations when implementing the Android Cisco MSE API app.

3.1 Function Requirements

The following specifies the functional requirements of the Android Cisco MSE API app.

- Device Functionality
  - The software artefact will be developed to run on an android device.
  - The android device must include a Wi-Fi radio.
  - The app can only be used in conjunction with LYIT 802.11 WLAN

- Android App Functionality

R1 - The app must display dialog box if Wi-Fi is disabled on the device.

R2 - The app must not display dialog box if Wi-Fi is enabled on the device.

R3 - The app will enable Wi-Fi given the users permission.

R4 - The app will retrieve location information from the Cisco MSE.

R5 - The app displays an error message if no location details are retrieved from MSE.

R6 - The app will display the map of the college.

R7 - The app will position a marker on the map to indicate the user's position.

R8 - The app will update the user's position every 50 seconds.

R9 - The user can zoom in on the map using pinch to zoom.

R10 - The user can zoom out of the map using pinch to zoom.

R11 - The user can zoom in on the map using double tap.

R12 - The user can zoom out of the map using double tap.

R13 - The user can pan the map.

R14 - The map will stop panning when it reaches edge of screen.

R15 - The user can view the full location details retrieved form the Cisco MSE.

R16 - The user can view the log file from the menu.

3.2 System Design

3.2.1 Web Service

It was initially intended to connect to the Cisco MSE and retrieve location details directly from the android device using the Cisco API. However, after some investigation it was clear that this was not feasible due to the API using Remote Method Invocation (RMI) which is not supported by android. Therefore, the decision was taken to deploy the necessary API methods on a web service, which in turn could be called by the android device. JAX_WS was chosen as the programming model as it simplifies application development and has replaced the Remote Procedure Call programming model (JAX-RPC) as the current industry choice (Apache, 2012).

The class diagram in Figure 3.2.1 shows the layout of the web service, including class attributes and methods. The web service consists of a main class which is used to login to the MSE, get the required location information and logout of the MSE. The Location class is used to hold the location information retrieved from the web service methods and is then passed to the android device. All of the web service methods were excluded from the WSDL, apart from two, which only consisted of the actual location details, hence hiding the connection calls to the MSE.

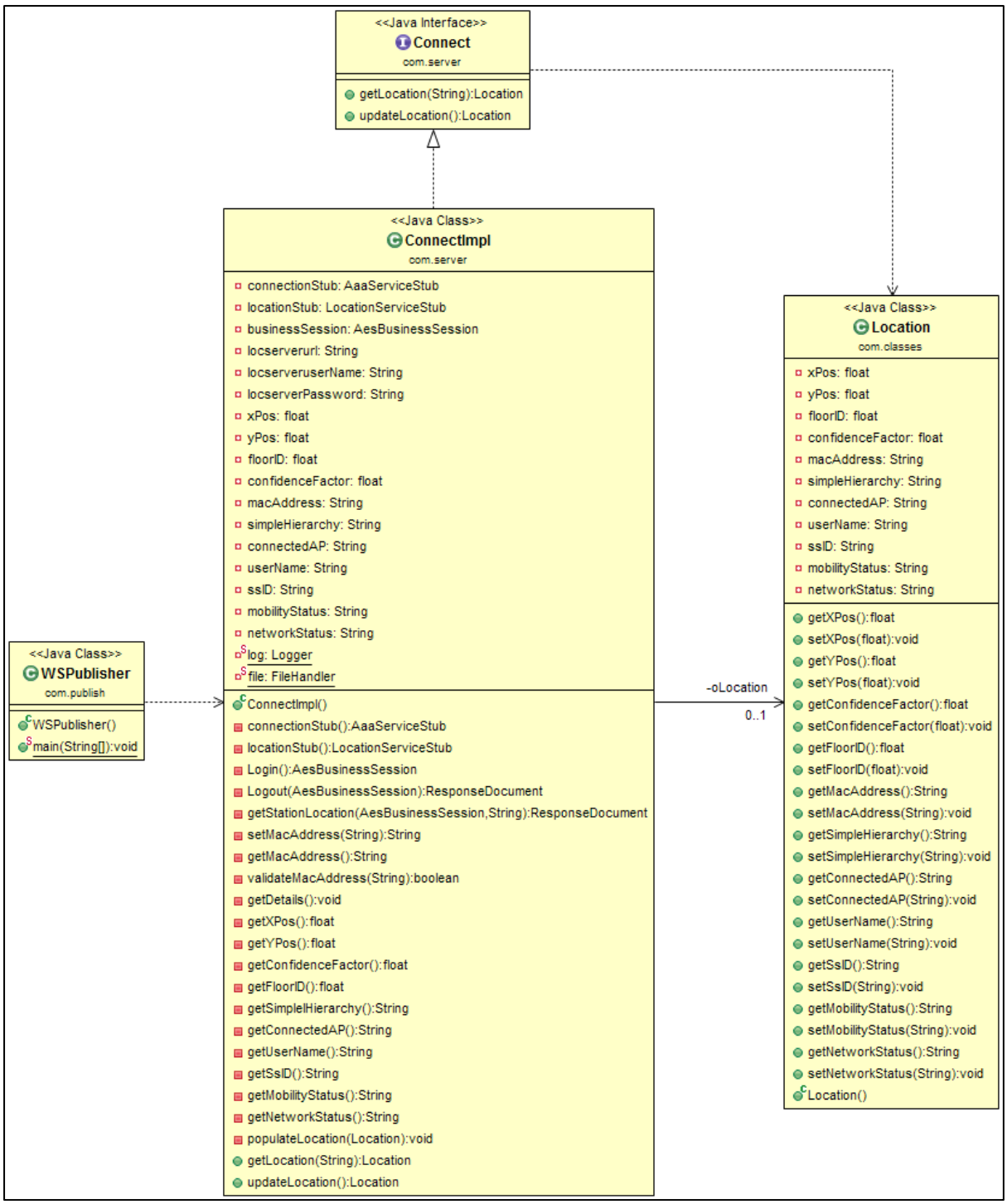

Figure 3.2.1: Web Service Class Diagram

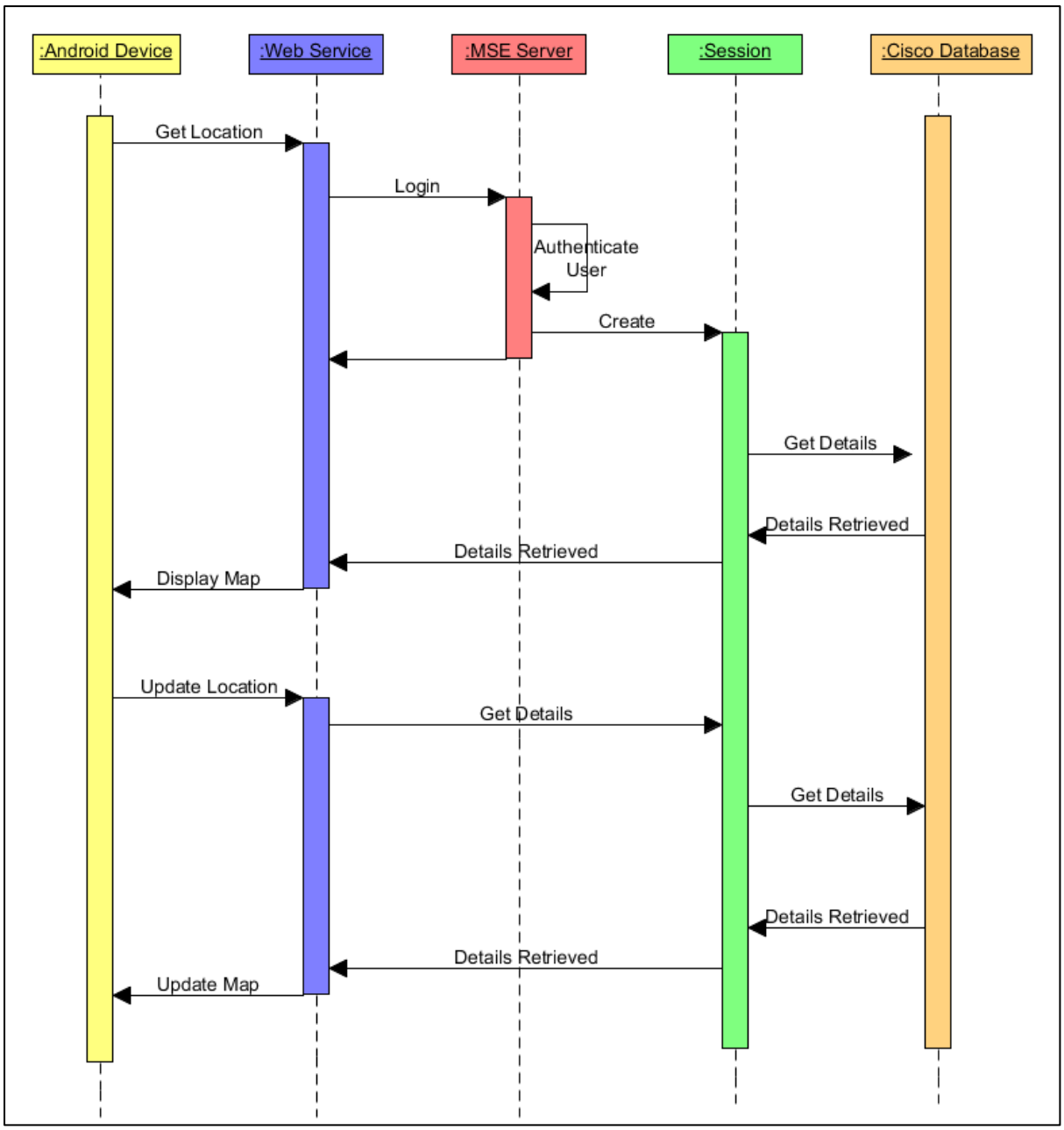

Figure 3.2.2: Sequence Diagram

The sequence diagram in Figure 3.2.2 shows the interaction between the android device, the web service and the Cisco MSE.

1. The android device calls the get Location web service method.
2. The web services call the API login method to login to the server.
3. The MSE server authenticates the user and creates a new session.
4. The session accesses the cisco database, retrieving the user's location information.
5. The location information is passed back to the session, to the web service and finally to the android device.

6. The android device then starts the activity to display the map, which contains the user's position.
7. The update Location method is performed in the same manner as above.

3.2.2 Android Device

The design of the android app is shown in Figure 8.1. This consists of three main sections: the android activities, the map and the web service calls.

The app consists of three activities. The first activity is the SignIn activity which checks if the Wi-Fi is enabled. If the Wi-Fi is disabled then the user is prompted for permission to allow the app to turn on the Wi-Fi. The Wi-Fi must be enabled on the android device in order for the MSE to detect its location. The SignIn activity then calls the web service method to get the devices location. Upon a successful response from the web service, the Map activity is started. The Map activity displays the map of the college, and placing the location of the device on the map. The Map activity is used to update the devices location by calling the web service method. It also contains all the necessary classes which allow the user to interact with the map, such as zooming and panning. The Details activity can be started from the Map activity, allowing the user to view the complete location details of the device. The functionality and flow of the application is shown in Figure 3.2.3.

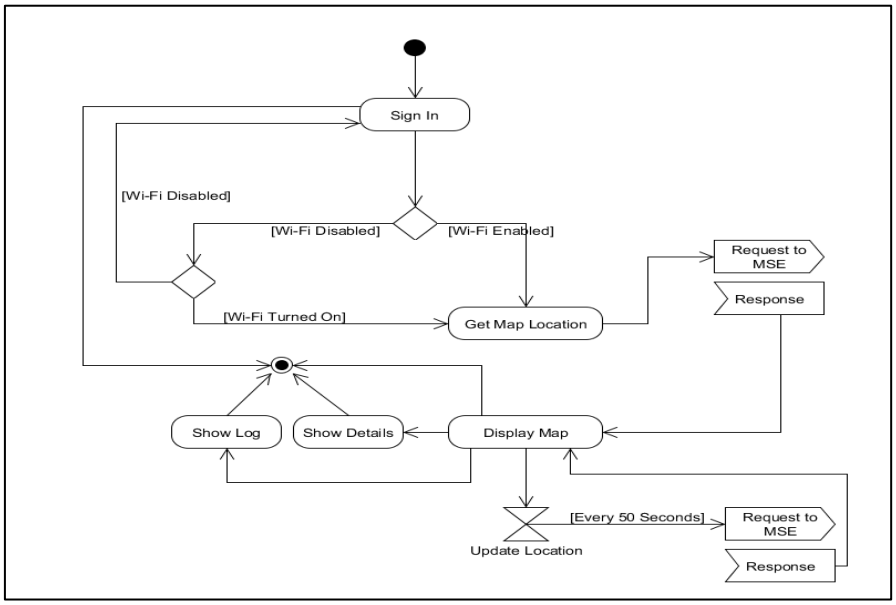

Figure 3.2.3: Android App Activity Diagram

The use case diagram in Figure 3.2.4 shows how the user interacts with the application.

