IEEE 802.11 Signal Source Mapping using Low Cost Spectrum Analysers

Submitted in partial fulfilment
of the requirements of the degree of
Bachelor of Science (Honours)
of Rhodes University

Daniel David Wells

5th November 2007

Abstract

The research presented by this dissertation, describes an investigation into visualising 2.4 GHz Wi-Fi networks and discovering the location of 2.4 GHz devices using a collection of low cost MetaGeek WiSpy 2.4 GHz Spectrum Analysers. By combining frequency VS amplitude data received from WiSpy spectrum analysers and graphically displaying the data we can effectively discover 2.4 GHz Wi-Fi devices and interference sources. Research was done into interference sources that hamper the performance of Wi-Fi throughput, these include any radio frequency devices which operate in the 2.4 GHz band.

To practically meet these goals the WiSpy SSM Tool was produced, which includes two applications (WiSpy SSM Collector and WiSpy SSM Compiler) and an webservice. Multiple WiSpy SSM Collector applications are deployed in the area under investigation and collect signal data using a WiSpy spectrum analyser connected to a computer. The collected data is transmitted to the webservice and stored in a database. The WiSpy SSM Compiler retrieves stored information and displays this to the user in a simple and easy to understand manner for interpretation. The results obtained from the WiSpy SSM Compiler were conclusive in that by using three spectrum analysers we were able to locate Wi-Fi devices fairly accurately.

Declaration

The work in this thesis is based on research carried out at Rhodes University, South Africa. No part of this thesis has been submitted elsewhere for any other degree or qualification and is all my own work unless referenced to the contrary in the text.

Acknowledgements

I would firstly like to thank my supervisors, Barry Irwin and Ingrid Siebörger for their limitless support throughout the year. Thank you for making sure my project was always heading in the best direction. I would like to thank Tom Wells for his software development advice, tips and hints. I would also like to thank my parents for their unconditional support which makes me reach higher goals every day.

I would specifically like to acknowledge the support from all my friends, who have been a pillar of strength for me this year.

Lastly I would like to take this opportunity to acknowledge the financial and technical support from the department of Computer Science at Rhodes University, Telkom SA, Business Connexion, Comverse SA, Verso Technologies, Stortech, Tellabs, Amatole, Mars Technologies, Bright Ideas Projects 39 and THRIP through the Telkom Centre of Excellence at Rhodes University.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Recent years have seen the rise of mobile users with PDA's, laptops and mobile phones, at the same time, computer network connectivity is becoming increasingly integral into a vastly increasing wireless environment. With the added convenience of wireless networking, new opportunities and threats have arisen. The integration of Wi-Fi technology into most areas has led to improved communications and an added strain on existing infrastructure.

Wireless networking has brought computer networks into a new, exciting and hostile environment. Factors that need to be considered and understood during implementation of wireless networks include interferences and security protocols. Setting up a Wireless Local Area Network (WLAN) is relatively simple, allowing users to achieve mobility, but in some cases, the default security configuration on the devices leads to inferior security measures implemented. Security leads to a higher implementation complexity.

Better utilisation of the 2.4 GHz radio frequency (RF) can be achieved by assessing the current Wi-Fi spectrum usage before a network administrator installs a wireless access point (AP). By considering the site location for a Wi-Fi network before installing hardware the wireless network can be used to it's full potential by minimising interference and operating over the best possible channel.

A Wi-Fi AP can be connected at any open port on a network, allowing unauthorised external users to gain access to the secure internal network, bypassing a vast majority of security measures. Discovering the location of rogue (unauthorised) APs aids the wireless user in protecting their secured network. Using spectrum analysis and trilateration, these rogue AP's can be located. In addition, the spectrum analysers can help network administrators better plan their wireless networks before deployment, hopefully minimising interference.

In this project we will be dealing specifically with 802.11b/g/n wireless technologies as they use the 2.4 GHz frequency and are the most common Wi-Fi devices.

1.1 Problem Statement

IEEE 802.11b/g/n Wi-Fi use the 2.4 GHz frequency [11,12]. As these technologies becoming increasingly popular for the home and business, the 2.4 GHz frequency is becoming cluttered therefore a need for optimal use of the medium is required. Wi-Fi throughput can be increased by selecting the best Wi-Fi channel, minimising interferences and removing rogue access points (APs). By combining the frequency VS signal amplitude data from three (or more) MetaGeek WiSpy 2.4 GHz spectrum analysers [21] it is possible to locate 2.4 GHz interferences and Wi-Fi devices. The data from the spectrum analysers is combined to produce a graphical display of a Wi-Fi network and devices are located using the method of trilateration.

1.2 Research Goals and Outcomes

The main goal of this project is to utilise the low cost WiSpy spectrum analyser and associated software to improve and build our own software solution that tracks spectrum use and can discover RF signal sources fairly accurately. This requires multiple collecting clients (WiSpy SSM Collector) which have WiSpy devices connected transmitting their signal data to a central repository (a webservice interface to a database). The collecting client needed to provide feedback to the user via a line graph of the frequency VS signal amplitude data read from the WiSpy device, this would aid in discovering interferences and choosing the least cluttered Wi-Fi channel. The compiling client (WiSpy SSM Compiler) needed to be designed to obtain the signal data from the repository and display it for the user to analyse. The compiler was to present a graphical display of a portion of the Wi-Fi network, with the different channels displayed and their associated signal strength from each collecting client node.

1.3 Document Structure

This dissertation begins in Chapter 2 by researching previous work in the realm of the 802.11 Wi-Fi standard and additional pertinent information to aid in producing a tool to meet the research outcomes. The difference between triangulation and trilateration is discussed. Security on Wi-Fi networks is researched, specifically the strengths and weaknesses of Wi-Fi security protocols WEP and WPA, and recommendations for security are given. Interference sources are researched and listed. It is recommended that interferences are kept at a minimum to allow maximum utilisation of the Wi-Fi network. Typical Wi-Fi devices and site surveys are discussed. The MetaGeek WiSpy spectrum analyser is researched and understood. Chapter 2 concludes that by using the WiSpy spectrum analyser it is possible to discover devices in the Wi-Fi network.

Chapter 3 delves into the the design of the tool which is used to discover Wi-Fi devices by using multiple spectrum analysers. Chapter 3 discusses the high level design of the WiSpy SSM Tool which provides an overall conceptual look at how it is structured and the numerous features that it provides. The tool is separated into three parts, the WiSpy SSM Collector, the webservice to store data and the WiSpy SSM Compiler. Each application's design is discussed in further detail.

Chapter 4 implements the design proposed in Chapter 3. The development platform choices used in implementation are discussed along with additional components to interface to the WiSpy spectrum analyser, use a SQLite [35] database and provide dynamic location updates using a GPS. Each application's implementation decisions are provided and discussed namely the user interfaces, important features and how data is transmitted and queried.

Chapter 5 evaluates the WiSpy SSM Collector and WiSpy SSM Compiler applications for accuracy of the results obtained.

Chapter 6 summarises the project conclusions, compares the results obtained to original goals and provides possible future extensions to this tool.

Chapter 2

Related Work

2.1 Introduction

The IEEE 802.11 family of technologies has been adopted on a global scale, not just in desktops, laptops and PDAs but in equipment diverse as mobile phones, parking meters, security cameras and home entertainment equipment, making Wi-Fi near ubiquitous [37].

Security in wireless networking has had to overcome some hurdles, with default settings on hardware most frequently set to the least secure mode to aid in user friendliness. The first Wi-Fi security protocol, WEP, was widely adopted and initially proved effective at preventing unwanted users gaining access. After scrutiny of the protocol by attackers and cryptologists, it was found that WEP had serious flaws. Security has since been improved with the release of WPA and WPA2, although still not widely adopted in the home or small office environment due to a lack of wireless education of the end-user.

Wireless networking, specifically Wi-Fi (802.11b/g/n) technologies, propagate over a cluttered band, 2.4 GHz, with a variety of other radio devices also operating in this frequency. Interference needs to be evaluated, understood and avoided otherwise network throughput will decrease. One particular method of evaluating interference is by conducting a site survey in the Wi-Fi environment to maximise network usage, but also keep unnecessary hardware and maintenance costs down.

Hardware and software to track user and AP locations have been produced, effectively locating devices to within a couple of meters. Proprietary hardware in this regard is expensive, although the cost of software running over an already existing wireless network is cheaper in comparison. The Wi-Spy 2.4 GHz spectrum analyser has been produced as a low cost site survey tool and has the potential to be used in location tracking.

Using three machines, in three locations, each with a Wi-Spy spectrum analyser,

Algorithm 1 Law of Sines

$$\frac{a}{sinA} = \frac{b}{sinB} = \frac{c}{sinC}$$

and by using their combined data collection (frequency and amplitude) together with trilateration it is possible to locate Wi-Fi devices within listening range.

2.2 Terminology

In this section, the terms trilateration and triangulation will be explained as well as the decibel milliwatts (dBm) unit of measurement. These concepts are important in helping understand how one is able to pinpoint the location of RF signal sources.

Trilateration is the method used in the Global Positioning System (GPS), allowing the location of a point to be discovered in space by using three known and fixed locations; although this method is similar to triangulation, no angle measurements are required. The measurement of signal strength (dBm) is used to approximate the distance that the signal has originated from. Using trilateration and this distance approximation from three points, we can discover an estimated location of the source of the signal.

2.2.1 Trilateration and Triangulation

Trilateration is the method of determining the relative positions of objects using the geometry of triangles in a similar fashion to triangulation. Triangulation uses angle measurements, with a minimum of one known distance, to find an object's location, where trilateration uses the known locations of two or more reference points and the measured distance between the target subject and each reference point [28].

Figure 2.1 shows a simple example of triangulation, with two known points, P_1 and P_2, the distance between them (d) and the two angles, a and b, to the third point P_3. Because of the nature of triangles, we know that the sum of all angles within a triangle is 180 degrees, which gives the angle c. Using the Law of Sines (see Algorithm 1), the lengths of the missing sides can be calculated [39].

Trilateration on the other hand, demonstrated in Figure 2.2, uses the distance from the three reference points to the object A. These distances will give the radii of three circles and object A will be at the intersection point. Having the radii of the circle and the centre points, we obtain the equations for the circles [28]. The equation for a 2-dimensional circle is seen in Algorithm 2 assuming that all the points are on the same plane. To find the

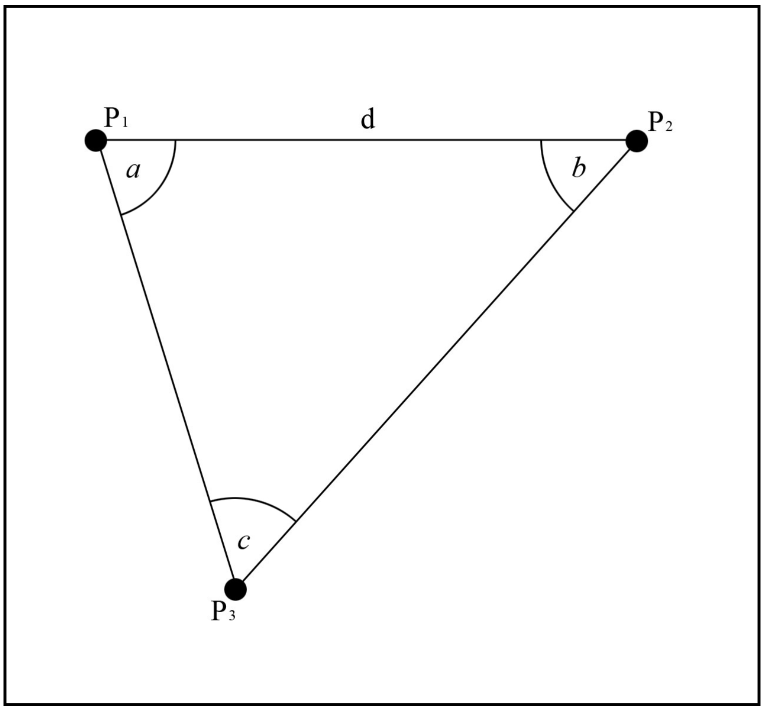


Figure 2.1: Triangulation

Algorithm 2 Two Dimensional Circle Equation

$$(x-a)^2 + (y-b)^2 = r^2$$

intersection point (x, y) of the circles, the equations of the three circles need to be solved simultaneously.

To solve for 3-dimensions (a sphere) the equation is extended to deal with the z-axis [28] as seen in Algorithm 3.

2.2.2 dBm units of Measurement

Signal strength in a wireless network is measured using dBm (decibel milliwatts), which is measured on a logarithmic scale. dBm and mW (milliwatts) both measure the power level, but dBm is most widely used as mW can become a very small meaningless number incredibly quickly. Devices will be marked with a receive sensitivity and a transmitter power output in this scale. An important fact about this scale is if you add 3 dBm, you

Algorithm 3 Three Dimensional Circle Equation

$$(x-a)^2 + (y-b)^2 + (z-c)^2 = r^2$$

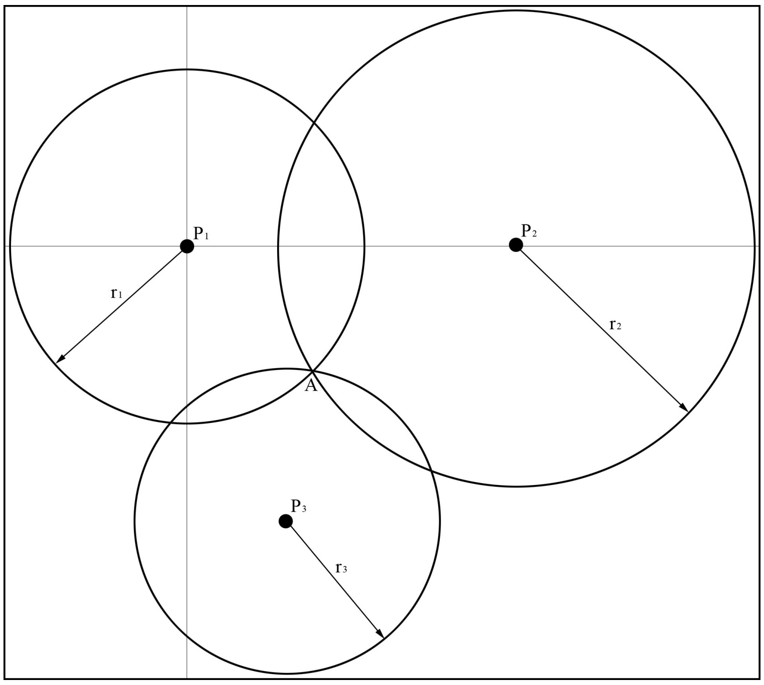


Figure 2.2: Trilateration

double the power output and subtracting 3 dBm will halve it [6].

This measurement is particularly useful when working out the distance a signal has travelled, if known at what strength the signal was transmitted.

2.3 Wi-Fi Security

Security is often viewed as a complicating factor with Wi-Fi and can be a difficult task to implement, even for skilled system administrators. Mobile users prefer the connection to be automated as they move between Wi-Fi networks, connecting to any AP, obtaining IP addresses, gateway information and DNS server addresses through DHCP. An automated connection process is acceptable in a highly mobile environment, but not when the network needs to be secure [5]. A secure wireless network, in theory, only allows authorised devices to connect and protects those devices from attack. Extensive overhead is carried out to secure the network which becomes increasingly complex as security improves, leading to weak security solutions implemented by the typical end-user. The responsibility placed on the user ranges from specifying the types of security protocol used and specifying passwords for AP's and clients, to managing a Public Key Infrastructure (PKI). A more secure network is more complicated to configure, leading to strong Wi-Fi security solutions being out of reach for typical end-users [5].

A trade-off exists between security and usability in any network, a more secure network significantly contributes to the cost of setting up and maintaining the network. In wireless networking environments, where there is a lack of physical barriers to access, a strong security implementation is crucial [5].

Numerous Wi-Fi security protocols exist to protect access to the network and prevent packets being interpreted by network packet sniffers. The original and outdated Wired Equivalent Privacy Protocol (WEP) and newer Wi-Fi Protected Access (WPA) and WPA2, together with their strengths and weaknesses will be discussed further.

2.3.1 Wired Equivalent Privacy Protocol (WEP)

WEP is the original security and encryption standard implemented in 802.11 wireless networks, which requires clients and AP's to share a single secret key which both use to encrypt all datalink layer communication [2]. The goals of WEP are to protect confidentiality, access control and integrity of user data from eavesdropping and tampering. This is the international standard and has been integrated by manufacturers into their 802.11 hardware, this protocol is still in widespread use [7].

WEP has many flaws (these are detailed further in Appendix A.1), and thus it is recommended to not use it in sensitive applications. At the time of its release it was considered secure, however after exploitations were released there was no alternative and marginal security is better than no security at all. Many companies strengthened WEP by deploying it together with other solutions like Virtual Private Networks (VPNs), 802.1x authentication servers and other proprietary software [40].

2.3.2 Wi-Fi Protected Access (WPA) and WPA2

Due to the vulnerabilities present in WEP, the numerous implemented attacks and concerns that a lack of strong Wi-Fi security would hinder the adoption of wireless devices in the market, a more secure protocol was necessary and WPA was introduced (WPA and WPA2 are detailed further in Appendix A.1).

WPA addresses all known vulnerabilities in WEP to ensure data authenticity and does so with a minimised impact on network performance. When WPA is properly installed, it ensures user data will remain protected and that only authorised users may access the network, it protects against some of the most targeted vulnerability attacks. Networks can now offer users the ease and flexibility of working wirelessly and securely without deploying add-on security solutions, such as VPNs [40].

WPA2 provides all the benefits of WPA, but uses a newer encryption scheme, the Advanced Encryption Standard (AES). The AES cipher algorithm employs variable key sizes of 128, 192 or 256-bits [40].

2.3.3 Recommendations on security for Wi-Fi

For a secure Wi-Fi network, under no circumstances should WEP be used, as multiple published attacks exist (see Appendix A.2 for details on published Wi-Fi networking tools and cracks) and allow attackers with basic network knowledge to gain access to the network. WPA or WPA2 is recommended in all situations where security is crucial [1,40]. Most Wi-Fi APs provide MAC address filtering, for added security this should be enabled as well.

2.4 Interference on WLANs

In this section interferences that can hamper the performance of 802.11b/g/n networks (2.4 GHz frequency) are discussed, together with the analysis of previous findings of performance degradation. We have decided to focus on 802.11b/g/n technologies because they tend to be near ubiquitous in the market place. 802.11a networks, which operate on a higher frequency of 5 GHz [10], are not commonly used, even though they operate on a less used band. 802.11a wireless is not discussed, as it is not as widely used on our campus, or in our testing.

2.4.1 Typical interference sources found

Wi-Fi interference can be separated into two broad categories, traffic from other users of the network and that arising from non-WLAN traffic operating in the same frequency (in this case 2.4 GHz) [34]. Non-WLAN traffic is any RF source which operates in the same frequency band as the wireless network, this includes a range of cordless phones, any Bluetooth device, cordless headsets, wireless bridges, cordless video-game controllers and microwave ovens [13]. A microwave oven can create interference from up to 50 feet away (15 meters) and incur relatively high packet retransmission [18]. Any source that has the same propagation medium as the wireless network will corrupt the signal reception [41]. Obstructions between antennas also lead to reduced throughput because the radio link depends on the energy diffracted around the object rather than direct radiation [8].

Traffic from other wireless users is of the most concern to those living in densely populated residential areas, or multi-tenant office buildings where wireless networks tend to be prevalent.

2.4.2 Wireless Denial-of-Service attack (WDoS)

Another source of interference, which is more malicious than those already discussed is produced by a simple device that can be either bought or built, which can be concealed and effectively prevent any wireless transmissions occurring in a given radius. These types of devices output onto a particular band and transmit either Gaussian white noise or a similar relatively high amplitude signal. These devices are illegal, yet plans and kits can be bought on the Internet [13].

2.4.3 Effects of interference and how to evaluate them

Depending on what sort of traffic the WLAN is being used to transport, the effects appear in different ways. In a scenario of asynchronous traffic (typical data), users will experience decreased or variable throughput resulting in low or non-existent service rates, in this case the signal strength meter on the user's computer will continue to suggest that all is well, even though throughput is down. It could also be the case that the signal strength meter is representing the strength of other wireless networks' signal.

Voice over IP over Wi-Fi (VoFi) systems affected by interference will experience dropouts (silent periods), as packets are lost due to collisions and this can occur in both directions of the call. These effects will be familiar to cell phone users, although in this case dropouts are caused by interference and not a fading signal.

Video being streamed over Wi-Fi is another scenario where high throughput is required to provide the user with real time video. Users will experience dropouts, square boxes in the video and freezing when interference exists. These glitches occur when key frames are lost from the compression of the video and can only correct itself when the next key frame is obtained [13].

A complicating factor when monitoring interference is typical network congestion in either wired or wireless networks that can have similar detrimental results. For interference analysis, two forms exist, protocol analysis (evaluating the response to interference on a given protocol) and energy analysis (using a spectrum analyser to evaluate effects) [13].

When conducting interference testing, a baseline benchmark of the throughput needs to be obtained without any interference and with a repeatable workload. Next, impairment testing is run identically to the benchmark, but with a known interference introduced. These results can be compared to evaluate the effects of the interference [13].

2.4.4 Recommendations

Interferences sources need to be controlled and managed effectively, allowing full use of the wireless network.

2.5 Considerations and Asset Tracking

Every building is different in the way they are built, how thick the walls are and where filing cabinets and staircases are situated. All these factors need to be considered when installing a Wi-Fi network, to maximise throughput and keep users happy. Wireless site surveys provide network planners with detailed reports about types of antennas to use, number of AP's and interference sources in the environment.

Detecting and locating devices and clients in the network (authenticated or rogue) may provide administrators with advantages in securing their network and controlling interference. In addition to the above, this section discusses typical Wi-Fi devices.

2.5.1 Wireless Site Surveys

WLAN applications have become increasingly popular over the past decade and become a core part of an enterprise communication infrastructure where it provides real-time access to information either on the Internet or intranet. These applications are expected to reliably support a significant amount of users and mission-critical services. Due to the current and predicted future increase of Wi-Fi networks, it is clear that the completion of an effective RF site survey is necessary before installing APs [18].

Unreliable wireless coverage could prevent network connectivity, limit network capacity, cause dropped network connections that may lead to thousands of rands in productivity lost as well as troubleshooting and maintenance costs as the organisation is disrupted.

The primary goal of performing a site survey is to determine the number and location of AP's that provide optimum signal strength for the organisation. A survey should be completed prior to installation, allowing the correct placement of AP's and a sufficient amount of signal overlap. The report from a site survey, together with a detailed map of the area, should supply the exact mounting points with enough detail so that the installation of actual AP's later will produce similar RF signal coverage. The best channel to utilise should also be included in the report. In addition, a site servery will allow network designers to locate any sources of interference that could negatively impact on the wireless network.