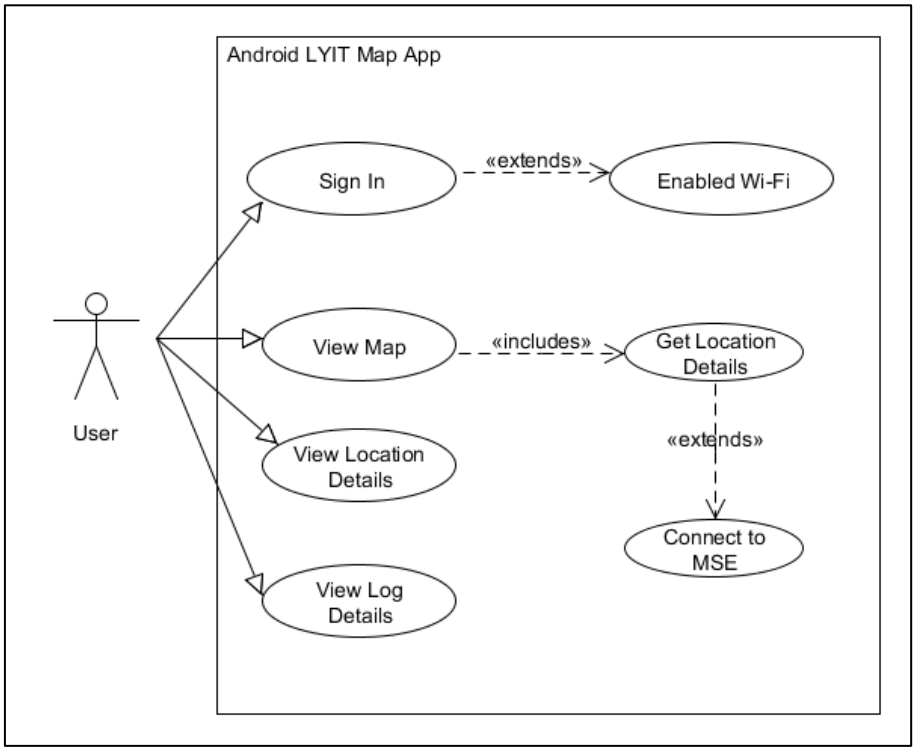

Figure 3.2.4: Android App Use Case

Use Case	Sign In
Objective	User enables Wi-Fi
Precondition	Sign In button must be pressed
Main Flow	1. User press the Sign In button 2. Wi-Fi is disabled, Wi-Fi dialog appears 3. The user selects to turn on Wi-Fi 4. The Wi-Fi is turned on by the WiFiManager
Alternative Flow	1. The user selects not to turn on Wi-Fi
Post Condition	Wi-Fi is enabled

The Map Package contains all the necessary classes that are required to display and interact with the map. This includes a MapLocation class which is used to hold the image of the map and the user's location. The EventHandler class is used to handle the user interaction with the device, such as zooming (double tap and pinch to zoom) and panning. The

AspectQuotient class is used to determine the aspect ratio of the device and the map image which is then used to display the map. The MapControl class is used to control the zoom level and limit the amount of panning that can be performed and the DrawMap class is used to draw the map and marker and limit its boundaries to that of the devices screen.

The Web Service package contains the necessary classes and methods that are required for the android app to communicate with the Web Service. This includes the relevant connection details and the building of the SOAP/XML. It also contains the necessary methods for calling the Web Service along with the parsing of the Web Service response from XML into a Location object.

The Web Service calls are run on a separate thread from the main UI thread. This is enforced by Android and requires the use of AsyncTasks. *"This class allows to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers*." (Google, 2013).
Using the AsyncTask the web service calls can be performed in the background and upon completion the results of the calls can then be published on the UI thread.

3.3 System Security

Security is an essential part of any application especially those that are accessible through the internet such as websites, web services and so forth. Even mobile applications such as Android and Apple apps are being targeted by hackers as highlighted in (Lawson, 2013) where up to 94% of all mobile malware is written for Android apps. However, due to the limited time period available during development, security was not considered as a key aspect of this thesis as the main focus was on developing a location aware Android application using Wi-Fi. Given more time the following security techniques would be applied to the application.

Enhancing the security would involve removing the IP of the MSE server from the actual web service itself. If the app were to be commercialised this IP would have to be added

through the app using a setup/signup activity allowing the user to enter the IP which would then be stored securely in a database. The IP would only be retrieved from the database once the user is authenticated to use the web service.

A user group would also need to be setup on the WCS with the appropriate privileges and using admin/admin as the username/password should be removed from both the web service and the MSE database as this is a vulnerability. The SignIn activity would be altered to prompt the user for a username and password. This username and password would then be authenticated first by the web service and then by the MSE server.

The web service itself would also need to be secured using WS-Security standards which is part of the Web Services Interoperability Technology (WSIT). "*To secure web services, you must consider a broad set of security requirements, including authentication, authorization, privacy, trust, integrity, confidentiality, secure communications channels, delegation, and auditing across a spectrum of application and business topologies.*" (IBM, 2013)

There are numerous ways in which to secure a web service, some of which include:

- XML Encryption & XML Signature
- Username token authentication
- X.509 certificate authentication
- Security Assertion Markup Language (SAML)
- Kerberos token authentication

The methods listed above secure the SOAP message at the message layer as they are included in the SOAP header using XML and therefore they don't need to rely on the communication transport to apply the security.

As the MSE authenticates the user, a username authentication with Symmetric key would be deemed sufficient to access the web service. This would be achieved using the UsernameToken element in the SOAP message. Passwords can be passed as plaintext or more securely as a digest hash using SHA1 hash. The password digest consists of the concatenation of the nonce, the creation time and the password. The nonce is a base64 encoded value. The UsernameToken should also contain a timestamp with a small window

between created and expired time values. The combination of the nonce and timestamp helps prevent replay attacks. The timestamp should also be signed using XML Signature to ensure that no tampering occurred to the timestamp. The password hash is created by the client and passed to the web service. The web service authenticates the user by first looking up the username and retrieving the password. It then recreates the password hash using the plain password and verifies that both password hashes match.

The web service should also be encrypted to ensure that the data passed to and from the web service cannot be read. This would be achieved using Symmetric Key cryptography which uses a single key to sign and encrypt the message. However, in this instance, instead of sending a key, the client sends it's username and password as mentioned above for authentication. The message should be encrypted using an encryption algorithm such as TripleDes.

The Android app should also be secured. In order to do this (Google, 2013) recommend that the code should be obfuscated using the Proguard obfuscation tool in order to make it difficult for attackers to reverse engineer the code. Google also recommends that nonces should not be reused and that a cryptographically secure random number generator should be used to generate them. Finally, the android application (APK file) should be signed with a licence key. The signing of the APK file is used to identify the author of the application. If the application was attacked and altered the attacker would not be able to publish the app without the same licence key. This also ensures that the altered app will not be able to access sensitive information that is available through the original app.

4 Implementation

This chapter details how some of the various elements described in the Design chapter were implemented.

4.1 Web Service

As mentioned in the Design chapter, the reason to implement the MSE API on a Web Service was due to RMI not being supported by android. The API communicates with the MSE using HTTP/SOAP and consists of two wsdls, the mse-aaa.wsdl contains the calls for logging in and logging out and mse-location.wsdl contains the location calls.

The first step in implementing the JAX_WS Web Service was to add the required methods to login and logout of the MSE. The login method consisted of creating a new login instance which involved passing the username and password. The username and password are validated against the users stored in the MSE database. If the login is successful, a business session ID is returned which is used to identify the session and also identify the user's privileges. This business session ID is required by all other methods of the MSE API. The logout method disconnects the user from the MSE, freeing up the business session and resources. Both the login and logout methods contain a AaaServiceStub for the mse-aaa.wsdl. Default username and password were used which granted access to the MSE server. The default username and password should be removed from the MSE database as it is a security exploit. The Web Service also requires an SSL certificate, which was retrieved from the MSE using the Cisco InstallCert Program and stored locally on the machine. The SSL certificate was added to the Web Service properties using the System.setProperty method in which the location of the certificate was passed to.

The next step was to implement the Station Location method. The Station Location method consisted of passing the business session ID, retrieved from calling the login method and the mac address of the android device. The Station Location method makes use of a LocationStub for the mse-location.wsdl. The mac address of the android device was passed up in the web service call getLocation and set using the setMacAddress method. Before the mac address is set, it is validated using regular expression to ensure that it a valid mac

address. The Station Location method returned a response document object which contained all the location details of the android device. This included information such as the x and y coordinate of the mobile device in feet, the floor id, the mac address of the device, the mac address of the associated AP, the SSID, username used to connect to Wi-Fi and a textual representation of the floor hierarchy. A Network status was also included which indicates whether the network is active, inactive or legacy. The confidence factor is also contained within the Station Location response document. The confidence factor is a floating point scalar used to calculate an area which the MSE is 95% confident that the mobile device is located in. A Location object was also implemented to hold the location information retrieved from the MSE. This Location object is initialised in the getLocation method and the location details are assigned to it using the populateLocation method. The Location object is then return by the method. All of the web service methods were excluded from the WSDL apart from two (getLocation and updateLocation) which only consisted of the actual location details, hence hiding the connection calls to the MSE.

4.2 Android Application

4.2.1 Activities

As mentioned in the design section, the Android app consists of three activities, SignIn activity, Map activity and Details activity. An activity is the presentation layer and handles the user interaction.

The first activity created was the SignIn activity. Although the app does not require a login, it does require the Wi-Fi on the android device to be enabled. The reason for this is that the MSE detects a mobile devices location based on RSSI retrieved from the devices Wi-Fi. To ensure that the Wi-Fi is enabled the SignIn activity first checks whether the Wi-Fi is enabled on the android device. This involves creating a WifiManager object which controls the devices network state. The WifiManager check if the Wi-Fi is enabled before allowing the app to proceed any further. If the Wi-Fi is disabled then a notice dialog is displayed indicating that Wi-Fi is required to use the app and provides the option to enable it. If the user selects to enable the Wi-Fi the WifiManager checks if the Wi-Fi is not currently being enabled and enables it. The app waits for fifteen seconds, allowing the device time to enable the Wi-Fi before proceeding.

When the Wi-Fi is enabled, the android devices mac address is retrieved using a WifiInfo object. A WifiInfo object is used to hold the Wi-Fi information which can be retrieved from the WifiManager. A WifiInfo object can contain information such as Mac Address, RSSI, IPAddress and SSID to name but a few. The mac address is used by the MSE to identify which mobile device to retrieve location details for. The mac address is passed to the getLocation web service method which is discussed later in this chapter. Upon a successful response from the web service, a Location object is instantiated within the SignIn activity. The Location object is used to hold all of the location details received from the web service. The Location class implements the Parcelable interface allowing it to be passed from activity to activity. The Location object is stored in a Bundle object and the Bundle object is contained within an Intent. The Intent is then used to start the Map activity.

The Map activity is the most important activity as it is used to display the android device's location on the college map. Using the Location object passed from the SignIn activity, the location details are used by a number of classes which display the map and allow the user to interact with the map, such as zooming and panning. These classes are discussed later in this chapter. The Map activity also contains a web service method to update the devices location. This method is automatically called every fifty seconds. The Map activity contains three options in the options menu, one to update the location manually, one to display the full location details and the other to display the log file.

The Details activity can only be accessed by pressing the Details button contained within the Map activity action bar. The Location object used by the Map activity is passed to the Details activity which simply displays the information held by the Location object using textviews.

The log file contains a record of the location details every time the getLocation or updateLocation web service methods are called. The log file can be accessed by and external application by pressing the View Log button contained within the Map activity action bar.

4.2.2 Map

4.2.2.1 MAPLOCATION

Upon the retrieval of location details from the web service, details are displayed on the android device. This is performed using a combination of classes. The MapLocation class is used to hold all of the details required to display the devices position on the map. This includes the X and Y position, the radius to display the confidence factor of the returned results indicating that the device is located within the displayed circle and the image of the marker. As the X and Y positions are returned in feet by the MSE, they require converting to suit the size of the map. This is done by first converting the positions from feet to inches. This is then divided by the scale of the map before being multiplied by the pixel DPI (Dots per Inch) of the image. The X and Y position are then given an offset so that they are placed in the correct position on the map. The pixel values are then converted to DIP (Density-Independent Pixel) so that regardless of the screen size of the device the location will be shown in the correct position on the map.

4.2.2.2 ASPECTQUOTIENT

The AspectQuotient class contains the quotient between the aspect ratio of the devices screen and the aspect ratio of the map. This is required because the map does not have the same ratio as the screen and is therefore required when zoom levels are applied. The reason for this is that the screen may have a different aspect ratio to that of the image and thus when a zoom value is applied only part of the image will fit the dimension of the screen.

4.2.2.3 ZOOMSTATE

The ZoomState class contains variables for the zoom level and the amount of panning both in the x and y axis. It also contains the methods to calculate the zoom level for both X and Y. In the getZoomX method the aspectQuotient value is passed to the method and the minimum value between the zoomLevel and the zoomLevel multiplied by the aspectQuotient is returned. In the getZoomY method the minimum value between the zoomLevel and the zoomLevel divided by the aspectQuotient is returned as shown in Code Listing 4.1 and Code Listing 4.2. The reason for this is, when a zoom level has a value of one,

only one dimension of the map will fit the view. This means that the other dimension will only partially cover the view which results in a zoom level lower than one for that particular dimension. By multiplying and dividing the zoom level by the aspect ratio, the appropriate zoom levels can be applied to x and y axis.

```
public float getZoomX(float aspectQuotient) {
        return Math.min(mZoomLevel, mZoomLevel * aspectQuotient);
}
```

Code Listing 4.1: ZoomState - getZoomX

```
public float getZoomY(float aspectQuotient) {
        return Math.min(mZoomLevel, mZoomLevel / aspectQuotient);
}
```

Code Listing 4.2: ZoomState - getZoomY

4.2.2.4 MAPCONTROL

In order to determine the levels of zooming and panning permitted, the MapControl class was created. The MapControl class contains instances of both AspectQuotient and ZoomState classes. The levels of zooming permitted are controlled using the ZoomState object which in turn uses the AspectQuotient object. Once the zoom level values are retrieved, they are checked to ensure that they are within the zoom boundaries defined within the class.

When the MapActivity is created the ZoomLevel is set to 1 and the PanX and PanY values are set to 0.5. These values place the un-zoomed image of the map directly in the centre of the screen. When performing the zoom operation the image must also be panned accordingly. This was achieved by retrieving the current ZoomX and ZoomY values before updating the ZoomState object and retrieving the new ZoomX and ZoomY values. The panning values for both x and y axes are then set using the old and new zoom values which keep the down coordinates invariant when zooming, allowing to zoom on a particular section of the map.

Movement of the image requires the use of the ZoomState object which is used to set the amount of panning in both x and y axis. This was achieved by retrieving the current pan value, adding the amount that the user wishes to move in both x and y axis and dividing these values by the ZoomX and ZoomY values. By performing this calculation, it allows the pan to follow the fingers and it also takes the zoom level into consideration.