Issues with radio signals are that they do not propagate in equal distances in all directions as obstacles such as walls, filing cabinets and other interferences discussed previously cause more or less signal attenuation. In the case of a microwave oven, an AP may have to be placed near that area for users to have lessened effects of the interference.

A survey should offer valuable information regarding the choice of antennas, whether they be directional or not and correctly placed to ensure boundaries inside and outside the building, and that no coverage holes exist [6].

2.5.2 Location Tracking

The mass production of portable computing devices (laptops, PDA's, even some mobile phones) have allowed users to remain connected whilst moving about inside buildings. This has generated a lot of interest in applications and services regarding the mobile user's physical location. Location information allows effective and targeted use of the mobile environment. Such advantages include printing a document to the closest printer, locating a mobile user within the environment or guiding a user through a building [4].

The granularity of the information needed varies from one application to another. For example, selecting which printer to use would require coarse-grained location information, whilst locating a book in a library would require fine-grained information. In general, the amount of precision required dictates the cost and complexity of the location tracking system [4].

Cisco Systems Inc. provides the Wireless Location Appliance, a proprietary solution to location tracking. The price for this device varies between $8500 and $10500 USD (`www.pcmall.com`; `www.cdw.com`). As a part of the Cisco Unified Wireless Network, this is a costly solution but can efficiently track devices to within a couple of meters. The device has a fully integrated Application Programming Interface (API), and many applications can utilise the data received to provide multiple services [9].

RADAR User Location and Tracking System is a software-only system built over an off-the-shelf RF wireless local area network. The location-aware services enabled by RADAR compliment an already useful data network, making the WLAN more valuable and reducing costs of specialized location tracking hardware. This software is able to track a mobile user to within a few meters of their location [4].

This project provides similar functionality to location tracking, where the location of active Wi-Fi devices can be estimated using low cost spectrum analysers.

2.5.3 Standard devices found on a Wi-Fi network

Two types of communication devices exist, clients and Wi-Fi APs. Clients can either connect to each other in a peer-to-peer configuration (meshed network structure) or by introducing an AP to provide a star network structure. Often an AP is a bridge between the wireless and wired portions of the network, allowing clients to use existing network infrastructure, but have the advantage of being mobile.

Peer-to-peer arrangements are typically used as a temporary solution to connect two clients in an ad-hoc fashion for short periods. Whereas an AP provides many network services (including DHCP and DNS) and attempts to prevent difficulties in media access control, typically in the case of the masked node (or hidden node) problem (one node trying to talk to a second node, whilst a third unseen node is attempting to do the same) [32].

2.5.4 Section Summary

A site survey is an extensive, labour intensive task to collect signal propagation data around the building being surveyed; a spectrum analyser can speed up this process. A low-cost spectrum analyser and its capabilities are explained in detail in the next section.

2.6 MetaGeek Wi-Spy 2.4 GHz Spectrum Analyser

A spectrum analyser takes measurements of signal strength across a radio frequency range. The Wi-Spy measures between 2400 MHz and 2483 MHz, which is the full range of 11 channels in Wi-Fi 802.11b/g/n networks. The Wi-Spy device has a form-factor of a small USB flash drive. Retailing at $99 USD [22], it is relatively inexpensive for the data that can be retrieved from it. It simply consists of a 2.4 GHz radio and low-speed USB controller. Another spectrum analyser on the market is the Cognio AirMagnet Spectrum Analyser which provides more advanced features (device recognition and interference signatures) and retails at $4000 USD [19].

In Figure 2.3, the type of output that can be received from the Wi-Spy is displayed using the included software package by MetaGeek called Chanalyzer; the frequency is reflected on the x-axis, and the amplitude (signal strength) on the y-axis. The figure shows the typical usage of channels in a busy area, with heavy usage on channel 6, and light occasional usage of channels 1 and 11. Notice how usage of channel 6 overlaps over multiple frequencies (utilising channels 4, 5 and 7, 8), when doing calculations with data received this will need to be taken into account. A single Wi-Fi channel overlaps 22 MHz.

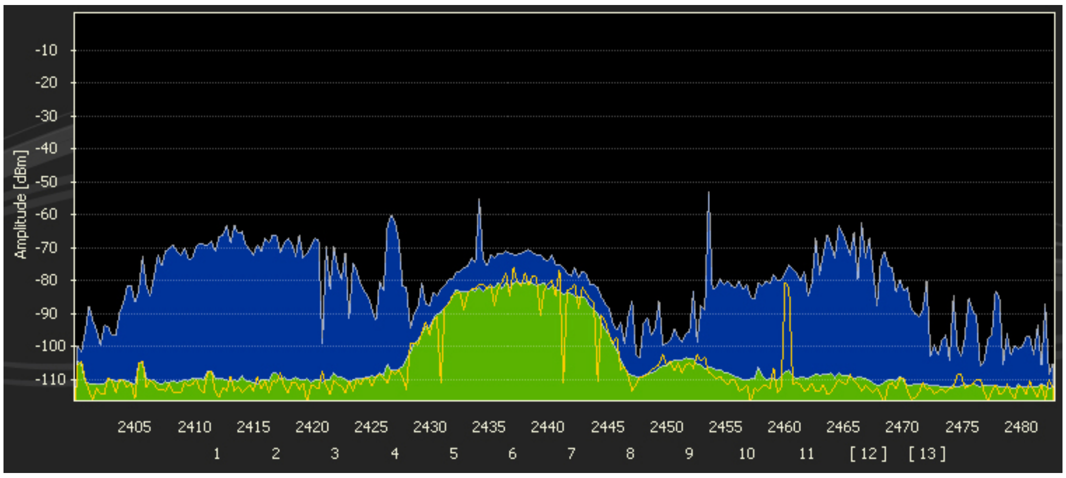


Figure 2.3: Sample Wi-Spy output using Chanalyzer

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
Frequency of X	RSSI of X	RSSI of X+1	RSSI of X+2	RSSI of X+3	RSSI of X+4	RSSI of X+5	RSSI of X+6

Table 2.1: Message received from Wi-Spy device after Get Feature Report

2.6.1 Technical Details/Hardware Interface

The Wi-Spy radio has a receive sensitivity of -90 dBm and has a top data transfer rate of 62.5 kbps (this equates to approximately five full sweeps per second). The device enumerates as a low-speed USB Human Interaction Device (HID), allowing multiple operating systems to use standard drivers. In order to receive data a Get Feature Report is sent and the device responds with an 8-byte feature report (see Table 2.1). The first byte is the current frequency position, and the following 7 bytes are the measured Receive Signal Strength Indication (RSSI) at that frequency and the next six frequencies in increments of 1 MHz.

Feature reports are sent sequentially and non-overlapping. The first report received will contain RSSI measurements for 2400-2406 MHz, and the next report will contain 2407-2413 MHz, and so on. When the report for 2477-2483 MHz is received, the hardware will return to 2400 MHz, and continue to loop as long as Get Feature Report's are received [20]. The device is constantly updating the information internally therefore when feature reports are received, it is always the most current data.

RSSI needs to be converted to dBm, when used in software, using the equation in Algorithm 4.

The dBm value for the frequency can be converted to give us an approximation of the distance that the signal has travelled.

Algorithm 4 Converting from RSSI to dBm

```
$dBm = RSSI * 1.5 - 97$
```

2.6.2 Usage of the Spectrum Analysis

Utilising multiple spectrum analysers to collect frequency data from different geographical locations, a scale grid can be drawn with the fixed points of the known locations of the listening nodes shown. Each node will retrieve RSSI data and calculate the distance the signal has travelled. This is drawn on the grid by circles with the radius of the calculated distance and the origin at the node's location. By combining this distance information and trilateration, the possible location of a device can be discovered.

A minimum of three listening nodes will be required for trilateration, yet more accuracy can be obtained with additional nodes. A margin of error will have to be catered for where no or only some circles intersect, this will decrease the accuracy of the trilateration but provide an area where the device may be. Signal strength can be influenced by interferences that were discussed in previous sections, and hence decrease accuracy of trilateration if they are present.

2.7 Chapter Summary and Conclusion

In this chapter, we discussed triangulation, trilateration and dBm units of measurement and suggested that by using trilateration and signal strength obtained from a spectrum analyser we can discover the location of devices in the Wi-Fi network. Thereby locating rogue AP's and helping secure the network. Security was discussed along with flaws discovered in the WEP protocol, which have led to many attacks and consequently we recommend using stronger protocols such as WPA or WPA2. Interferences that affect Wi-Fi adversely affect performance and need to be kept to a minimum. Site surveys aid in discovering interferences and efficient placement of AP's. Lastly, using three low cost devices, the Wi-Spy spectrum analyser, and the method of trilateration we can locate Wi-Fi devices. During the trilateration calculation, a margin of error needs to be incorporated due to interferences affecting signal propagation.

In the following chapter, the software design implemented in order to achieve the project goal is discussed.

Chapter 3

Design

3.1 Introduction

The design goal of this project is to develop a graphical representation of portions of Wi-Fi networks in action and determine the effectiveness in discovering the physical locations of Wi-Fi devices using spectrum analysers. This chapter describes the conceptual design of the solution, based on the evaluation of past research work and literature. It includes all application features, graphical user interface design and the underlying connections between multiple parts of the solution.

3.2 Conceptual Design and Feature Specification

The goal of the research project is to discover where Wi-Fi devices are physically located. A system was produced to meet this goal. The entire system has been named the WiSpy SSM Tool, SSM for Signal Source Mapping. The system has been developed in two parts (applications) with a webservice to connect them, see Figure 3.1 for the overall conceptual design. This design is a disconnected client server architecture, with two client applications, each with different requirements, and one server, each developed separately. The architecture provides advantages such as centralisation, scalability and flexibility.

The first part of the solution is the collecting client; it interfaces with the spectrum analyser and transmits this data to the webservice (the server). No limit exists on how many collecting clients can exist, as the more collecting clients present will achieve a higher accuracy when discovering the location of RF devices. The webservice receives this data and stores it in a local lightweight database. The compiling client sends requests (queries with specific requirements) to the webservice for data, and the webservice responds if it

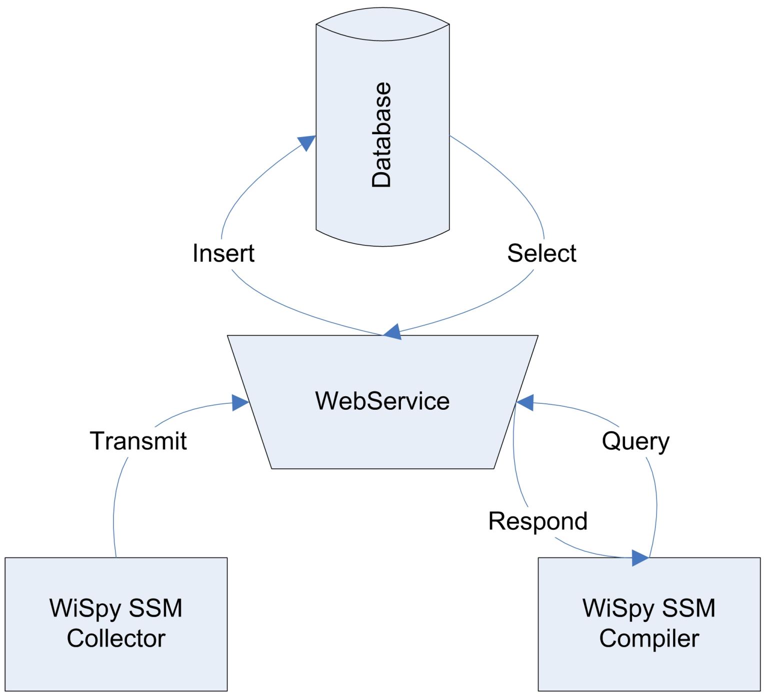


Figure 3.1: Conceptual Design of WiSpy SSM Tool

has data to match that specific query. Finally the compiling client compiles and sorts the data by time to graphically display the surrounding Wi-Fi signal for specific time periods.