To prevent the user from panning the image of the screen, the panX and PanY values are set accordingly so that they do not exceed their min and max values as shown in Code Listing 4.3. This is carried out by retrieving the ZoomX and ZoomY values using the aspectQuotient. These values are then passed to the getMaxPanDelta method which returns the maximum value that the user can pan from the centre of the screen based on the zoom level.

```
private float getMaxPanDelta(float zoom) {
        return Math.max(0f, .5f * ((zoom - 1) / zoom));
}
```

Code Listing 4.3: MapControl - getMaxPanDelta

The min and max values are then set for x and y by subtracting the MaxPanDelta value from 0.5 for the min value and adding the MaxPanDelta value to 0.5 for the max value as shown in Code Listing 4.4: MapControl – setting the values to limit the panning .

```
final float panMinX = .5f - getMaxPanDelta(zoomX);
final float panMaxX = .5f + getMaxPanDelta(zoomX);
final float panMinY = .5f - getMaxPanDelta(zoomY);
final float panMaxY = .5f + getMaxPanDelta(zoomY);
```

Code Listing 4.4: MapControl – setting the values to limit the panning

4.2.2.5 EVENTHANDLER

Allowing the user to interact with the map, events such as zooming and panning is handled by the EventHandler class. The EventHandler class contains an instance of the MapControl class which is used to limit and control the zooming and panning. The class is used to determine panning, double tap to zoom and pinch to zoom.

Pointer refers to the touching of the screen with one's finger and touch gestures refers to the movement made by this pointer. Android provides APIs to create and detect different types of gestures. To determine how much the user wishes to pan and zoom the image requires the onTouchEvent callback to be overridden. The onTouchEvent makes use of androids MotionEvent which stores pointer information in an array. Each pointer has an ID which is used to find the index of the pointer in the array.

The ACTION_DOWN touch event is called when the first pointer touches the screen which starts the gesture. Here, the x and y coordinates of the pointer along with the pointer ID are recorded. The ACTION_MOVE touch event is called when a change has occurred during a press gesture. The pointer index is retrieved using the pointer ID and with the pointer index, the x and y coordinate of the pointers current position is recorded. The x and y coordinates from the ACTION_DOWN touch event are subtracted from the new coordinates and divided by the map width and height for each respective axis. These new values indicate how much the pointer has moved and are passed to the MapControl object to pan the map.
The ACTION_UP touch event is called when the last pointer leaves the screen and here the pointer ID is set to a default value and the same is done for the ACTION_CANCEL touch event. The ACTION_POINTER_UP touch event is called when a non-primary pointer leaves the screen, but at least one other pointer is touching the screen. The pointer ID is retrieved and checks to see that the active pointer ID is not referring to the pointer that just left the screen. If it is the same then a different pointer is selected as the active pointer and its coordinates are save so that it can be used by the ACTION_MOVE event.

To detect the motion for zooming, when using both double tap zoom and pinch to zoom, GesturesDetectors are used.
"*GestureDetectors are small filter objects that consume MotionEvents and dispatch higher level gesture events to listeners specified during their construction*" (Powell, 2009). To perform the double tap zoom, a GestureDeture object is created in which a GestureListener object is passed to its constructor. The GestureListener is a class that extends Gesture.SimpleOnGestureListener and overrides the onDoubleTap method. If the onDoubleTap method is called, the zoom level is set to a max or min value, depending on whether the image is currently being zoomed in on or not. Then using the MotionEvent

object the x and y coordinates of the pointer are returned and along with the zoom level, are passed to the MapControls controlZoom method. To perform the pinch to zoom, a new ScaleGestureDetector is created which takes in a ScaleListener in its constructor. The ScaleListener class extends ScaleGestureDetector.SimpleOnScaleGestureListener and overrides the onScale method. Within the onScale method the current zoom level of the map is retrieved from the MapControl class and is multiplied by the scale factor from the scaleGestureDetector. The scale focus points are then retrieved using the scaleGestureDetector getFocusX and getFocusY methods, which indicate which part of the map the user wished to pinch to zoom to. These values are then passed to the MapControls controlZoom method.

To allow these gestures to detect a gesture event the MotionEvent object is passed to the gestures onTouchEvent methods. To determine when the user is attempting to scale or pan the map, the ScaleGestureDetectors isInProgress method is used. This method returns a Boolean value indicating whether a scale gesture is in progress or not. Only when this value returns false is the map allowed to be panned.

4.2.2.6 DRAWMAP

Previously discussed in this section is the handling of events, how the map is zoomed, scaled and panned and finally all that remains is how the map and marker are drawn on screen by the DawMap class. The DrawMap class contains an AspectQuotient object, a ZoomState object and a MapLocation object. It is in this class that the actual map image is set and the aspectQuotient value is based on this image. This class is also used to set the map marker on the map with the use of the MapLocation class. To display the map image, two Rectangle objects were created, the source rectangle for the part of the map to be drawn and the destination rectangle for the area of the screen to be drawn to. The four corners of the destination rectangle are set as the screen's four corners. The four corners of the source rectangle are set using a combination of the screen's width and height, the image width and height, the amount of panning in the x and y axis and the amount of zooming in the x and y axis. The source rectangle is then adjusted so that it fits within the image and the destination rectangle is also adjusted so that it displays the full/zoomed image. The image of

the map was then drawn by passing the image of the map, the source and destination rectangles and a paint object to the Canvas.DrawBitmap method.

Drawing the marker simply involved passing the marker image along with the x and y coordinates to the Canvas.DrawBitmap. This resulted in the marker being drawn on top of the map as intended. However, when the map was zoomed or panned the marker would move along with the screen as the transformations were not being applied to it. To rectify this issue the map and marker image were combined as one. This was accomplished by creating a new bitmap called mMapMarker using the Bitmap.createBitmap method which took in the map width, the map height and the map configuration which describes how the pixels are stored. A new Canvas object was then created in which the new bitmap (mMapMarker) was passed to its constructor. Then, using this canvas the image of the map was drawn at position 0, 0. Again, using the same canvas object, the map marker was drawn at its correct position. Once this was achieved, a circle was drawn around the map marker indicating the confidence factor of the location; this was again done using the same canvas object. As the same canvas object was used to draw all the images, it meant that they were all layered in the correct order and stored in mMapMarker bitmap. It is this mMapMarker image that is drawn to the android screen using the Rectangle objects.

4.2.3 Web Service Calls

As Android does not contain native support to consume a web service, the KSoap2 library was used. KSoap2 is a lightweight library that allows Android applications to consume SOAP-based web services. Using KSoap2, a SOAP envelope is created, followed by a SOAP object (request) in which the web service namespace and method name are passed.

In the getLocation web service method the mac address of the android device is added to the request object and the request object is then added to the envelope. A HttpTransportSE object is then created which uses the url of the web service. The HttpTransportSE object calls the web service method on the server using the namespace and SOAP envelope. Another Soap object (response) is returned by the web service. The details contained within the response object are then parsed and stored as a Location object. To ensure that the response object contains the necessary location information, the kSoap2 hasProperty

method was used. This method was called before each property of the location object was set.

The Web Service calls are run on a separate thread from the main UI thread. This is enforced by Android and requires the use of AsyncTasks. *"This class allows to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers*." (Google, 2013).
Using the AsyncTask the webservice calls can be performed in the background and upon completion the results of the calls can then be published on the UI thread.

The AsyncTask consists of four steps when executed:

- onPreExecute(): This is invoked before the task is executed and is used to setup the task whilst displaying a progress bar in the UI.
- doInBackground(Params…): This is invoked on the background thread after onPreExecute is finished executing and is used to perform the background computation such as a web service calls and so forth.
- onProgressUpdate(Progress…): This is invoked after the call publishProgress(Progress…) which can be used by doInBackground. It is usually used to display a progress bar whilst the background computation is executing.
- onPostExecute(Result): This is invoked on the UI thread once the background computation finishes executing. The result of the background computation is passed as a parameter.

For this application the onPreExecute method is used to display whether the location details are being retrieved for the first time or whether they are being updated. The doInBackground method calls the web service methods, retrieves the location details and parses the returned details into a Location object. Finally, the onPostExecute method retrieves the results of the background computation and updates the classes that handle the positioning and display of the map. This is carried out for both the getLocation method and updateLocation method.

During development some issues did occur, such as, when the screen was rotated in the MapActivity the location values were set to the values that were passed from the SignIn activity. This was occurring because when the screen was rotated, the MapActivity would

reset to the values when it was first created as its onCreate method was being called. This issue was resolved by saving the location object using the onSavedInstanceState method. When the onCreate method is called the savedInstanceState is checked to see if it contains a location object. If it does not contain a location object then it gets the location object passed from the SignInActivity. Another issue was that the location on the map differed depending on the screen size of the android device. This was due to physical pixels being used to place the location on the map. This was resolved by converting the pixels into density-independent pixels (DIP) which ensures that the location on the map remains the same regardless of the size of the screen size.

5 Testing and Results

This chapter discusses the chosen environment and the results of the tests that were performed.

5.1 Testing Environment

As the Android app must connect to a Cisco MSE, the chosen environment had to be the LYIT campus. The campus consists of three buildings, however only the main campus building was used for testing purposes. This building was chosen as it accounts for the most activity within the campus. This allowed the app to be tested at both busy and quiet periods in order to determine if network activity and interference affected the MSE results. The app was tested on an ASUS Nexus 7 android tablet running android version 4.2.2 Jelly Bean.

5.2 Testing

Testing was carried out in the main building using all three floors. Ten locations were chosen as shown in Figures 8.2, 8.3, 8.4 and the system was tested ten times resulting in one hundred sample points. The tests were carried out at different times with various numbers of other clients connected to the college network.

Testing was focused around three key elements. This included Level Differentiation Testing which determined how often the MSE detected the mobile device on the correct floor. The results are discussed in section 5.3.1, where an accuracy of 100% relates to the device being detected on the correct floor for each test and an accuracy of 0% relates to the device never being detected on the correct floor. The AP Differentiation Testing determined the consistency of the APs by noting the mac address of the detecting controller. The results are discussed in section 5.3.2, where a consistency of 100% relates to the device being detected by the same AP for each test and an accuracy of 0% relates to the device never being detected by the same AP. The Distance Accuracy Testing determines how accurate the MSE is in terms of location. The results are discussed in section 5.3.3, where an accuracy of x amount in feet relates to the distance of the detected location from the actual location. The Distance Accuracy Testing also determined the accuracy of the MSE confidence factor which

is an area that the MSE is 95% confident that the device is located in. The confidence factor values listed in section 5.3.3 relates to the radius of the detected area.

5.3 Results

5.3.1 Level Differentiation Testing

The MSE determines which floor the mobile device is detected on and returns the floor ID for the corresponding floor. This floor ID is returned by the web service and is used to determine the accuracy of the MSE detecting the correct floor within a multi-storey building. Of the one hundred sample points recorded, only forty of these had 100% accuracy of being detected on the correct floor. Of the remaining sixty sample points, twenty had an accuracy of 90% and twenty had an accuracy of 80%. Of these sample points, location five was the least accurate, resulting in only 10% accuracy of being detected on the correct floor, taken from ten sample points.

5.3.2 AP Differentiation Testing

Cisco's MSE uses the Fingerprint method to determine the mobile devices location. As discussed previously, the Fingerprint method revolves around the measuring of RSS from the mobile device. This signal strength is recorded by the APs which detect the mobile device, therefore, it was deemed applicable to record the consistency of these APs. Contained within the Cisco API is a method to return the Mac Address of the AP that detected the mobile device. This Mac address was noted for all one hundred sampling points. The results revealed that only one location (location ten) had 100% consistency. Five locations had a consistency of 80% with another three locations having a consistency of 60%. Of the ten locations, location five again performed the poorest with a consistency of 40%.

5.3.3 Distance Accuracy Testing

The x and y coordinates along with the confidence factor are recorded in feet by the MSE. Therefore, for simplicity, the testing of locations and results were also recorded in feet.

Note that due to the small scale of the map used within the android app all sample points are measured with an accuracy of plus or minus 10ft.

The results revealed that location one performed the best with and average accuracy of 34ft and an average confidence factor of 93ft. Locations four, seven, eight and ten also performed reasonably well with an average accuracy of 48ft, 53ft, 61ft and 65ft respectively. However, locations four and seven's confidence factor was over twice as high as that of location one with an average accuracy of 189ft and 203ft respectively, with location eight having an average confidence factor accuracy of 123ft. Of all the tests, location ten had the lowest average confidence factor with an accuracy of 53ft, however, on 80% of occasions, the mobile device was not detected within the confidence factor range.

Locations two and three had near identical results with an average accuracy of 65ft and 68ft respectively, with an average confidence factor accuracy of 221ft and 192ft respectively. Locations six and nine were amongst the poorest performers with an average accuracy of 116ft and 150ft respectively. These two locations also recorded the highest confidence factor values with an average accuracy of 275ft and 217ft respectively. It was expected that given the results of the level differentiation and AP differentiation tests that location five would also record the poorest accuracy, but this was not the case. Although still relatively poor, location five had an average accuracy of 114ft and an average confidence factor of 189ft.

The number of wirelessly connected clients on the college network was also recorded to see if the levels of network activity had any bearing on the performance of the locations. The results indicate that the level of activity within the college did not seem to have any bearing on the accuracy, as the results recorded during busy periods outperformed that of quiet periods in some instances and vice versa.

(Cisco, 2012a) state that an accuracy of less than ten meters should be attainable 90% of the time and an accuracy of less than five meters should be attainable 50% of the time. This differs vastly with the results retrieved from the MSE in the LYIT where the results reveal an estimated average accuracy of 77ft (7 – 45meters) and an estimated average confidence

factor of 175ft (13 – 80 meters) for all one hundred samples. The results also differs vastly to that of other techniques mention in Chapter 2, as it lacks in comparison with ToA and AoA techniques which claim to be capable of accuracies of 1 - 2 meters. It also lacks in comparison to RSSI, TDoA and Location Patterning techniques which claim to be capable of accuracies of 2 - 5 meters. However it must be noted that these other techniques would require testing in the LYIT or a similar environment in order to properly compare them to Cisco's Context-Aware mobility.

Figure 5.3.1 shows the average accuracy for the ten sample locations.

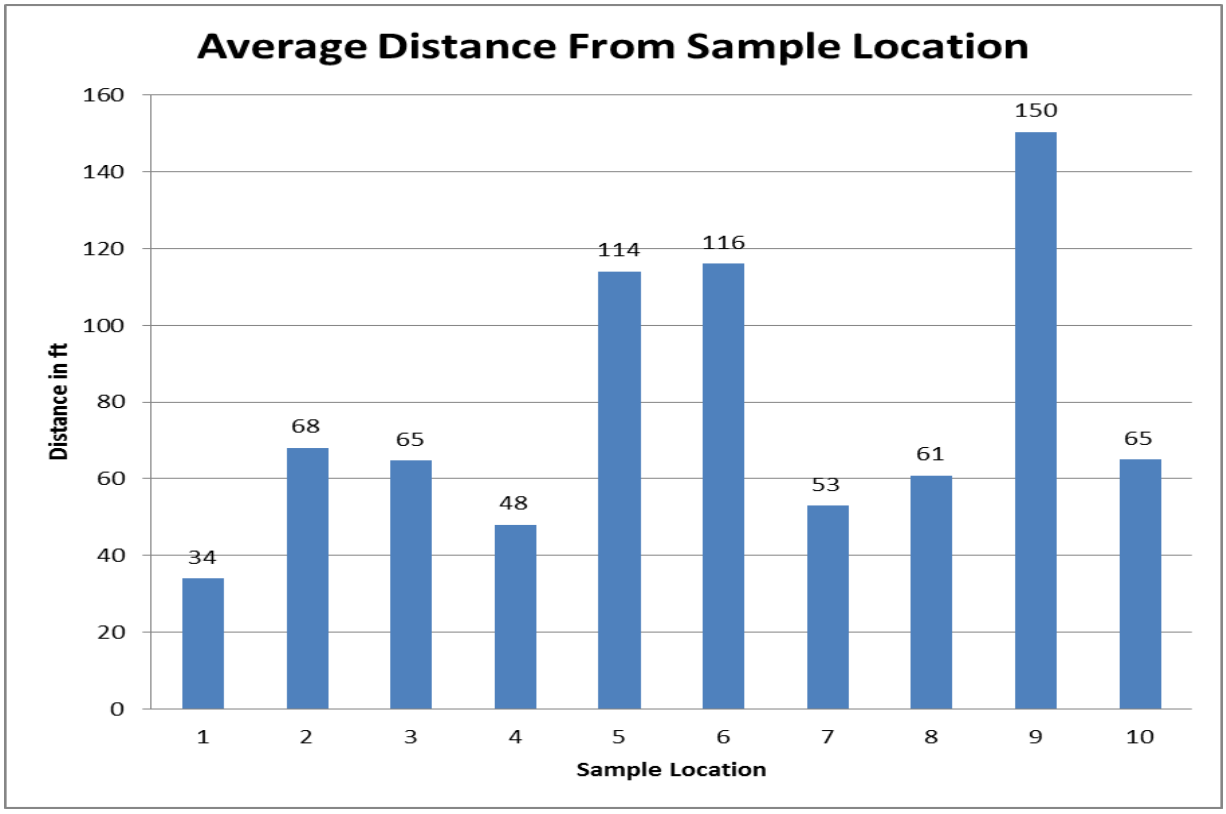

Figure 5.3.1: Average Distance Accuracy

Figure 5.3.2 shows the average confidence factor for the ten sample locations.

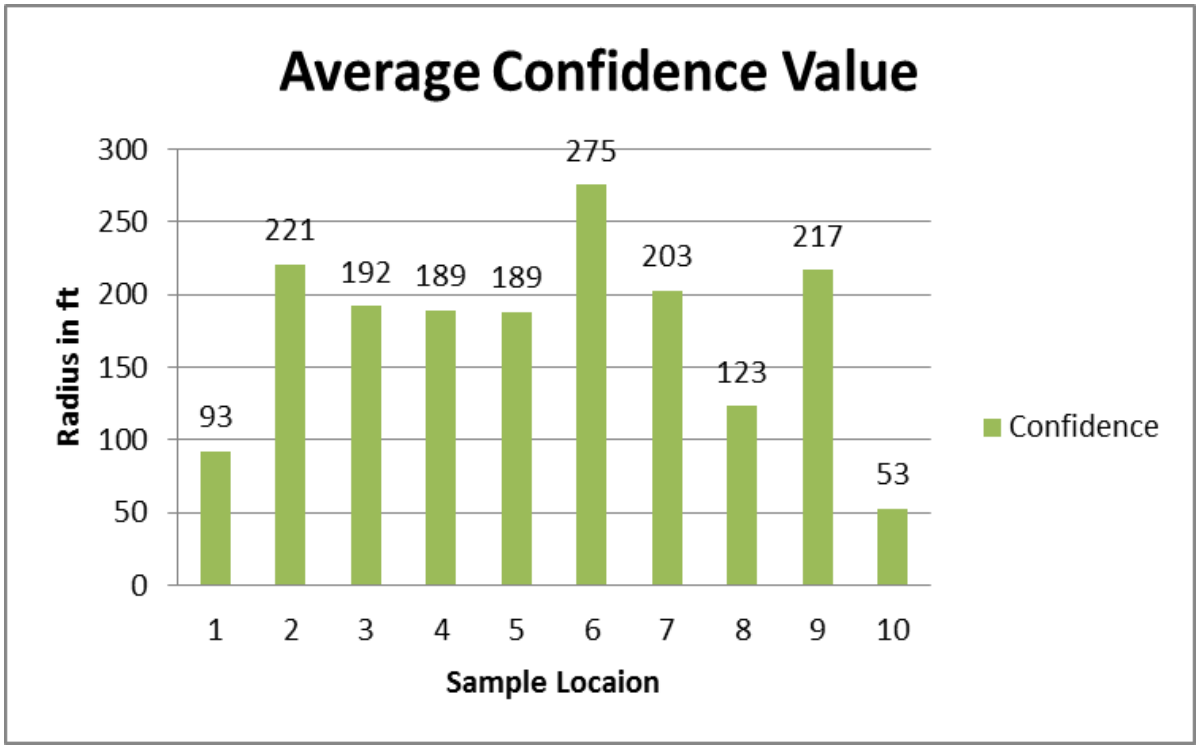

Figure 5.3.2: Average Confidence Factor

5.4 Interpreting the Results

To correctly interpret the results of the tests mentioned above, a number of APs and RSSI values would need to be collected for all of the one hundred sample points. The Cisco Context-Aware API provides this capability by providing a list of neighbouring AP data and a list or RSSI values. However when this functionality was implemented in the Web Service, both the neighbouring AP data list and the RSSI list returned empty. Due to time constraints, it was not possible to investigate this further and therefore the interpretations are mainly based on assumptions. Location one performed the best and it is assumed that this is due to it being in close proximity to four APs with two of them located in the same room as shown in Figure 8.1. This was also verified through the WCS by checking that the associated AP was visible by the other three APs. This indicates why location one had good results, as stated in Chapter 2, the higher number of detecting APs, the better the accuracy.

Location six and nine had the least accurate results of all ten locations. When these were investigated, it was clear to see in Figure 8.2 and Figure 8.4 that both locations are in close proximity to only one AP, therefore, it is assumed that the poor results are due to them being detected by mainly one AP. This was also verified through the WCS where both the closest APs were checked. The WCS confirmed that both APs for location six and nine were not visible to their surrounding APs.

Location ten was also investigated as it had the smallest confidence factor range, however for the majority (80%) of tests, the device was not located within this range. It was clear from the results and from viewing Figure 8.4 that although there was an AP directly positioned at location ten, the majority of results were detecting the device close to a different AP which was located just a number of meters away. It was assumed that due to these results that there could have been a calibration error. The mac address of the two APs involved were first physically checked and then checked on the WCS where it was clear that the two APs were mixed up and therefore configured incorrectly for the MSE. This confirmed why the results were closer to the AP that was further away and why the confidence factor was so small. If these APs weren't mixed up and configured correctly, then the device would be located within the confidence factor range and the distance from the sample point would be the smallest, hence, location ten would have the best accuracy.
It was also noted during testing that the response time of location requests made to the MSE was slow, with each response taking around thirty seconds. The greater the number of devices being used within the college, the greater the processing load on the MSE. This affects the latency of the network, where in the case of the MSE, it refers to the delay between when the RSSI information of a mobile device is received by the MSE and when the location of this mobile device is calculated by the MSE. Latency can also be related to the NMSP (Network Mobile Service Protocol) aggregation window. This however can be tuned, allowing the user to set the time between location calculations and updates.

It is assumed that other factors have also impacted on the results of the ten locations. These could vary between the surrounding structure (steel, concrete, plasterboard), multipath interference, positioning of APs and also the configuration and calibration of the MSE, all of which can have an impact on the accuracy of the system.

Factors such as impedance from fire doors, concrete walls and windows can have an impact on accuracy. Signal attenuation varies depending on the material that the signal passes through. Figure 5.4.1 provides estimated information on signal loss for various objects, as different countries use different materials in the construction of objects and buildings.

Object in Signal Path	Signal Attenuation Through Object
Plasterboard wall	3 dB
Glass wall with metal frame	6 dB
Cinder block wall	4 dB
Office window	3 dB
Metal door	6 dB
Metal door in brick wall	12 dB
Human body	3 dB

Figure 5.4.1: Signal Attenuation through Objects (Cisco, 2012a)

An example of this would be if a signal of 15dBm enters through an office window, its signal strength will be reduced to 12dBm when it exits the window. The LYITs main building consists of concrete and brick walls and also contains a large amount of metal content. A high amount of metal content results in reflected signal creating multipath distortion.

Calibration and positioning of APs can also have an effect on accuracy. To achieve optimal accuracy the MSE must be configured correctly. (Cisco, 2012a) state that a value of -75dBm is used as the minimum signal level which must be recorded from a minimum of three AP's located on the same floor. It also states that AP's should be staggered for areas such as long narrow corridors and that there is sufficient AP density and perimeter coverage. To help identify correct locations for AP placement and density, the WCS Planning Tool can be used.

5.5 Software Testing

Test	Description	Req Met	Pass/Fail
1	Display dialog box if Wi-Fi is disabled	R1	Pass
2	Don't display dialog box if Wi-Fi is enabled	R2	Pass
3	User selects enable Wi-Fi in dialog box and Wi-Fi is enabled	R3	Pass
4	App calls web service which retrieves location details form MSE	R4	Pass
5	App displays error message when connection fails or no details are retrieved	R5	Pass
6	On successful retrieval of location information, map is displayed	R6	Pass
7	Map marker is displayed on map indicating users location	R7	Pass
8	Web service is called every 50 seconds to update user position	R8	Pass
9	Users zooms in on map by pinching the screen	R9	Pass
10	Users zooms out of map by pinching the screen	R10	Pass
11	Users zooms in on map by double taping the screen	R11	Pass
12	Users zooms out of map by double taping the screen	R12	Pass
13	Users pans the map by dragging finger across screen	R13	Pass
14	Map image will not be panned beyond the screen boundary	R14	Pass
15	Details activity is displayed when selected from menu	R15	Pass
16	Log file is selected to be displayed by different app when selected from menu	R16	Pass

6 Conclusion

The objective of this thesis was to investigate if the Cisco Context-Aware Mobility provided a viable solution to the ever growing needs of enterprises, universities and shopping centres to name but a few, by providing a high accuracy RTLS using Wi-Fi.

GPS's poor performance indoors has placed greater emphasis on indoor positioning based on WLAN 802.11. Indoor positioning based on WLAN can provide the benefits of being cost effective and easy to implement as it makes use of an already existing infrastructure. This thesis includes a Survey chapter which discussed the different Wi-Fi measuring techniques that can be used to determine ones location. The different techniques included Cell of Origin (COO), Time of Arrival (ToA), Time Difference of Arrival (TDoA), Received Signal Strength Indication (RSSI), Angle of Arrival (AoA) and Location Patterning, all of which have their advantages and disadvantages.

COO is one of the simplest methods of location positioning, however, it lacks accuracy as it makes no attempt to determine the exact position of the user. Instead, it returns the cell that the user was detected in and can therefore only provide accuracies of up to one hundred meters and over. It is relatively simple to implement and does not require complex algorithms and therefore, it can provide fast positioning performance.

ToA is based on the time taken for a signal sent from a mobile device to two or more receiving sensors. This technique requires accurate timing and so it requires dedicated hardware (synchronized clocks). ToA systems are flexible and highly adaptive to changing environments. They can provide accuracies of up to 1 to 2 meters provided the investment is made in purchasing synchronized clocks. A downside to ToA is that any errors in the timing can result in large positioning errors. A timing error of 1 microsecond can lead to a positioning error of up to 300m.

TDoA, like ToA, is based on the arrival time of a signal sent from a mobile device to three or more receiving sensors. Unlike ToA, TDoA only requires the receiving sensors to have synchronized time sources, which means that it does not require knowledge of the

transmission start time. TDoA systems are best suited to large, open buildings with little obstruction. TDoA systems mainly suffer from Non Line of Sight (NLOS) which increases the travelling distance of the signal due to reflection and diffraction which decreases positioning accuracy.

RSSI make use of a mobile device or a receiving sensor to measure the signal strength of a transmission. The locations of the receiving sensors are known and by passing the received signal strength to a calculation engine, the location of the mobile device is retrieved. Like TDoA, multipath interference impacts on the accuracy of the system which means the signal has to travel further, which results in lower signal strength. RSSI is a cost effective solution which does not require any specialised hardware and is best suited to environments where a direct line of sight is attainable.