These are the basic requirements and features of the overall design; each individual part is discussed in further detail in the sections below.

3.3 Collecting Client Design (WiSpy SSM Collector)

The main requirement of this application is to obtain the necessary signal strength data from the spectrum analyser and transmit it to the webservice (see Figure 3.1). Another important requirement is that the physical location where the signal strength is being read is transmitted with this data, and that the data is time sensitive and needs to have a related time field for each signal read. The collecting client is required to be able to transmit all this data in a single data type to the webservice to store. An additional, but not core, requirement is that the application can interface with a GPS device to continually update its location. A GPS device provides actual longitude and latitude coordinates, but is not necessary to the design; users can provide static location coordinates if the locations are known.

To transmit data, the computer needs to be connected to the same network as the webservice (or have an Internet connection). The application can also collect and store signal data offline and then transmit the data once reconnected, or the data can be stored as a file to be transmitted at a later stage.

This application is designed in such a way to minimise interference with normal computer usage and can be installed and set up to run unobtrusively to the normal user. This allows the application to be used in all areas of the network and run on all Microsoft Windows machines, without the user having knowledge of what the application is doing.

The application stores configuration settings which allow the program to be closed and restarted without any input needed.

The interface requirements are for a simple, clean and intuitive interface. The interface provides minimal feedback to the user by presenting a simple line graph displaying the total Wi-Fi spectrum and current signal strength data. With minimal input the application is running and transmitting data to the webservice.

A separate configuration window displays current settings, information about the spectrum analyser and multiple text boxes exist for information entry. Information includes the node name and location, the webservice address, how often sweeps are conducted and how many sweeps per webservice request are sent (granularity). A lower granularity would be required for a mobile listening node, as location information would be updated quicker, whereas a higher granularity would suit a fixed listening node. The configuration window also has information about the GPS device (whether it is connected or used); the GPS device can either be autodetected or the serial port and baud rate can be selected from a list manually.

See Figure 4.1 for a of the design that was implemented for the WiSpy SSM Collector.

3.4 Webservice Design

The webservice is the server in this client-server architecture and is required to handle requests and then respond to those requests. The architecture is described as disconnected, as the webservice is never connected to its clients; it just replies to requests. The webservice is required to be easily deployed on a network where the two clients can communicate with it. This serves as an interface to the signal database, allowing data to be inserted and data to be queried and sent. The webservice is required to provide a small set of methods to achieve this goal. These methods are to the tables in the database (password protected), get all the node names and associated record count, insert new signal data and return data from a node between specific time values.

3.4.1 Database Design

The database is required to hold the signal data, its associated node name and start time. In chapter 4, further implementation details make this requirement clearer.

3.5 Compiling Client Design (WiSpy SSM Compiler)

The compiling client is required to extract meaningful data from the webservice database and display this data to the user for analysis. The application is required to be able to select from which nodes the data is required and the specific time frame. The user may also select to open a previously stored recording. Once the data is collected it needs to be compiled and sorted by time to be displayed in chronological order. When data has been retrieved the user may select to save the recording, or they may proceed to view the data. The application takes the data received and compiles it into meaningful data that graphically displays a portion of a Wi-Fi network to the user.

As with the collecting client, the compiling client's user interface is required to be simple and intuitive. A configuration window is provided to modify the displayed channel colours, change the webservice address and reset the database.

See Figure 4.6 for a screenshot of the WiSpy SSM Compiler that was implemented.

3.5.1 User Interface Design

The application is required to have dynamic node location updates. The Wi-Fi channels (1-13) can be shown or hidden from view. The colours for the displayed channels can be modified in the configuration window. This application works with time sensitive data, which needs to be replayed to the user in the same order it was recorded. The user can either play the data at the same speed at which it was recorded or incrementally move through or skip through using a slider.

As an additional requirement to the design goal, this application provides translation, rotation and scaling transformations on the drawn data to aid in analysing the Wi-Fi network.

3.6 Chapter Summary and Conclusion

This chapter discussed the requirements and feature specifications for the three part solution, with the overall design being a disconnected client-server architecture. Chapter

4 goes into lower level and further detail regarding how these requirements were met by discussing the choices used in implementation.

Chapter 4

Implementation

4.1 Introduction

The previous chapter discussed requirements of the project solution, this chapter will delve into the implementation decisions to meet those design goals. The primary design goal of the WiSpy SSM Tool was to build a high performance tool suited to performing Wi-Fi network analysis and in locating Wi-Fi APs and interference sources. In this chapter we develop the three sections of the project, the two client applications and the webservice (server).

A working copy of these applications can be found on the accompanying CD (details in Appendix B), along with all esoteric information required in using the software.

4.2 Development Platform and Software Libraries

This project has utilised numerous components to provide the application features and requirements specified in chapter 3. This section motivates why specific components have been used to develop the solution and any alternatives that were investigated. Detailed here are the platforms used and the motivation for those choices. How the platforms were used is discussed in sections 4.3, 4.4 and 4.5.

4.2.1 Microsoft Visual Studio 2005, Microsoft .NET Framework 2.0 and C#.NET

Microsoft Visual C# is widely used and an enormous amount of support and documentation can be found on the Internet (an example is the Microsoft Developer Network [27]). Three components external to the .NET Framework have been utilised in this project;

firstly the HID API [3] that has been used to interface with the WiSpy USB HID device (section 4.3.1), secondly the GPSTools [16] component (section 4.2.5) to interface with GPS devices and lastly the SQLite [35] database interface (section 4.2.4). These three components and their features are explained in sections below.

The applications in this project have been developed using Microsoft Visual C#, using windows forms for their Graphical User Interfaces (GUI) and two dimensional graphics which utilise the System.Drawing.Drawing2D component in the .NET Framework. Both applications can use anti-aliasing to remove sharp edges from the graphs displayed, but this is optional to the user, as anti-aliasing can reduce the frame rate of the animation. Various other Microsoft .NET Framework components have been used and their use is explained in later sections in more detail.

4.2.2 ASP.NET Webservice

Creating a server-client architecture is made simple by using an ASP.NET Webservice [26] and is easy to integrate into any existing project. Designing a custom server which accepts multiple connections using TCP socket connections [29] is complex and not within the scope of this project therefore a webservice was chosen.

A webservice essentially accepts requests and responds to these requests, it will not start the communication. It contains WebMethods (publicly accessible methods to anyone who can access the webservice) and private methods which can only be called from within the webservice.

The webservice is instantiated as a Service object in Visual C#, and it's webmethods are called like public methods with parameters and return types. The data types can be any complex or primitive. Requests and response to and from the webservice are sent as Simple Object Access Protocol (SOAP) messages [14], these messages are generated by Visual C# when the webmethods are called. Due to these messages being sent in XML, all data types sent and received have to be serializable[1].

Microsoft Visual Studio provides an ASP.NET Development Server for the webservice to run on while in the development phase. When development and testing are complete, the webservice is deployed on Microsoft Internet Information Service (IIS). The development server runs on the local machine and is inaccessible to other users on the network, hence the need to deploy to IIS.

[1]Two dimensional arrays are not serializable, but jagged arrays (arrays of arrays) are

4.2.3 Deploying to Microsoft Internet Information Service (IIS)

Once the ASP.NET webservice was completed, it was deployed to Microsoft IIS [24] to be visible to all network users. The webservice is published using Microsoft Visual Studio to a virtual directory mapped by IIS, to make it visible to web users. The 'ASPNET' user has to be given write access to the SQLite database file. Finally ASP.NET has to be installed for IIS by running '*aspnet_regiis.exe -i*' in a command prompt in the .NET framework directory.

4.2.4 SQLite Lightweight Database

SQLite [35] was chosen as it is a lightweight database which is perfectly suited for a webservice where minimal amounts of space are available. It "is a small C library that implements a self-contained, embeddable, zero-configuration SQL database engine" [35]. This database has a small code footprint (less that 250 KB) as it includes a subset of features compared to other database implementations. SQLite has support for five data types; NULL, INTEGER, REAL, TEXT and BLOB. In the WiSpy SSM webservice database the TEXT and REAL data types were used.

Transactions are atomic, consistent, isolated and durable (ACID). The database is stored in a single disk file, it has a simple and easy to use API, is self contained and the source code is dedicated to the public domain.

The Finisar.SQLite [15] Data Provider is an easy way to use the SQLite database in Visual C# and has been used in this project.

4.2.5 Franson GPSGate and GPSTools

Two Franson Technology software releases, GPSGate and GPSToolsNET, were utilised to interface to the GPS device. Both applications are not free products but come with a 14 day free trial. Alternatives exist to these solutions, but it was found that these were the easiest to implement into this project, as they have Microsoft .NET assemblies and the trial period provided gave a sufficient time for application testing.

Franson GPSGate [17] is a background application which accepts GPS input and converts them into multiple outputs. Common inputs can be from a GPS device connected via USB or serially, GPS data sent over UDP or TCP/IP, or from a pre-recorded log file. The pre-recorded input was most useful during implementation development as GPS signal strength indoors is generally very low. GPSGate converts the input into a range of user-selected outputs allowing multiple GPS dependent applications to utilise the same

GPS device. The outputs range from starting a TCP/IP or UDP server to setting up virtual COM ports or logging the data to a file. This application can work with any GPS device that outputs with NMEA GPS data or any Garmin GPS device. This application is necessary as directly connecting the GPS device through a USB connection does not provide a serial port through which we can obtain data. GPSGate does not only work with GPS devices but can also split COM ports to many virtual COM ports.

Franson GPSToolsNET SDK [16] provides a series of .NET assemblies to interface with GPS devices through multiple inputs, convert position data between different coordinate systems and map drawing facilities. GPSGate provides an output called 'GPSGate Direct' and by using this output in conjunction with GPSTools allows the GPS device to be autodetected and also obtain information from it. Although any serial NMEA output can be utilised by GPSTools.

The SDK released provides many sample applications to aid in the development process. Using these assemblies is as simple as including a reference in the Visual C# project and making sure you have the latest Microsoft Visual C++ Redistributable Package [25] installed, as the assemblies have been written in Visual C++. The redistributable package allows applications developed with Visual C++ to run on a computer that does not have Visual C++ installed.

4.3 WiSpy SSM Collector - Collecting Client

This application interfaces via an HID API (see section 4.3.1) to the WiSpy spectrum analyser and transmits the collected signal data (and related time, location and node information) to the webservice to be stored in the database. This application provides feedback to the user with a simple line graph to show current signal data. The signal data is not modified, it is serialized into SOAP messages and transmitted to the webservice or it is serialized to XML and stored locally to be transmitted at a later time. This application has the option of obtaining exact location information from a GPS device.

Core to this application is the interface to the WiSpy device, section 2.6.1 explained what format the data is received in and section 4.3.1 goes into further detail on how this is accomplished using Visual C# HID code to enumerate feature reports from the device. When the 'Collect' button (see Figure 4.1) is pushed the application first checks to see if a WiSpy device is connected then requests feature reports from it continuously until the user either pauses or closes the application. Each feature report returns 8 bytes and stores these sequentially until a full sweep has been collected, this sweep is temporarily stored for processing and the next sweep begins.