AoA calculates the angle of incidence of signals received by at least two receiving sensors with the use of an antenna array. The use of three or more receiving sensors increases positioning accuracy. The location of the mobile device is determined by the intersecting lines. AoA techniques have previously been used by cellular companies to determine a mobile devices location in cases of emergency. AoA struggles when a clear line of sight is not attainable and like ToA, a small error in angle measurement can result in a large positioning error.

Location Patterning is a technique which is most commonly implemented using RSSI, but can also be implemented using ToA, TDoA and AoA techniques. Location Patterning is carried out in two phases, calibration and operation phase. The calibration phase consists of walking around the chosen environment with a mobile device and recording the signal strength received at receiving sensors (APs) at marked locations. At each marked location, an array of RSS values (one for each AP detected) are recorded in a database along with the coordinates of the location. This is known as a radio map. The operation phase consists of the APs forwarding the RSS from the mobile device to a calculation engine. The calculation engine performs complex algorithms and compares the data to that of the radio map and determines the mobile devices location. The Location Patterning technique does not require any specialised hardware and is fully implemented in software. It is thought to provide

higher accuracy than COO, ToA, TDoA and AoA techniques. However a high number of APs are required to provide high accuracy and the calibration phase can be labour intensive. The radio maps provide little re-use with changing environments requiring re-calibration.

The survey also looked at the Cisco Context-Aware Mobility and looked at how it could be implemented using the Context-Aware API. The Cisco Context-Aware Mobility makes use of the Cisco RF Fingerprinting technique to measure Wi-Fi signals. It uses RSSI and works in the same way as Location Patterning, however, Cisco state that it is more efficient, quicker and provides better accuracy than Location Patterning. This is achieved by building a RF propagation model using data collected from calibration phase, which calculates path loss and shadow fading deviation to account for propagation discrepancies. Cisco RF Fingerprinting does not require the same calibration effort as Location Patterning nor does it require re-calibrating as often as Location Patterning. It can also make use of pre-package RF models for similar environments.

Cisco's Context-Aware Mobility Service is used to capture information on mobile assets including devices, people or products. The information captured can include location, temperature, availability and applications used. The Context Aware Mobility runs on the MSE. The MSE communicate with WLAN Controllers (WLC) and the WCS, which is a WLAN management tool that provides the ability to manage network controllers through a web based user interface. The MSE provides real time location tracking and historical data of Wi-Fi clients, RFID tags, and rogue device. Applications can communicate with the MSE through APIs such as the Context-Aware API using XML/SOAP.

With the ever increasing popularity of mobile devices, an Android application was developed to determine if the Context-Aware Mobility was capable of providing a high accuracy RTLS, that could potentially be used by a large number of students/staff. The app was designed to be used within the main building of the LYIT campus, to allow for simple interaction with the college map and that could be easily deployed onto a mobile device.

The implementation of an Android application made use of the Context-Aware API through the means of a web service. Using the API, the Android application was able to retrieve

location details from the LYIT MSE and use them to display the mobile devices position within the college on a map. The Context-Aware API provides a vast array of functionality, many which have not been implemented in the developed software artefact. These include the ability to retrieve location history for a single mobile device, location history for multiple mobile devices, network design information and notifications based on location to name but a few.

Testing was carried out in the main building with one hundred sample points collected, covering ten different locations which were taken from three different floors. Three key elements were noted during testing: if the device was detected on the correct level, how often was it detected by the same AP and finally what was its distance from the actual sample point. The response time of the MSE was deemed to be slow as each response took at least ten seconds or more. This could be a critical issue in a live system as it would have a large impact on the accuracy of the system. In order to retrieve an accurate representation of the Cisco system, sufficient time was given at each sample point, allowing the application to retrieve a response from the MSE which was relative to the mobile devices location. This resulted in the tests not taking into account the active movement of the mobile device when the location request was made to the MSE and this is an area that would need to be investigated further.

The results indicated that of the one hundred sample points, only forty contained a 100% success rate of being detected on the correct floor. Only one location had a 100% success rate of being detected by the same AP, with a further five locations having a success rate of 80%. The distance accuracy of the tests returned an average of 7m – 45m with 2.7m the best recorded distance of all one hundred sample points. The level of network activity within the college surprisingly had no bearing on the accuracy of tests as the results were just as good and better in some instance when there was a high level of activity.

It was concluded from the results that the density of APs in the college is sufficient for 100% data coverage, but not sufficient enough to enable high positioning accuracy. To obtain optimal accuracy, (Cisco, 2012a) state that the device must be detected by four or more APs. It also states that an AP must be placed every 12 – 20 linear meters or one AP every

230 – 450 square meters. In order to fully test the quality of the Cisco Context-Aware Mobility, a full quality assurance and validation process would be required. This would involve ensuring that all APs correctly match up to the MSE, that all heat maps are correct and a walk test of the campus to ensure that every location in the building is visible by multiple APs. It would also require the MSE to be re-calibrated to account for changes to the building which can affect RF propagation. To ensure that these steps are carried out to a sufficient standard the WCS Planning Tool should be used to identify correct locations for AP placement and also the WCS accuracy should be used to generate accuracy reports for the specified reference points used during testing.

Due to the strict time constraints a number of key aspects could not be implemented, a significant element being security which is discussed in Chapter 3. Given more time, features like app configuration and registration would allow for it to be easily commercialised. This would involve both server and client side changes, but it would allow for the app to be installed without any in-house installation or configuration. Location history would also be an additional feature that would be added. Location history details can be retrieved through the Cisco API. Other features would include downloading the maps from the MSE, or allowing the user to load their own maps. The college maps were not used directly from the MSE in the final implementation due to their poor quality, and therefore, new maps were used which were stored within the app. Another core aspect that could be investigated would be the retrieval of neighbouring AP data and RSSI lists as it would allow for a more detailed interpretation to be carried out on the results of the testing. Other forms of testing could also be carried out, such as testing how the Web Service and MSE would handle a high number of concurrent users. Testing the application on different devices would also provide a good insight on how RF is omitted by different Wi-Fi radios.

Overall, the intended outcome was achieved and the project progressed as planned. The software artefact provides a strong base that will allow for a more dynamic and commercial application. Using the full capability of the Cisco Context-Aware Mobility API this application could be developed to gather statistical information on client location and trends and could be incorporated with Cisco's Network Service Discovery (Kerner, 2012) where location specific information and deals is presented to the users.

7 References

- Abdul-Latif, O., Sheperd, P. and Pennock, S. (2007) 'TDOA/AOA Data Fusion for Enhancing Positoning in an Ultra Wideband System', Signal Processing and Communications, p1531-1534.
- Apache (2012) *Jax-WS Guide*, [Online], Available: http://axis.apache.org/axis2/java/core/docs/jaxws-guide.html.
- Atia, M.M., Korenberg, M. and Noureldin, A. (2012) 'A Consistent Zero-Configuration GPS-Like Indoor Positioning System Based on Signal Strength in IEEE 802.11 Networks', Position Locarion and Navigation Symposium, p1068-1073.
- Brown, D.R. and Dunn, D.B. (2011) 'Classification Schemes of Positioning Technologies for Indoor Navigation', Proceedings of IEEE Southeastcon, p125-130.
- Chang, N., Rashidzadeh, R. and Ahmadi, M. (2009) 'Differential Access Points for Indoor Location Estimation', Electro/Information Technology, p256-259.
- Chang, N., Rashidzadeh, R. and Ahmadi, M. (2010) 'Robust Indoor Positioning using Differential Wi-Fi Access Points', *Consumer Electronics*, vol. 56, no. 3, p1860-1867.
- Ching, W., Teh, R.J., Li, B. and Rizos, C. (2010) 'Uniwide WiFI Based Postitioning System', International Symposium onTechnology and Society, p180-189.
- Chon, Y. and Cha, H. (2011) 'LifeMAp: A Smartphone-Based Context Provider for Location-Based Services', Pervasive Computing, p58-67.
- Chun, S.M., Lee, S.M., Nah, J.W., Choi, J.H. and Park, J.T. (2011) 'Localization of Wi-Fi Access Point using Smartphone's GPS Information', Mobile and Wireless Networking, p121-126.
- Cisco (2006) *Wi-Fi Location-Based Services—Design and Deployment Considerations*, Cisco Systems, Inc.
- Cisco (2008a) 'Cisco Location Based Services Architecture', in *Wi-Fi Location Based Services Design Guide 4.1*, Cisco Systems, Inc.
- Cisco (2008b) 'Location Tracking Approaches', in *Wi-Fi Location Based Services Design Guide 4.1.*, Cisco Systems, Inc.
- Cisco (2008c) *The Technologies behind a Context-Aware Mobility Solution*, Cisco Systems, Inc.

- Cisco (2010a) 'Architectural Overview', in *Cisco Context Aware Mobility API*, Cisco Systems, Inc.
- Cisco (2010b) 'Benefits', in *Cisco Conext Aware Mobility API*, Cisco Systems, Inc.
- Cisco (2010c) *Cisco Context Aware Mobility Solution Technology*, Cisco Systems, Inc.
- Cisco (2010d) 'Introduction to Context-Aware APIs', in *Cisco Context-Aware API Getting Started Guide*, Cisco Systems, Inc.
- Cisco (2011) 'Cisco Wireless Control System Configuration Guide, Release 7.0.172.0.' Cisco Systems, Inc.
- Cisco (2012a) *Cisco Mobility Services Engine - Context Aware Mobility Solution Deployment Guide*, Cisco Systems, Inc.
- Cisco (2012b) 'Cisco MSE API Specification Guide-Context Aware Services of MSE, Release 7.2.' Cisco Systems, Inc.
- Cisco (2012c) 'Overview', in *Cisco Context-Aware Service Configuration Guide, Release 7.3*, Cisco Systems, Inc.
- Ciurana, M., Giustiniano, D., Neira, A., Barcelo-Arroyo, F. and Martin-Escalona, I. (2010) 'Performance stability of software ToA-based ranging WLAN', Indoor Positioning and Indoor Navigation, p1-8.
- Cypriani, M., Lassabe, F., Canalda, P. and Spies, F. (2009) 'Open Wireless Positioning System: A Wi-Fi-Based Indoor Positioning System', Vehicular Technology Conference Fall, p1-5.
- Cypriani, M., Lassabe, F., Canalda, P. and Spies, F. (2010) 'Wi-Fi-Based Indoor Positioning: Basic Techniques, Hybrid Algorithms and Open Software Platform', Indoor Positioning and Indoor Navigation, p1-10.
- Del Mundo, L.B., Ansay, R.L., Festin, C.A. and Ocampo, R.M. (2011) 'A Comparison of Wireless Fidelity (Wi-Fi) Fingerprinting Techniques', ICT Convergance, p20-25.
- Drane, C., Mac Naughton, M. and Scott, C. (1998) 'Positioning GSM Telephones', *IEEE Communications Magazine*, vol. 36, no. 4, p46-54, 59.
- Driscoll, C.J. (2013) *Access Science*, [Online], Available: http://www.accessscience.com/loadBinary.aspx?filename=YB001541FG0010.gif [8 April 2013].

- Elkamchouchi, H. and Mofeed, M.A. (2005) 'Direction-Of-Arrival Methods (DOA) and Time Difference of Arrival (TDOA) Position Location', Radio Science Confrence, p173-182.
- Etutorials (2013) *etutorials.org*, [Online], Available: http://etutorials.org/shared/images/tutorials/tutorial_158/fig436_01.jpg [8 April 2013].
- Fang, S.H. and Wang, C.H. (2011) 'A Dynamic Hybrid Projection Approach for Improved Wi-Fi Location Fingerprinting', Vehicular Technology, p1-6.
- Gallagher, T., Li, B., Dempster, A.G. and Rizos, C. (2010) 'Database Updating Through User Feedback in Fingerprint-Based Wi-Fi Location Systems', Ubiquitous Positioning Indoor Navigation and Location Based Service, p1-8.
- Google (2013) *AsyncTask*, [Online], Available: http://developer.android.com/reference/android/os/AsyncTask.html [28 April 2013].
- Google (2013) *developer.android.com*, 6 May , [Online], Available: http://developer.android.com/google/play/billing/billing_best_practices.html [6 May 2013].
- Hossain, M., Van, H.N., Jin, Y. and Soh, W.-S. (2007) 'Indoor Localization Using Multiple Wireless Technologies', Mobile Adhoc and Sensor Systems, p1-8.
- IBM (2013) *Securing web services applications at the transport level*, 9 July, [Online], Available: http://pic.dhe.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Ftwbs_secwsaatl.html [11 July 2013].
- Jagoe, A. (2002) *Mobile Location Services: The Definitive Guide*, Prentice Hall PTR.
- Kerner, S.M. (2012) *Enterprise Networking Planet*, 15 November, [Online], Available: http://www.enterprisenetworkingplanet.com/nethub/indoor-gps-cisco-debuts-new-location-awareness-technology.html [08 Aug 2013].
- Kim, Y., Chon, Y. and Cha, H. (2012) 'Smartphone-Based Collaboative and Autonomous Radio Fingerprinting', *Systems, MAn and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 1, p112-122.

- Kim, J.I., Lee, J.G. and Park, C.G. (2008) 'A Mitigation of Line-of Sight by TDOA Error Modelling In Woreless Communication System', International Conference on Control, Automation and Systems, p1601-1605.
- Kim, Y., Shin, H. and Cha, H. (2012) 'Smarphone-based Wi-Fi Pedestrian-Tracking System Tolerating the RSS Variance Problem', Pervasive Computing and Communications, p11-19.
- Koo, J. and Cha, H. (2012) 'Unsupervised Locating of WiFi Access Points Using Smartphones', *Systems, Man and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 1, p1341-1353.
- Kos, T., Grgic, M. and Kitarovic, J. (2007) 'Location Technologies for Mobile Networks', International Workshop on Systems, Signals and Image Processing, p319-322.
- Lawson, S. (2013) *Info World*, 22 May, [Online], Available: • http://www.infoworld.com/t/mobile-technology/growing-mobile-malware-threat-swirls-mostly-around-android-219147?source=IFWNLE_nlt_sec_2013-05-23 [30 May 2013].
- Le, T.D., Le, H.M., Nguyen, N.Q., Tran, D. and Nguyen, N.T. (2011) 'Convert Wi-Fi Signals for Fingerprint Localization Algorithm', Wireless Communications, Networking and Mobile Computing, p1-5.
- Leu, J.S. and Tzeng, H.J. (2012) 'Received Signal Strength Fingerprint and Footprint Assisted Indoor Positioning BAsed on Ambient Wi-Fi Signals', VEHICULAR Tehnology Confrence, p1-5.
- Liu, H.H. and Yang, Y.N. (2011) 'WiFi-Based Indoor Positioning for Multi-Floor Environment', TENCON, p597-601.
- Llombart, M., Ciurana, M. and Barcelo-Arroyo, F. (2008) 'On the scalability of a novel WLAN positioning system based on time of arrival measurements', Workshop on Positioning, Navigation and Communication, p15-21.
- Lui, G., Gallagher, T., Li, B., Dempster, A.G. and Rizos, C. (2011) 'Differences in RSSI Readings Made by Different Wi-Fi Chipsets: A Limitation of WLAN Localization', Localization and GNSS, p53-57.