As each sweep is collected, a time stamp is generated and accompanies that sweep. The granularity set in the Configuration window (see section 4.3.2.2) defines how many sweeps are collected per location update, it also defines the number of sweeps that are stored in each *SingleChunk* (see Table 4.2 for the definition). Depending on user settings, as each *SingleChunk* becomes full it is transmitted to the webservice or stored to file.

If no Internet or network connection to the webservice is present and a WiSpy is connected, the application can begin collecting signal data immediately and store collected data to a file. This file can then, later, be sent to the webservice when the Internet or network connection is re-established. This functionality is particularly important if a mobile listening node (a laptop with a WiSpy and this application) is used, as Wi-Fi connectivity onboard will interfere with collected signal data.

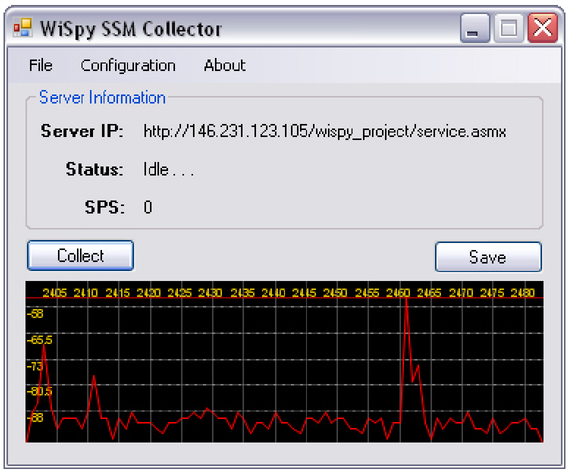
4.3.1 USB HID Code

The utilised code [3] provides an API to interface with any HID device. The WiSpy spectrum analyser is an HID device and by using its unique Vendor ID (VID) and Product ID (PID) we can gain control of the device using this HID API. Data is received from the device by requesting feature reports, which returns 8 bytes at a time (see Table 2.1 for a representation of these reports). The HID API has been modified and specialised for this project and the WiSpy spectrum analyser.

Most importantly to this project are the following methods within the utilised HID API: *FindTheHid()* attempts to connect to the device using the VID and PID, it retrieves the information of all connected HID devices and searches the list for the device we want. It returns true if our device is found. *OnDeviceChange()* is triggered when any device is attached or detached, it then tests if the device change is the device that is currently being communicated with. *ExchangeFeatureReports()* writes a blank feature report to the device and then reads a feature report which contains the signal data in bytes.

4.3.2 Simple User Interface

The GUI has been designed to be simple, clean and intuitive to the user, and the application has been implemented to follow this design. See Figure 4.1 for a screenshot of WiSpy SSM Collector running. Upon application start up, the user is presented with minimal menus, a line graph and two buttons. One button to begin displaying data on the line graph and the other to store the displayed data. The GUI is intuitive as the 'Save' button will only be enabled once the 'Collect' button has been selected.



Server IP: http://146.231.123.105/wispy_project/service.asmx

Status: Idle . . .

SPS: 0

Figure 4.1: WiSpy SSM Collector Screenshot

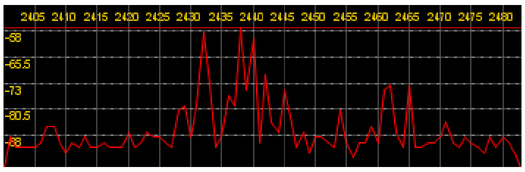


Figure 4.2: Line Graph in WiSpy SSM Collector

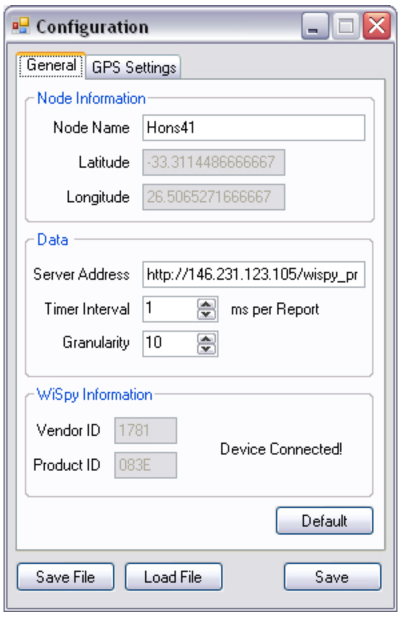
4.3.2.1 Line Graph

The line graph is shown to display the signal spectrum activity to the user and also provide feedback that the WiSpy device is working. A full signal sweep consists of 84 MHz of signal data, with a reading at 1 MHz intervals. The graph is produced by enumerating through the signal readings and drawing a line between the current point and the next point until the last point has been reached. The maximum signal read for that sweep is displayed as a horizontal line at the maximum signal strength position.

Each axis is marked, the x-axis with frequency (in MHz) and the y-axis with signal strength (in dBm). See Figure 4.2 for an example of this line graph. Analysing Figure 4.2 informs you that it displays a Wi-Fi Channel 6 download (amplitude peaks at 2437 MHz, the centre of Wi-Fi channel 6). More analysis of the WiSpy SSM Collector output is provided in section 5.3.1.

4.3.2.2 Options, Configuration and GPS settings

The menus of the WiSpy SSM Collector consist of File, Configuration and About (see Figure 4.1). The 'File' menu allows the user to select 'New File', this sets up the application to store the collected data to memory instead of transmitting it immediately



Node Information
Node Name: Hons41
Latitude: -33.3114486666667
Longitude: 26.5065271666667

Data
Server Address: http://146.231.123.105/wispy_pr
Timer Interval: 1 ms per Report
Granularity: 10

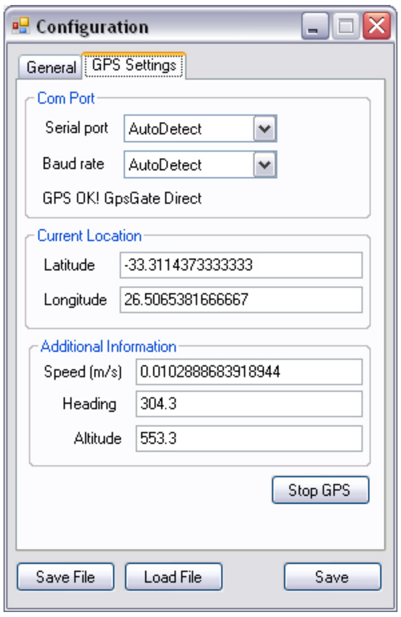
WiSpy Information
Vendor ID: 1781
Product ID: 083E
Device Connected!

Figure 4.3: Configuration Window (General Tab)

upon collection. 'Save' saves the signal data in memory to file, and 'Close File' stops the application from storing data and reverts to transmitting directly to the webservice. 'Transmit File' allows the user to browse for a pre-recorded file and transmit it to the webservice.

Selecting 'Configuration' on the menu brings up the configuration window. In the 'General' tab (see Figure 4.3) the user can change the name of the node, address (URL) of the webservice, how often reports are requested from the WiSpy device (in ms) and the granularity of the *SingleChunk*s. The location can manually be set and GPS coordinates are requested in DD.DDDDD° format. Device information is displayed for the WiSpy spectrum analyser, most importantly whether it is attached or not to the computer running the WiSpy SSM Collector application. The settings contained here can be saved and loaded from file by serializing the configuration to XML, allowing settings to be pre-created for nodes and easily disseminated ensuring that all nodes are transmitting their data in a similar format.

The 'GPS Settings' tab (see Figure 4.4) in the configuration window contains connection settings to the GPS. Franson GPSTools automatically detects settings for the GPS, but the user can manually edit these settings if they so wish. Once the 'Use GPS' button is selected, GPSTools attempts to discover the device and begin communicating location information. The latitude and longitude of the node are updated automatically via the use of a timer and event handlers. Additional GPS information is displayed to the user; current speed (in m/s), heading and altitude.



Com Port

Serial port: AutoDetect
Baud rate: AutoDetect
GPS OK! GpsGate Direct

Current Location

Latitude: -33.3114373333333
Longitude: 26.5065381666667

Additional Information

Speed (m/s): 0.0102888683918944
Heading: 304.3
Altitude: 553.3

Figure 4.4: Configuration Window (GPS Settings Tab)

The configuration settings and GPS interface are implemented using Singleton patterns [31] in order for the information to be available throughout the application.

4.4 Webservice

The webservice is the server in this client-server architecture. It receives SOAP messages and responds to them. The collecting client sends its signal data via SOAP messages to the webservice and the data gets stored in the SQLite database. The compiling client sends its queries via SOAP messages which are regenerated into SQL queries to run on the SQLite database. The selected data is then returned to the compiling client via SOAP.

4.4.1 WebMethods

Four WebMethods implemented on the webservice provide the interface for the collector and compiler to transmit and receive data. Methods to initialise the database (discussed further in section 4.4.1.1), store data, get all the names of the collecting nodes and retrieve data for a specific time period are provided. The methods which get or insert data take parameters which are included in the SQL queries generated.

Algorithm 5 MD5 hash password checking

```
string thePassword = "0D4958F6165341A88685FA0E9ACA3A7D";
                      //MD5 Hash of the password

[WebMethod]
public bool initDatabase(string password)
{
  //obtain hash of received password to compare with existing
  byte[] hash
       = new System.Security.Cryptography
          .MD5CryptoServiceProvider()
          .ComputeHash(ASCIIEncoding.ASCII.GetBytes(password));

  if (ByteArrayToString(hash).CompareTo(thePassword) == 0)
  {  //password match
     //continue with database initialisation SQL
     . . .
     return true;
  }
  return false;  //password mismatch
}
```

4.4.1.1 Protection Against Database Initialisation

As a webservice is publicly accessible to the network, the database initialisation method had to be protected against an accidental reset or intentional tampering. The password is saved as an MD5 hash of the string password. The initialise method takes one parameter, the user's password, which is hashed and then compared to the hash of the stored password. If the hashes are same then the method continues initialising the database (or resetting it if it already exists). See Algorithm 5 for a portion of this code implementation in the Webservice, the method returns true if the passwords match.

4.4.2 Database Implementation

The Finisar.SQLite assemblies are referenced in the webservice and are used to control our database. The database is stored as a single file (*signalDB.db*). The webservice database is a holding area and only has one table for storing signal data. Only one table is necessary, as the data is received as a complex data type (*SingleChunk*, see section 4.4.2.1 and Table 4.2) from the collecting client. For each record in the table, there exists the node name (collecting client who sent the data), the earliest DateTime in the *SingleChunk*, and then

signal	TABLE
nodeName	TEXT
startTime	REAL
data	TEXT

Table 4.1: Table 'signal' in SQLite Database

SingleChunk	class
nodeName	string
locX	double
locY	double
data	byte[][]
times	DateTime[]

Table 4.2: SingleChunk Class

the *SingleChunk* (see Table 4.1).

As SQLite only supports a small amount of data types, certain transformations had to occur on the data before it could be stored. The node name needed no data type conversion as it is already a string and can be stored as TEXT. The time, was stored as a DateTime in the *SingleChunk* but needed to be stored as an accurate measure, and therefore was converted to Ticks[2]. The *SingleChunk* data also needed to be transformed before it could be inserted, because in a previous attempt to store it in a BLOB field, Finisar.SQLite called the object's .ToString() method, of which there is none and therefore failed. As a result the *SingleChunk* was converted to a byte array then base 64 encoded to transform it in order to store in the TEXT field.