- Meng, W., Xiao, W., Ni, W. and Xie, L. (2011) 'Secure and Robust Wi-Fi Fingerprinting Indoor Localization', Indoor Postitioning and Indoor Navigation, p1-7.
- Morgan, Y. (2009) 'Accurate Positioning Using Short-Range Communications', Ultra Modern Telecommunications & Workshops, p1-7.
- Niculescu, D.S. and Nath, B. (2003) 'Localzed Positioning in Ad Hoc Networks', Sensor Network Protocols and Applications, p42-50.
- Powell, A. (2009) *Android Developers Blog*, 2010 June, [Online], Available: http://android-developers.blogspot.ie/2010/06/making-sense-of-multitouch.html [10 Feb 2013].
- Retscher, G. and Fu, Q. (2010) 'Continuous Indoor Navigation with RFID and INS', Position Location and Navigation Symposium, p102-112.
- Shin, H. and Hojung, C. (2010) 'Wi-Fi Fingerprints-Based Topological MAp Building for Indoor User Tracking', Embedded and Real-Time Computing Systems and Applications, p105-113.
- Shin, B.J., Lee, K.W., Choi, S.H., Kim, J.Y., Lee, W.J. and Kim, H.S. (2010) 'Indoor WiFi Positioning System for Android-based Smartphone', Information and Communication Technology Convergence, p319-320.
- Shu, X., Du, Z. and Chen, R. (2009) 'Research on Mobile Location Service Design Based on Android', Wireless Communications, Networking and Mobile Computing, p1-4.
- Wong, C., Klukas, R. and Messier, G. (2008) 'Using WLAN Infrastructure for Angle-of Arrival Indoor ser Location', Vehicular Technology Conference, p1-5.
- Xiao, W., Ni, W. and Toh, Y.K. (2011) 'Integrated Wi-FI Fingerprinting and Inertial Sensing for Indoor Positioning', Indoor Positioning and Indoor Navigation, p1-6.
- Yu, Y., He, J., Wang, Q., Liu, F. and Xu, C. (2012) 'A Query-driven Indoor Location System based on Smartphone', Enabling Tehnologies for Smartphone and Internet of Things, p25-29.