4.4.2.1 Data Storage

All signal data that is received or transmitted from the webservice is of the *SingleChunk* type (see Table 4.2). This object contains public attributes and has no constructor or methods. This data type is designed this way to be serializable. For each *SingleChunk*, numerous signal sweeps are stored in a jagged byte array called *data.* The *data* array holds numerous arrays of byte arrays. In addition, the *SingleChunk* object also stores the associated times for when the data was collected. Each object only stores the name of the node and its location once per *SingleChunk*, but this granularity can be modified in the WiSpy SSM Collector configuration window.

[2]The value of this property represents the number of 100-nanosecond intervals that have elapsed since 12:00:00 midnight, January 1, 0001.

4.5 Wispy SSM Compiler - Compiling Client

Once the signal data has been collected by the WiSpy SSM Collector and stored in the database via the webservice, it needs to be processed and displayed in order to discover the location of 2.4 GHz RF devices.

When the WiSpy SSM Compiler is started (see Figure 4.5), the user is shown a form with only certain options (commands enabled) available to them. If this is the first time the application is started, the user will need to open the Configuration window (see Figure 4.7) to insert the server address of the webservice. By double clicking on a colour for any channel in the configuration window, a colour dialog is shown (see Figure 4.8) which allows the user to select whichever colour they desire for each channel. The configuration window also allows the user to reset the database provided they have the correct password and they can also choose whether they would like the display to be anti-aliased or not. The scale of the display can be modified by setting the initial latitude, longitude and the width of the display. Once the configuration has been completed, the user can begin to use the application.

The user has two options to begin viewing data; by selecting File->Open Recording, a previous recording can be opened and displayed, or the user may select the 'Refresh' button (see Figure 4.6, bottom left) to obtain a list of stored nodes from the webservice and begin structuring a query for data from them.

4.5.1 Querying the Database

Once a list of existing nodes in the database has been received from the webservice, the user can structure a query using the query builder (see Figure 4.6, bottom middle). The user can select the time frame from which data is required and either select a specific node from the node list or obtain data for all nodes.

By providing a query builder, the application can quickly download specific data for a particular time frame, making analysis of the data more efficient as the user doesn't have to wait extended periods of time before displaying the data. The time to download signal data depends on the amount of data being received and how fast the network connection is to the webservice. The most efficient location to run the compiler application is on the same machine the webservice is served from. After downloading the specific data, it needs to be sorted before it can be displayed.

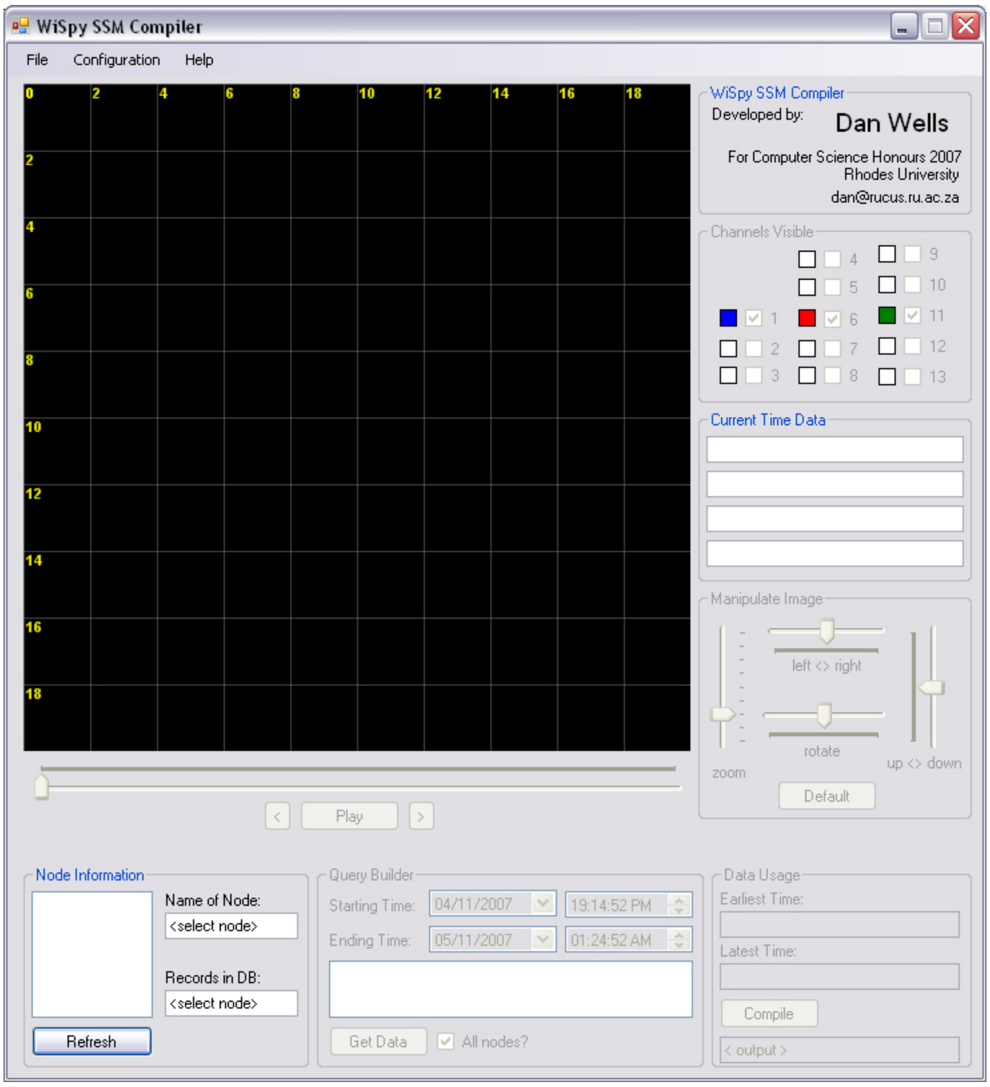
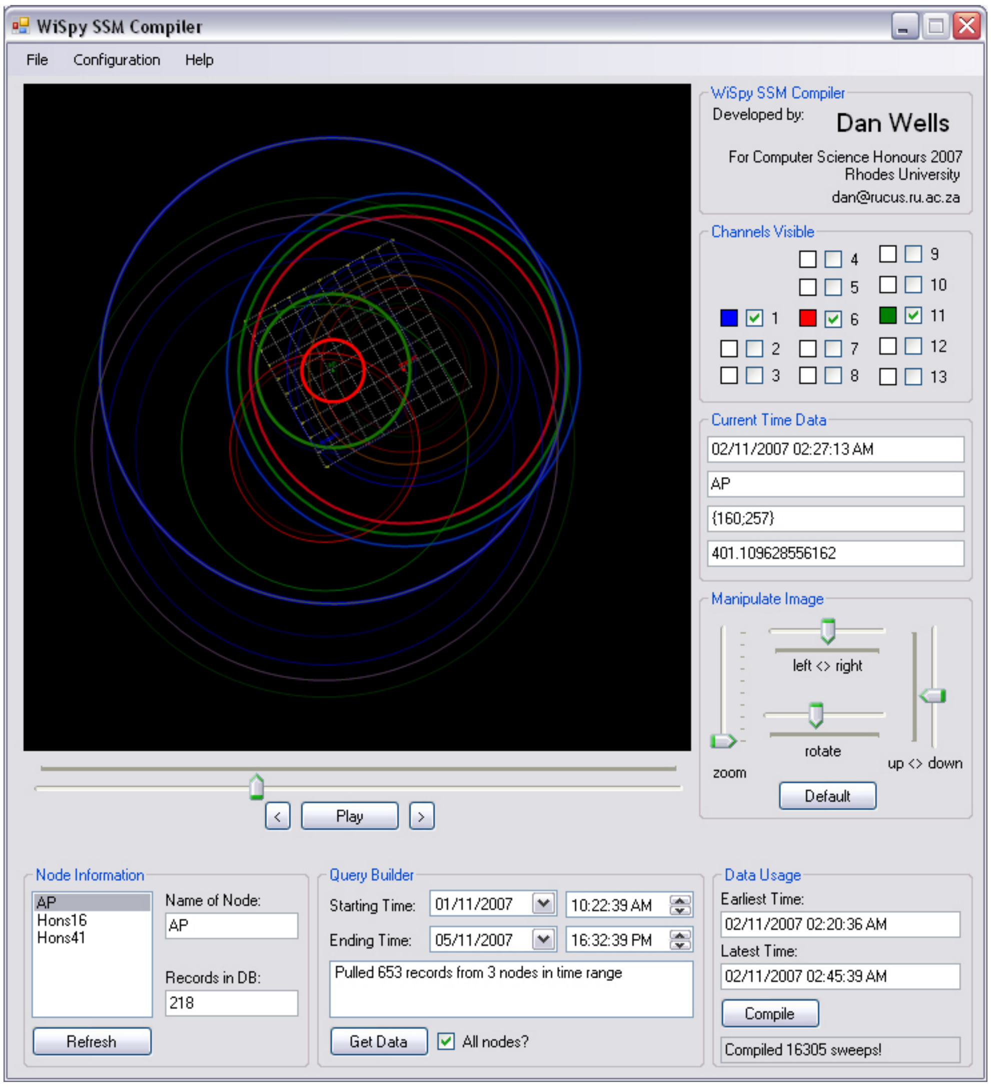


Figure 4.5: WiSpy SSM Compiler Screenshot (on start up)



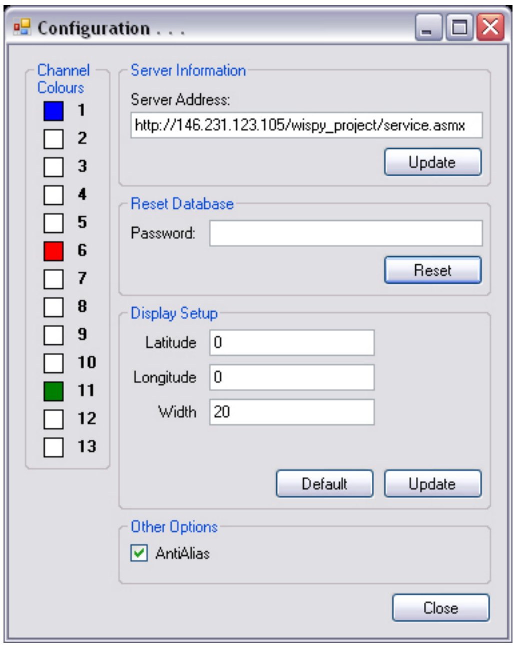
Current Time Data: 02/11/2007 02:27:13 AM; AP; {160;257}; 401.109628556162

Node Information: AP, Hons16, Hons41
Name of Node: AP
Records in DB: 218

Starting Time: 01/11/2007 10:22:39 AM
Ending Time: 05/11/2007 16:32:39 PM
Pulled 653 records from 3 nodes in time range
All nodes? ✓

Earliest Time: 02/11/2007 02:20:36 AM
Latest Time: 02/11/2007 02:45:39 AM
Compiled 16305 sweeps!

Figure 4.6: WiSpy SSM Compiler Screenshot (running)



Server Address: http://146.231.123.105/wispy_project/service.asmx

Password:

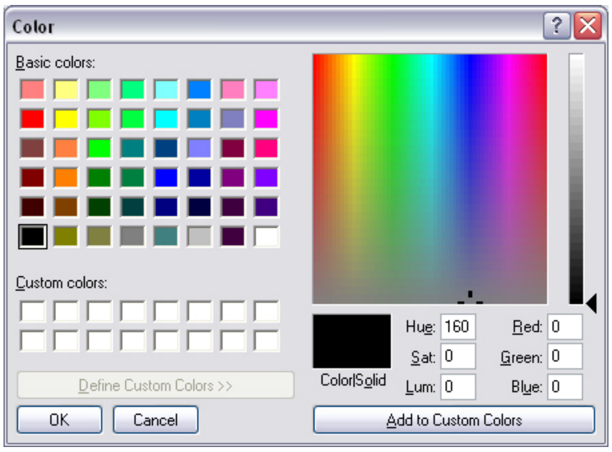
Latitude 0

Longitude 0

Width 20

[✓] AntiAlias

Figure 4.7: Configuration Window for WiSpy SSM Compiler



Hue: 160 Red: 0

Sat: 0 Green: 0

Lum: 0 Blue: 0

Figure 4.8: Standard .NET Colour Dialog Component

Current Data	data type
workingTime	DateTime[]
workingNode	string[]
workingPoint	PointD[]
workingData	byte[]

Table 4.3: Working Data

PointD	class
x	double
y	double

Table 4.4: PointD Class

4.5.2 Sorting the Received Signal Data

As the data is stored in the database in no particular order, after it has been downloaded it needs to be sorted in chronological order. The user clicks the 'Compile' button (see Figure 4.6, bottom right) to sort the data before being displayed. The data is received as a jagged array of type *SingleChunk*, which is deconstructed into four arrays (see Table 4.3) to be sorted synchronised to time. The sort used is a recursive quick sort, modified so that when it swaps two working time values, it swaps the other three working values (node, point and data) as well. This provides the application with a sorted data set to display.

In Table 4.3 *workingPoint* is of the data type PointD (see Table 4.4 for the definition), as the location data is received as a double. When displaying the data on the screen, Visual C# provides the Point (integer) and PointF (float) data types. As a result of a possible loss of precision while processing the data, the PointD class was created.

4.5.3 Rendering and Timing

So far, the data has been downloaded according to a specific query and compiled and sorted into chronological order. The signal data now needs to be displayed on screen for the user to analyse. The application graphically displays all the data in a .NET PictureBox component as Graphics from a Bitmap image. The Graphics can be anti-aliased as well (done by default) depending on whether the user has selected this option in the configuration window or not.

Graphical transformations occur as the track bars in the 'Manipulate Image' group box are manipulated (see Figure 4.6, right hand side and middle of the application). Transformations allow the user to view the display from different positions. Scale transformation

Algorithm 6 Formula for Distance Signal is Transmitted

$$\frac{P_{rx}}{P_{tx}} = \frac{G_{tx} \times G_{rx} \times c^2}{(4 \times \Pi \times d \times f)^2}$$

allow the display to be seen closer or from further away. The rotation transformation allows the display to be rotated 180 degrees from the centre. Translation transformations move the display up or down and left or right.

The grid lines and axis labels are displayed, the nodes are displayed, and the sweeps are displayed. The grid is a standard 10 x 10 grid and its axis labels are displayed according to the latitude, longitude and width values set in the configuration window.

4.5.3.1 Signal Data Representation

The data received is drawn to screen using circles for each channel that originates from the node location. The larger the circle the further the signal is transmitted, and the smaller the circle the closer the signal is transmitted from. The circles provide a representation of how close a particular signal has been transmitted to the WiSpy spectrum analyser. As each circle is drawn it is the brightest and widest, and the previous 9 circles fade out slowly. The formula used to calculate the distance is in shown Algorithm 6.

The symbols used in the signal equation are as follows: P_{rx} is the received power (in watts). P_{tx} is the transmitted power (in watts). G_{tx} is the gain of the transmitting antenna. G_{rx} is the gain of the receiving antenna. c is the speed of light (3×10^8). pi is approximately 3.14159. d is the distance (in meters). f is the frequency (in Hz).

Algorithm 6 is for the ideal line-of-sight situation, which almost never holds in a real world environment. In reality the antenna gains will be hard to quantify and multipath propagation will have unpredictable effects.

Data received from the WiSpy spectrum analyser is in RSSI, which needs to be converted into dBm (see Algorithm 4) and then into watts. The RSSI converted to watts provides the received power. The transmitting power has been estimated at 18 dBm (63mW). The APs used in evaluation transmit at this power $\pm$2dB. The receiving and transmitting antenna gain is estimated at 1 dB, the AP antenna's used have this gain and the WiSpy device as well. Finally frequency differs depending on which channel is currently under observation, each channel exists between 2400 MHz and 2483 MHz (see Table 4.5).

Wi-Fi Channel	Start Frequency	Centre Frequency	End Frequency
1	2401	2412	2423
2	2406	2417	2428
3	2411	2422	2433
4	2416	2427	2438
5	2421	2432	2443
6	2426	2437	2448
7	2431	2442	2453
8	2436	2447	2458
9	2441	2452	2463
10	2446	2457	2468
11	2451	2462	2473
12	2456	2467	2478
13	2461	2472	2483

Table 4.5: Wireless 802.11b/g Channel Frequencies (in MHz)

4.5.3.2 Animation Timing

When the user clicks the Play button the data is played in order and in real time. The 'Play' button is situated at the bottom of the display, see Figure 4.6. A timer increments the value of the trackbar situated underneath the display, which has an event handler to redraw the display when its value changes. The user may slide the track bar to quickly move through the current working data, or the user may use the '<' and '>' buttons to increment and decrement one value at a time for closer investigation.

4.6 Chapter Summary and Conclusion

This chapter implemented the design discussed in chapter 3. The WiSpy SSM Tool was written in Microsoft Visual C# and the webservice utilised ASP.NET. The WiSpy SSM Collector interfaces with the WiSpy spectrum analyser and transmits its signal data to the webservice, which stores the data in the SQLite lightweight database. The WiSpy SSM Compiler retrieves specific signal data from the database through the ASP.NET webservice via queries and displays the compiled data in chronological order to the user to analyse.

Chapter 5 evaluates the solution and provides results to confirm the goals of the project.

Chapter 5

Evaluation and Results

5.1 Introduction

The previous chapters discussed how the WiSpy SSM Tool was implemented. In this chapter the tool is evaluated to determine its effectiveness. Results from both applications (WiSpy SSM Collecter and WiSpy SSM Compiler) are discussed.

5.2 Experiment Setup

This section is to provide an overview of how the experiments were conducted. Firstly which devices were used in the evaluation, secondly how these devices were laid out to run the experiments and lastly what environmental effects we expected.

5.2.1 Specific Hardware Used in Experiments

Many devices were used to gather data for our experimentation. Firstly we had the WiSpy devices, wireless APs, wireless PCI cards, a laptop, 3 PCs and finally a GPS device:

- 3x MetaGeek WiSpy 2.4GHz USB Spectrum Analyser V1.0
- LinkSys Broadband Wireless-G Router, WRT54G (RF Power Output: 18 dBm)
- D-Link AirPlus Xtreme G 2.4GHz Wireless Access Point, DWL-2000AP (RF Power Output: 15 dBm±2dB)
- D-Link AirPlus Xtreme G 2.4 GHz Wireless 108G Access Point, DWL-2100AP (RF Power Output: 15 dBm±2dB)

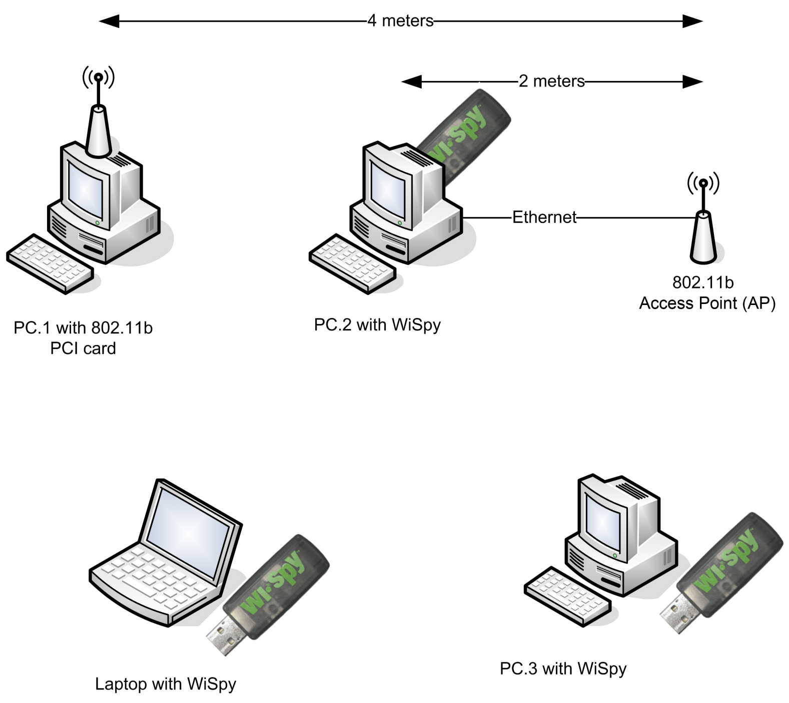


Figure 5.1: Layout of Experiments

- D-Link DWL-520 Wireless 802.11b PCI Adapter
- D-Link DWL-520+ Wireless 802.11b PCI Adapter
- Acer TravelMate 4154LMi with onboard 802.11b/g wireless LAN
- 3x Desktop PCs - Intel Core 2 Duo CPU with 2 GB of RAM (2x Microsoft Windows XP Service Pack 2, 1x Microsoft Windows Vista)
- Garmin etrex Vista C GPS

5.2.2 Layout of Experiments

The basic layout of all experimentation is shown in Figure 5.1 (the test setup). We have installed the 802.11b wireless PCI card into PC.1 and connected PC.2 via wired Ethernet directly to the AP. PC.1 and PC.2 were placed two meters apart with the AP two meters away from PC.2. PC.3 and the laptop had the WiSpy SSM Collector application and a WiSpy spectrum analyser. In these experiments the AP had been configured to only allow the MAC addresses of the laptop and the 802.11b wireless PCI card. MAC address filtering was used for security reasons (section 2.3) and to prevent interference with the experimentation. PC.1 and PC.2 allow Remote Desktop connections and can be managed from the laptop. The AP, PCs and laptop were all on the same subnet (192.168.0.0/24). The laptop's Wi-Fi connection was disabled during evaluation to prevent its transmissions affecting the result set.

Experiments were conducted by downloading files from PC.2 to the PC.1. By downloading files this way it ensured the AP was transmitting most of the time and not PC.1 with the 802.11b wireless PCI card (i.e. in the direction of the so called "down link"). Most experiments were run for short (20 minutes) periods but the occasional experiment ran for longer (three days). All results collected and shown were from nodes at fixed locations, although evaluation with GPS dynamic location updates were conducted and were mostly successful they are not shown here.

Firstly we evaluated the WiSpy SSM Collector for an accurate reading from the spectrum analyser (section 5.3.1). Once we were satisfied with the outcome of this evaluation we obtained results from the WiSpy SSM Compiler (section 5.3.2) utilising the information gathered for the WiSpy SSM Collectors.

5.2.3 Environmental Considerations

These experiments do not take into account differing environmental factors. The formula the application uses to estimate the distance the signal has been transmitted does not take into account the resulting overall path loss by obstructions present in the Wi-Fi network.

5.3 Performance of Experiments

This section describes some of the findings of the tools' evaluation. Firstly concentrating on the WiSpy SSM Collector, then on the WiSpy SSM Compiler.

5.3.1 WiSpy SSM Collector

The WiSpy SSM Collector was evaluated to ascertain whether the data passed onto the webservice database was correct or not. All the data collected for this portion of the evaluation was from a single WiSpy SSM Collector application 5 meters away from the test setup. Three tests were conducted, each running on different channels and at different times but with the same AP.