8 Appendix

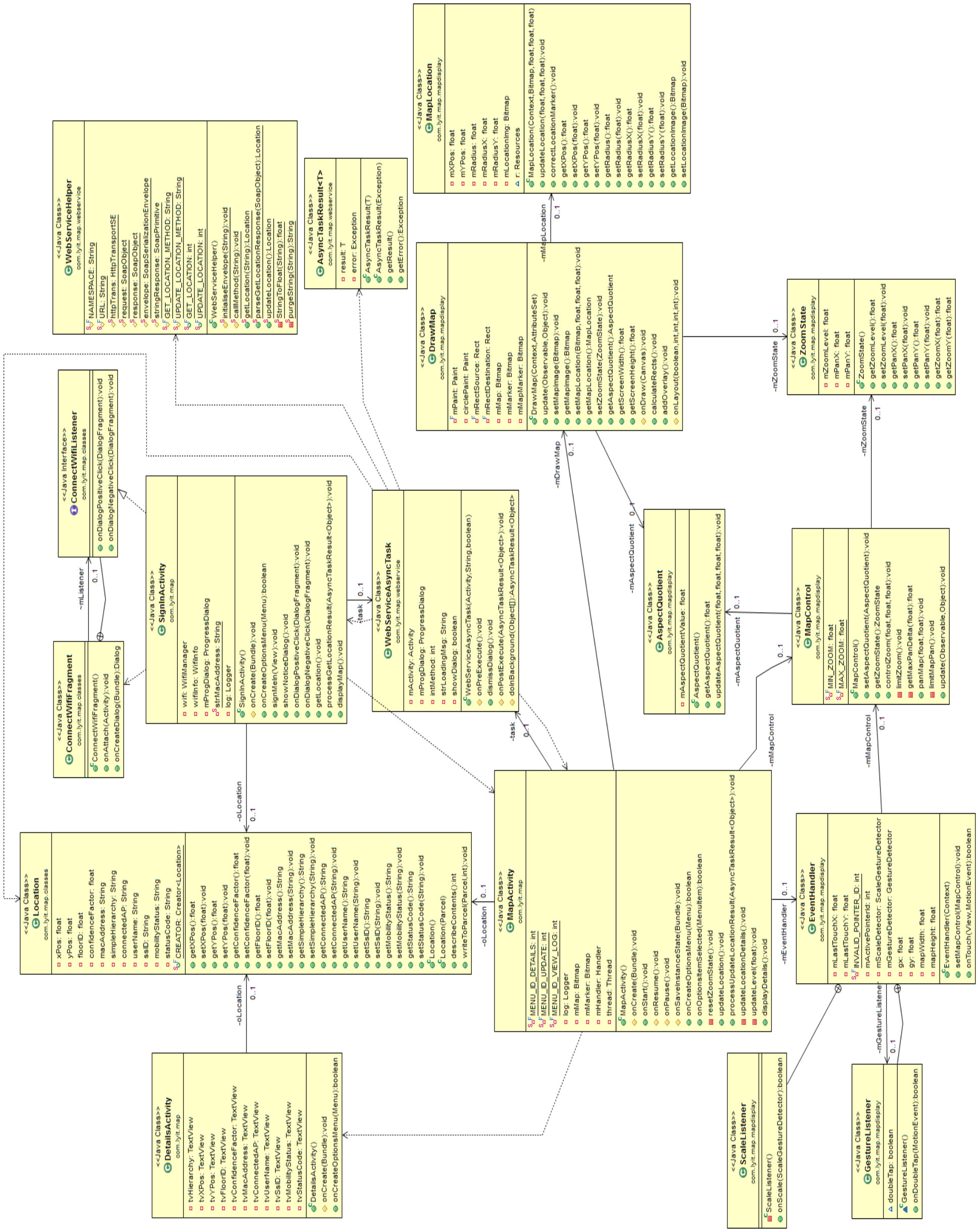

Figure 8.1: Android App Class Diagram

Testing Locations

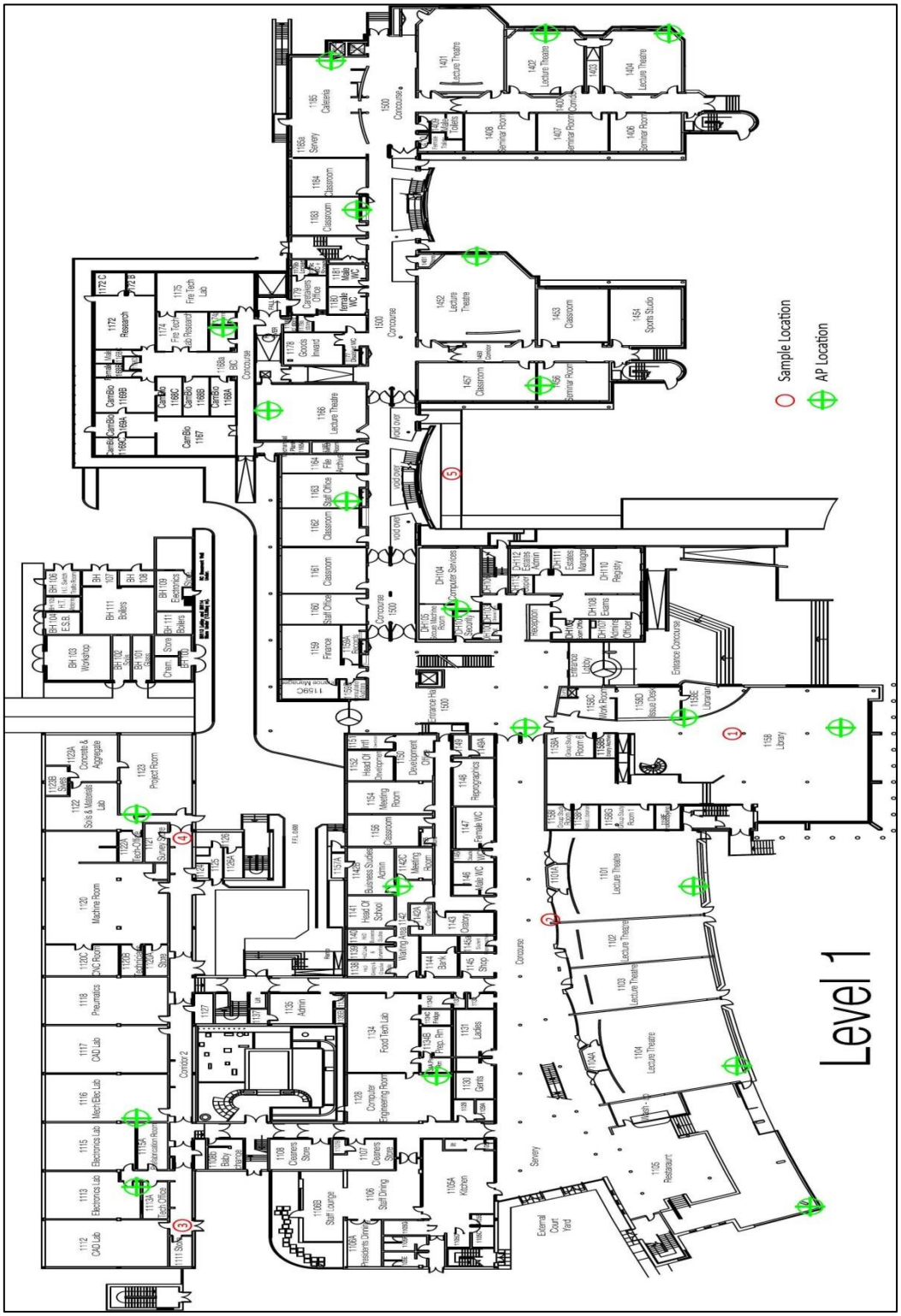

Figure 8.2: Level 1 Test Locations

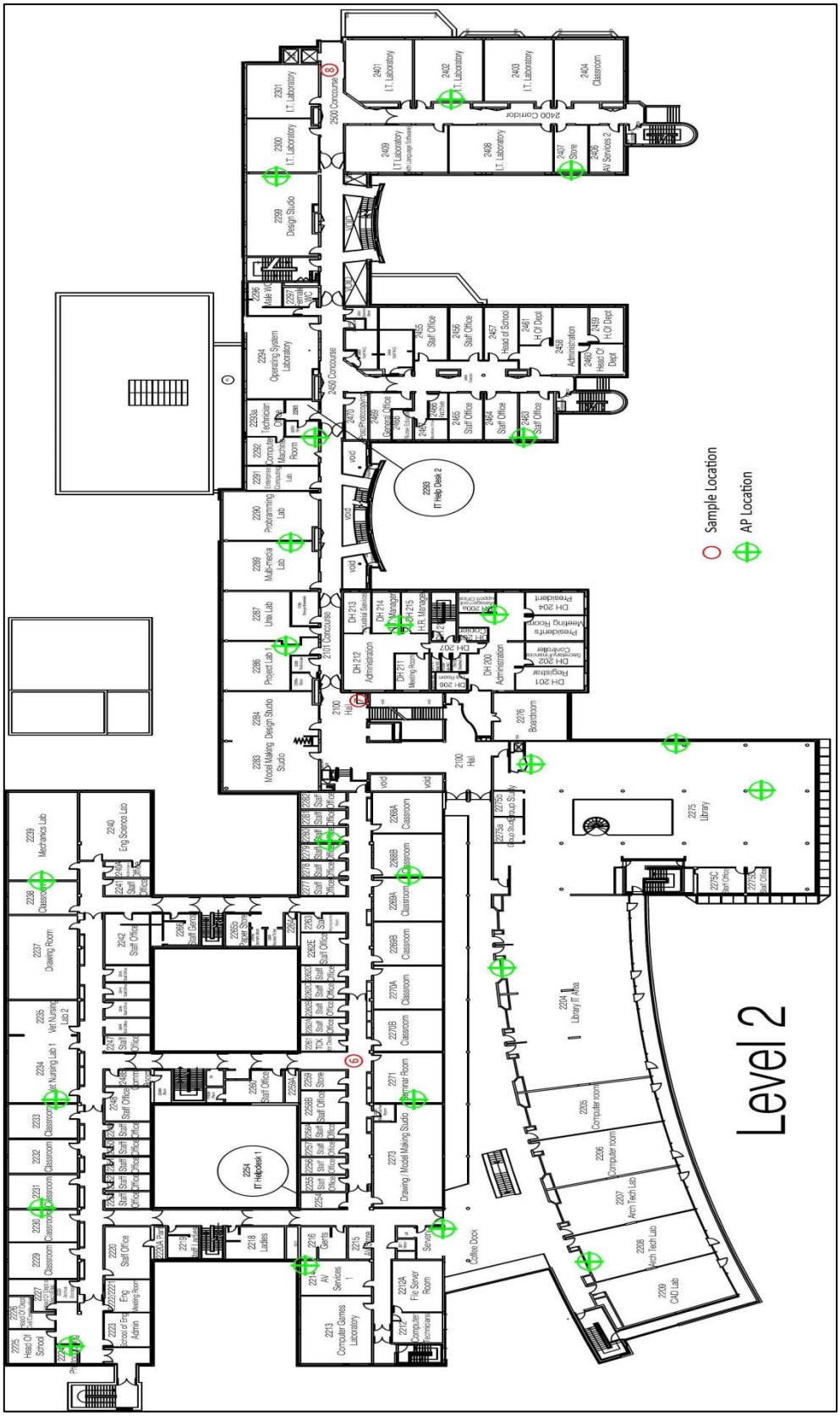

Figure 8.3: Level 2 Test Locations

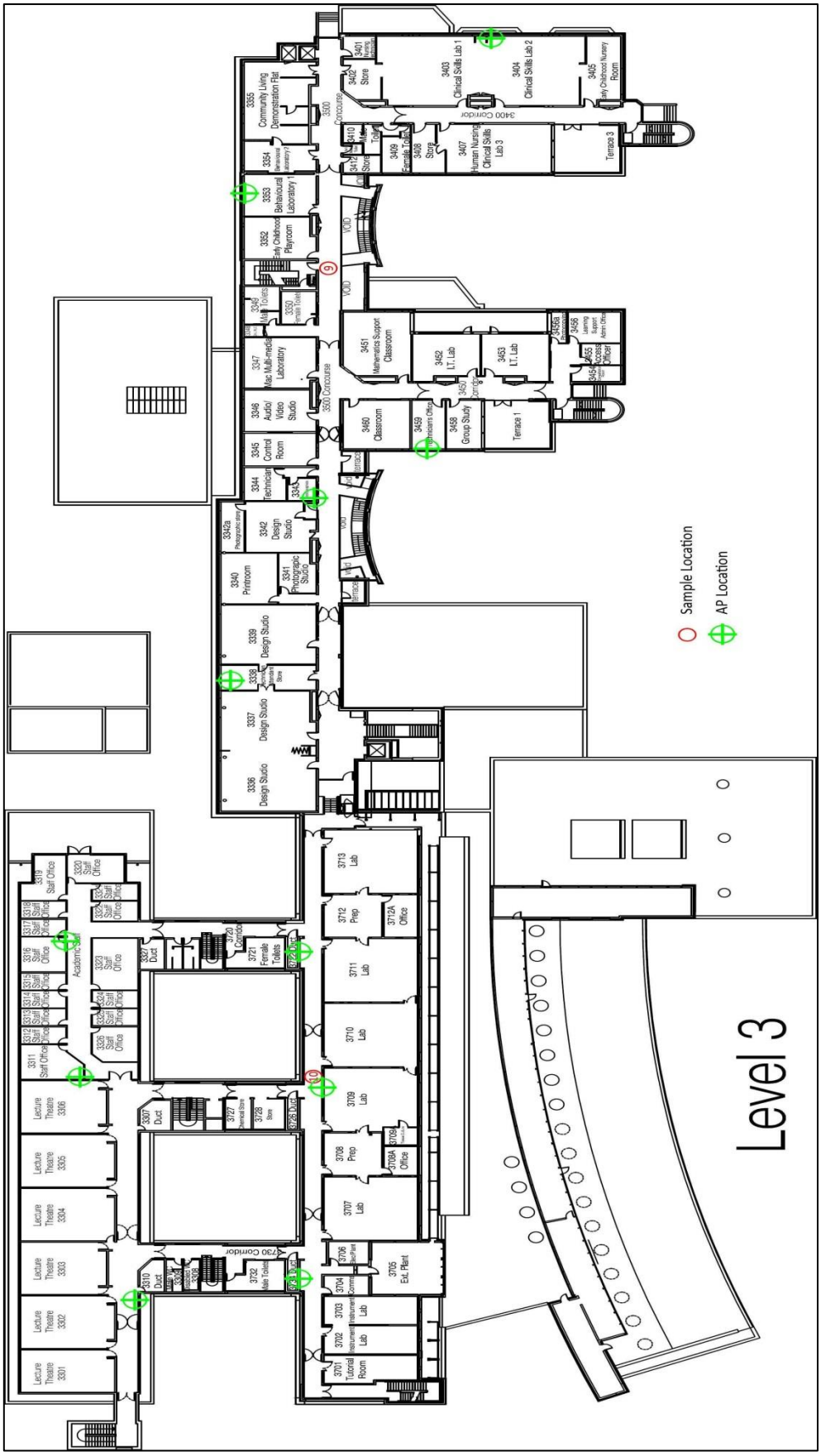

Figure 8.4: Level 3 Test Locations

Testing Results

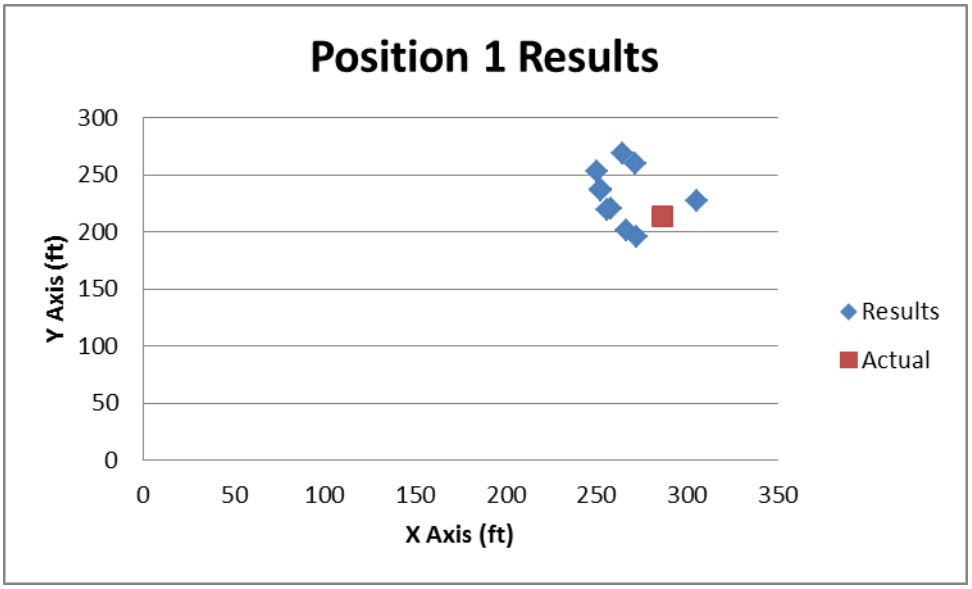

Figure 8.5: Position 1 Results - Scatter Diagram

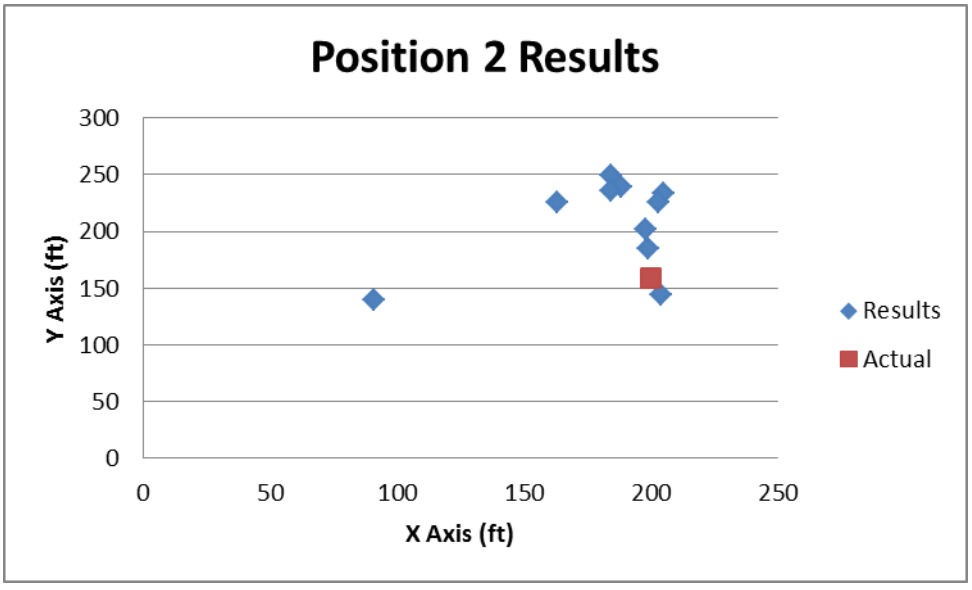

Figure 8.6: Position 2 Results - Scatter Diagram

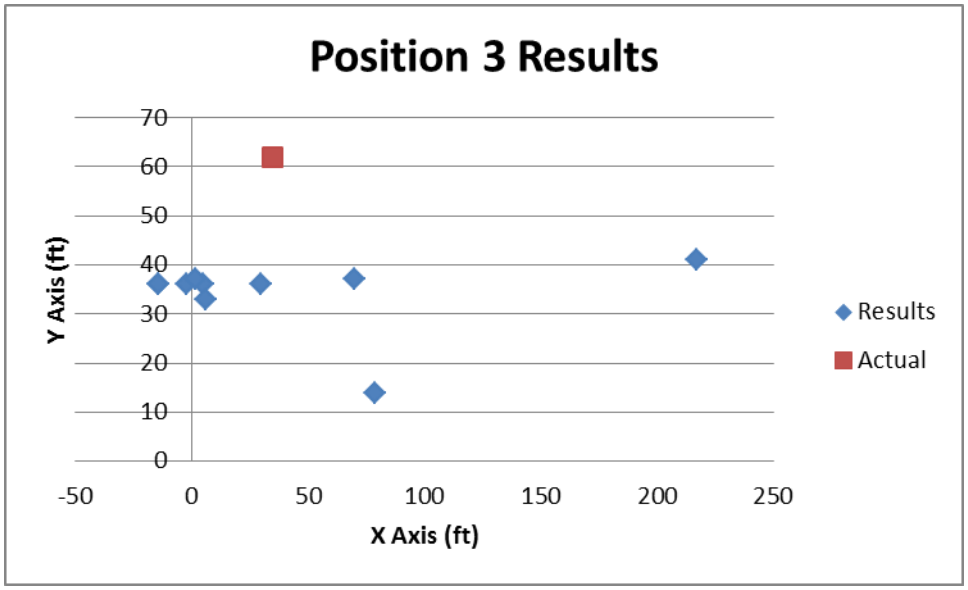

Figure 8.7: Position 3 Results - Scatter Diagram

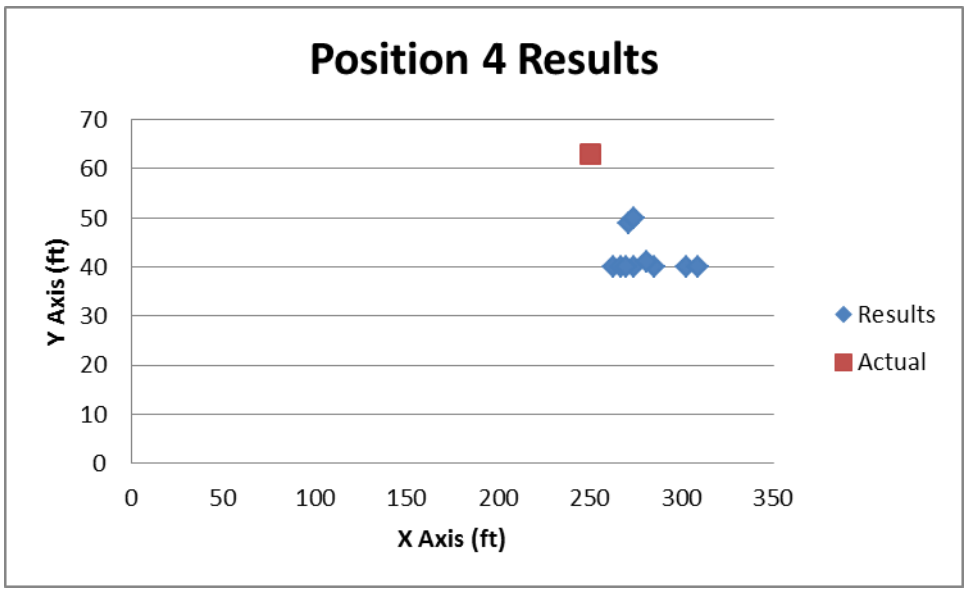

Figure 8.8: Position 4 Results - Scatter Diagram

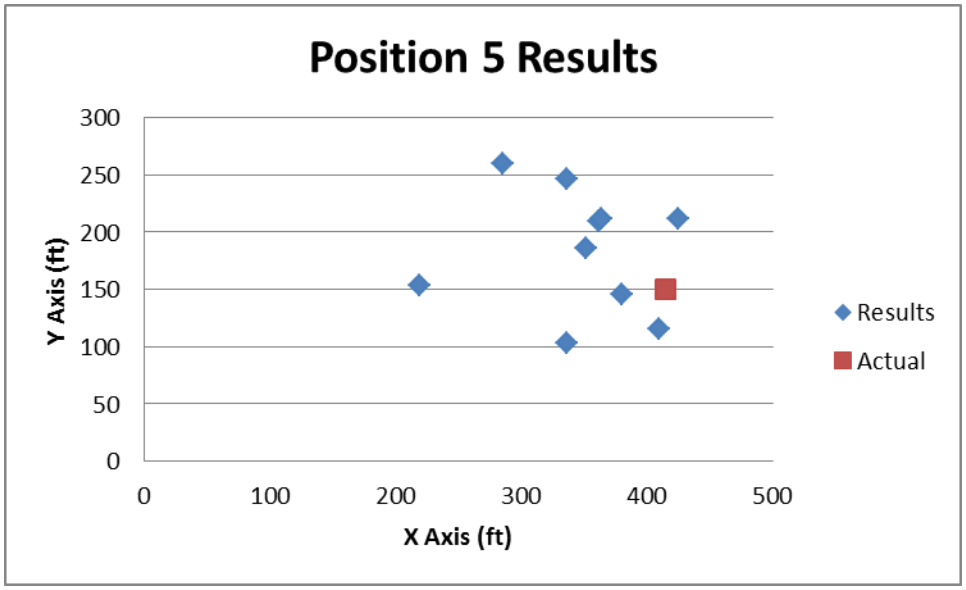

Figure 8.9: Position 5 Results - Scatter Diagram

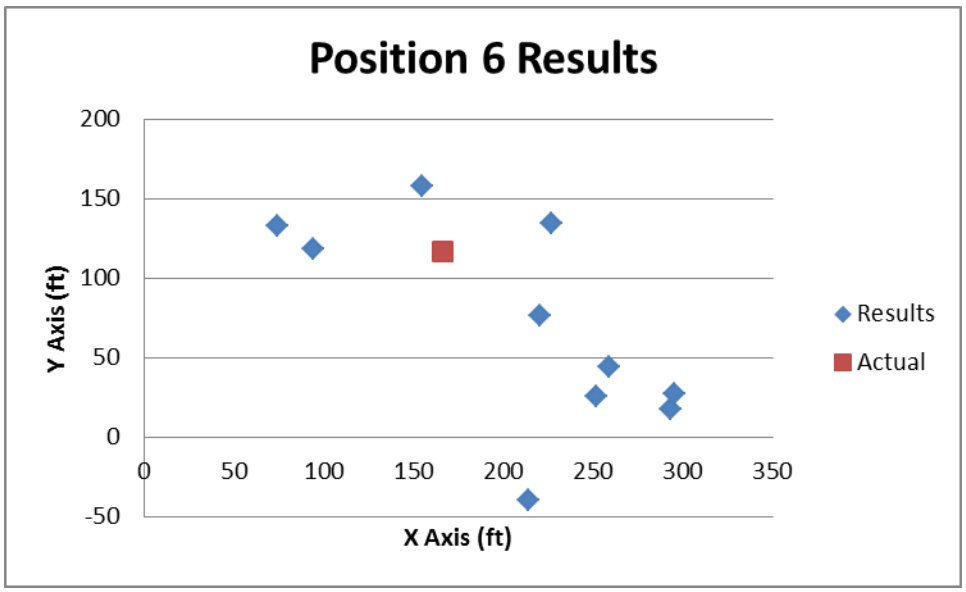

Figure 8.10: Position 6 Results - Scatter Diagram

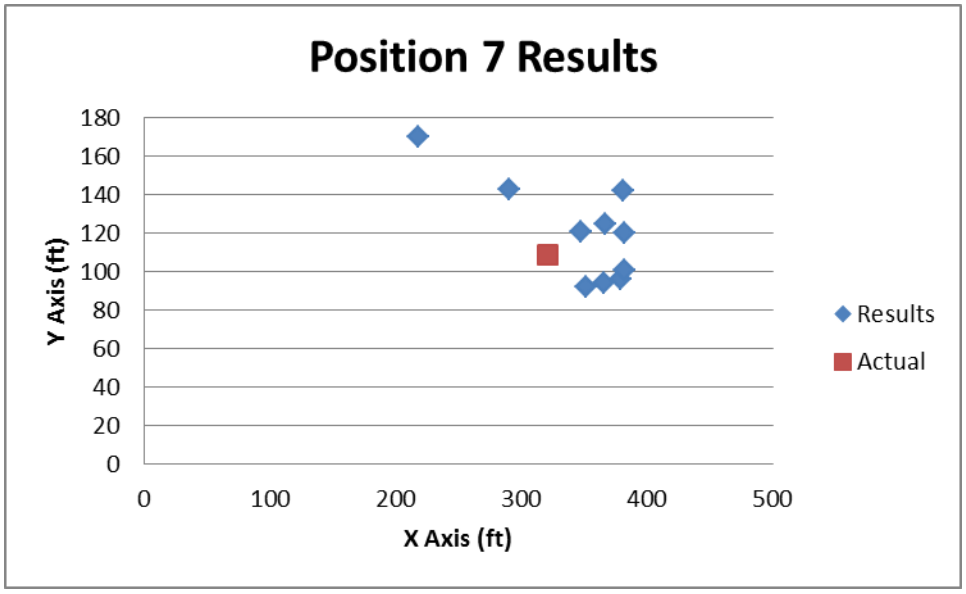

Figure 8.11: Position 7 - Results Scatter Diagram

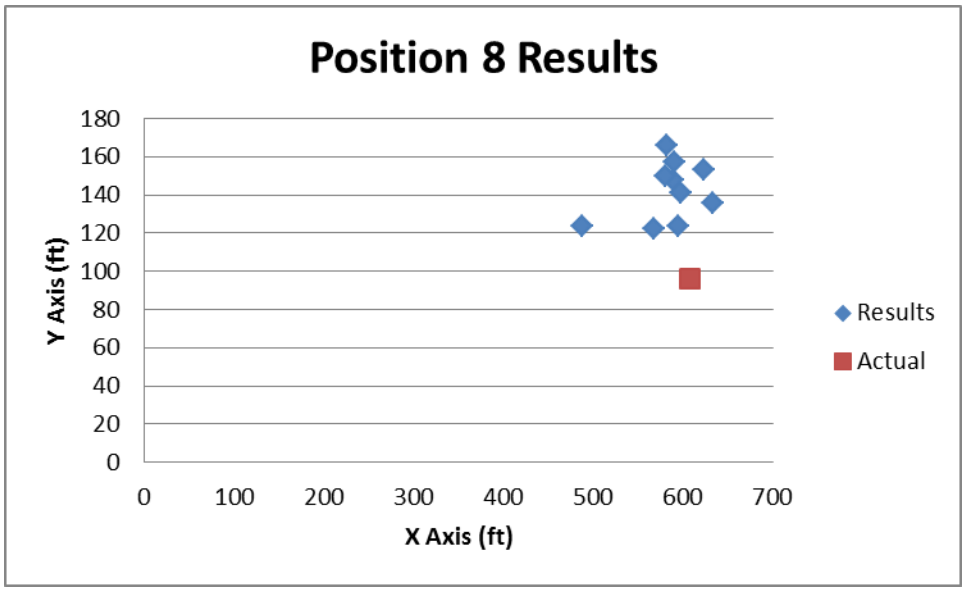

Figure 8.12: Position 8 Results - Scatter Diagram

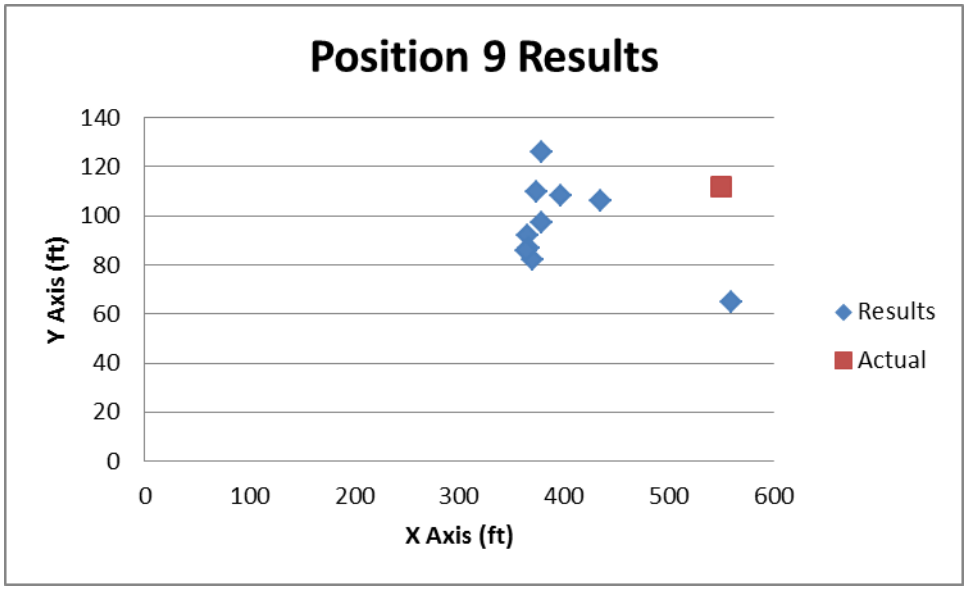

Figure 8.13: Position 9 Results - Scatter Diagram

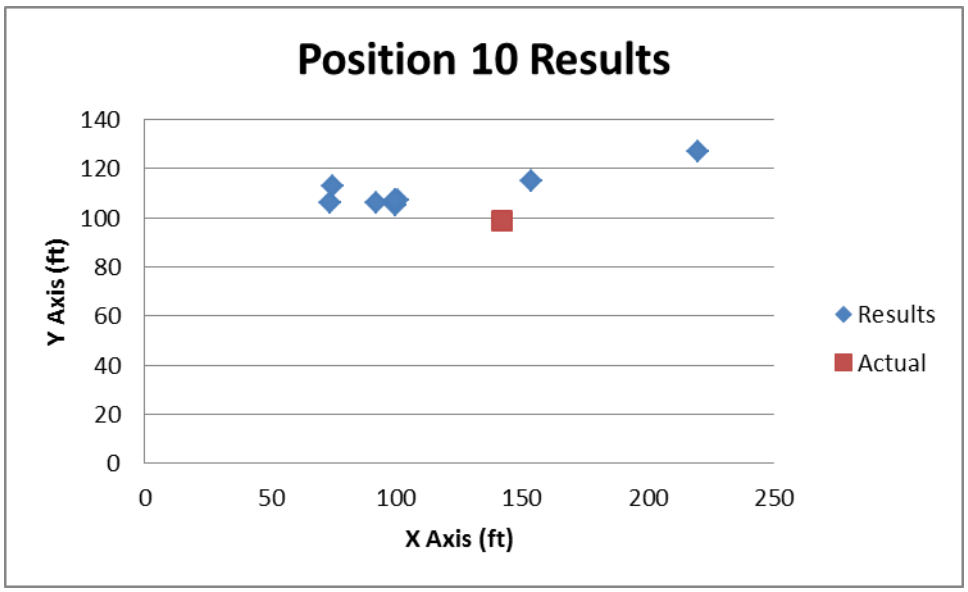

Figure 8.14: Position 10 Results - Scatter Diagram

Application Screenshots

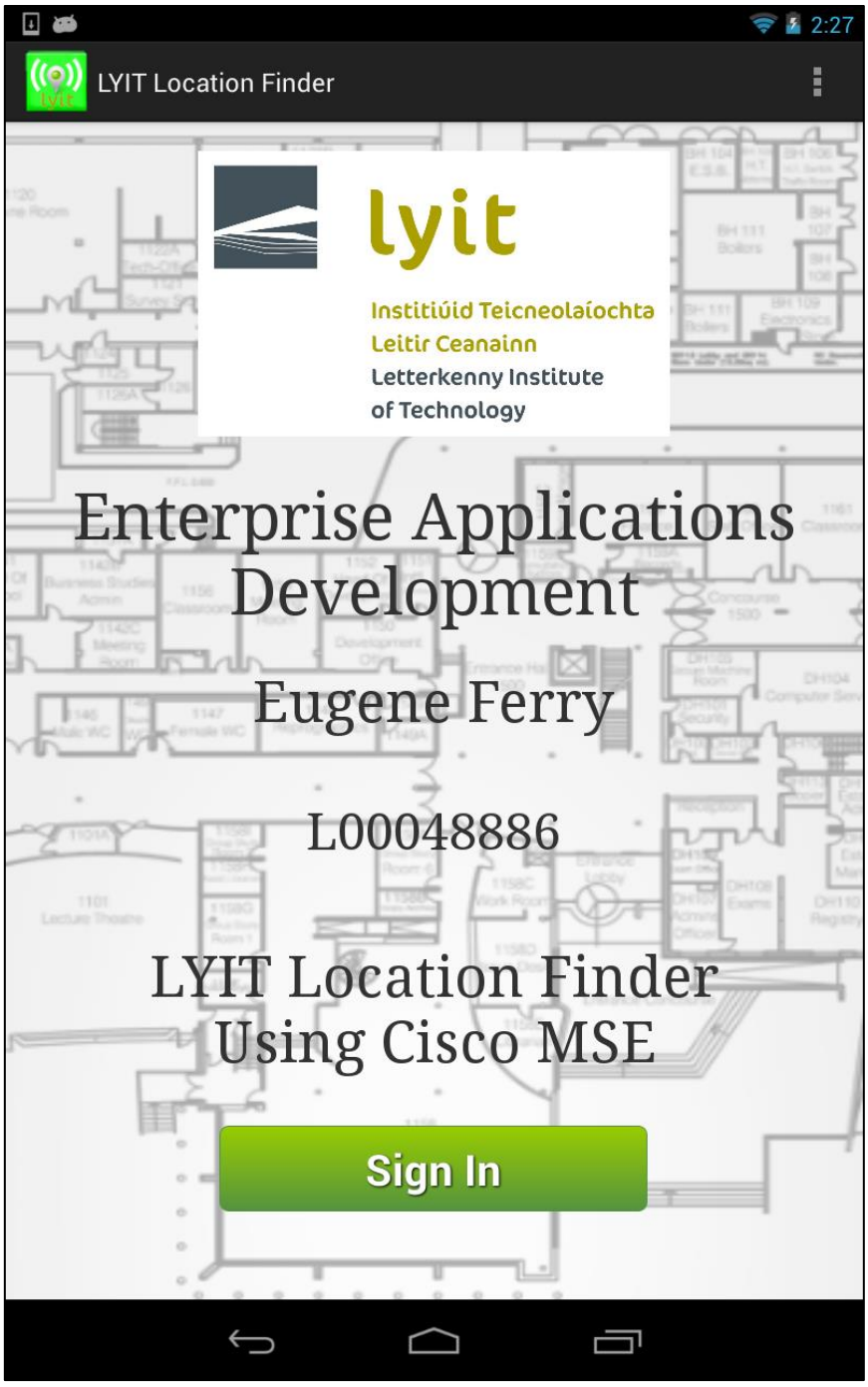

Figure 8.15: SignIn Activity

Figure 8.16: Map Activity

Figure 8.17: Details Activity