The first test was a regular file download on Wi-Fi channel 1 on the test setup. Figures 5.2 and 5.3 show higher activity between 2401 MHz and 2422 MHz, giving us a width of activity just short of 22 MHz and the mid-point around 2411 MHz. If we compare the activity data seen to the frequencies in Table 4.5 we can confirm that our spectrum

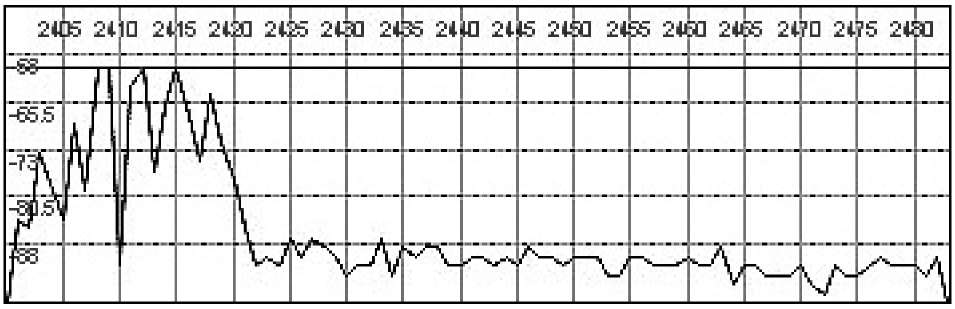


Figure 5.2: WiSpy SSM Collector - Channel 1 Download - Result 1

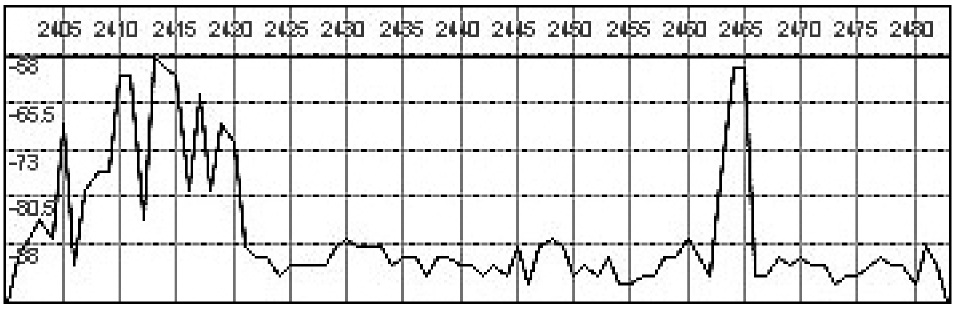


Figure 5.3: WiSpy SSM Collector - Channel 1 Download - Result 2

analyser is giving us accurate data. Figure 5.3 shows an interference spike around 2465 MHz which could have been caused by perhaps a Bluetooth device in the area.

This test was repeated for channels 6 and 11 under the exact conditions of the first test and similar results were achieved. Figures 5.4, 5.5 show the evaluation for channel 6 and Figures 5.6, 5.7 show the evaluation of channel 11.

Another test was conducted with the laptop and the WiSpy SSM collector during a file download on the test setup. Starting near the test setup with a high amplitude displayed for channel 11, we slowly walked away from the AP. As we moved away from the transmitting AP, the signal strength read for channel 11 dropped. The reduced signal amplitude was to be expected because as the spectrum analysers is moved further away from the transmitted RF signal, the signal would have to travel further and therefore would incur more free space loss. As we moved out of range of the transmitted signal, no activity was shown for channel 11.

From our evaluation we confirmed that correct data was being read from the WiSpy spectrum analyser and interpreted correctly. We could now continue our evaluation with the WiSpy SSM Compiler.

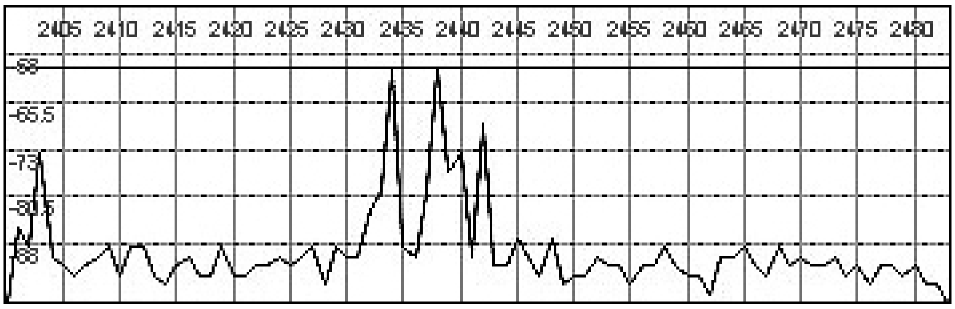


Figure 5.4: WiSpy SSM Collector - Channel 6 Download - Result 1

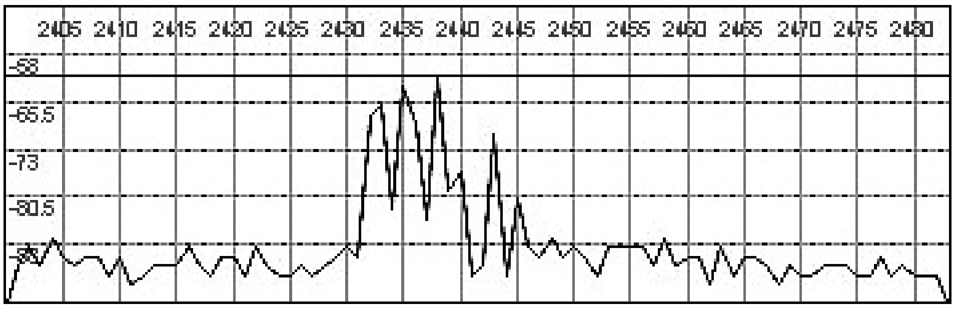


Figure 5.5: WiSpy SSM Collector - Channel 6 Download - Result 2

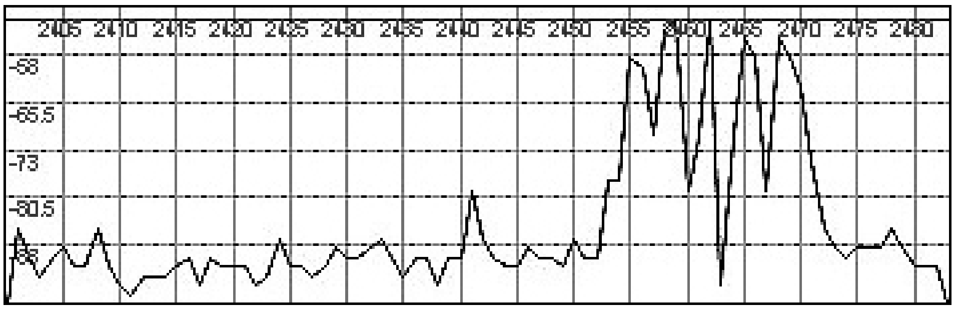


Figure 5.6: WiSpy SSM Collector - Channel 11 Download - Result 1

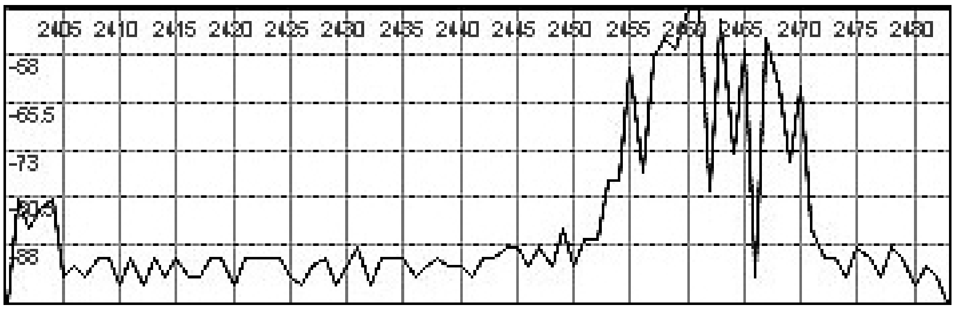
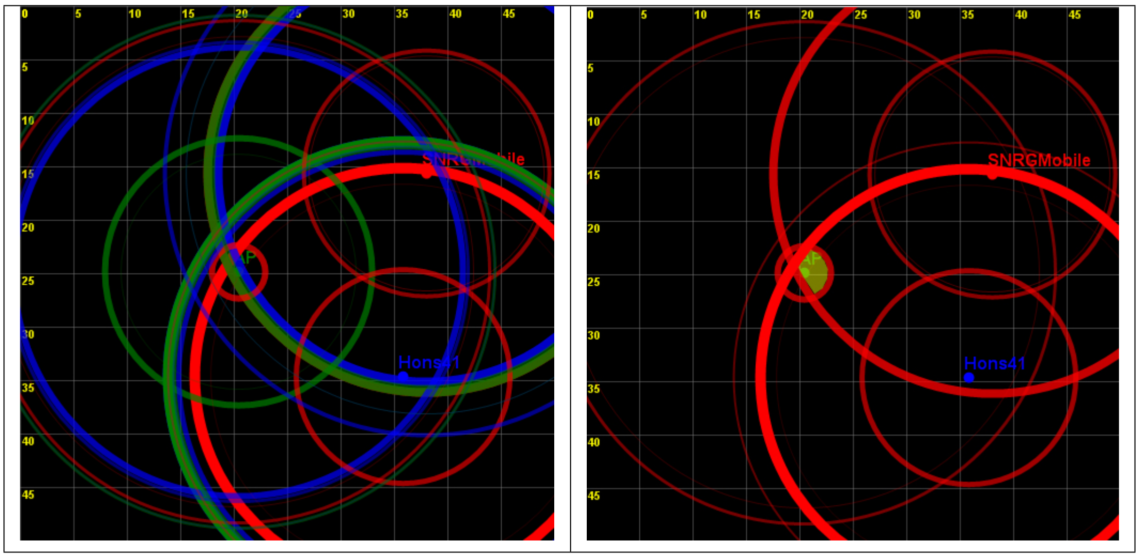


Figure 5.7: WiSpy SSM Collector - Channel 11 Download - Result 2



Result 1A - Cluttered	Result 1B - Only Channel 6 Shown

Figure 5.8: WiSpy SSM Compiler - Result Set 1

5.3.2 WiSpy SSM Compiler

We have evaluated the WiSpy SSM Collector and concluded that the data received is correct and meaningful to process further. Many hours of signal data have been collected and some of the most interesting results will be discussed.

In the results shown (Figures 5.8-5.11) the test setup is situated at the node named 'AP' and intermittent small file transfers are being downloaded to PC.1. Intermittent file transfers was chosen over large file downloads as we wanted to mimic real world Wi-Fi usage. The scale used in all the results is meters.

Screenshots from the application are explained and analysed.

5.3.2.1 Result Set 1

In Figure 5.8 Result 1A displays a typical WiSpy SSM Compiler output which is showing the most commonly used channels; 1, 6 and 11. The display is cluttered with overlapping colours and circles. But after quickly scrolling through the data and analysing it, the user can make a decision regarding which channel they want to have a closer look at. In Figure 5.8 the results marked 1B are the same as those marked 1A except that 1B only shows the values for channel 6, while 1A shows channels 1, 6 and 11.

In Figure 5.8 Result 1B shows three listening nodes and their output for channel 6. The brightest circles show the last signal data to be displayed, and these three circles intersect (highlighted in yellow) within approximately 2 meters of the AP. This result is

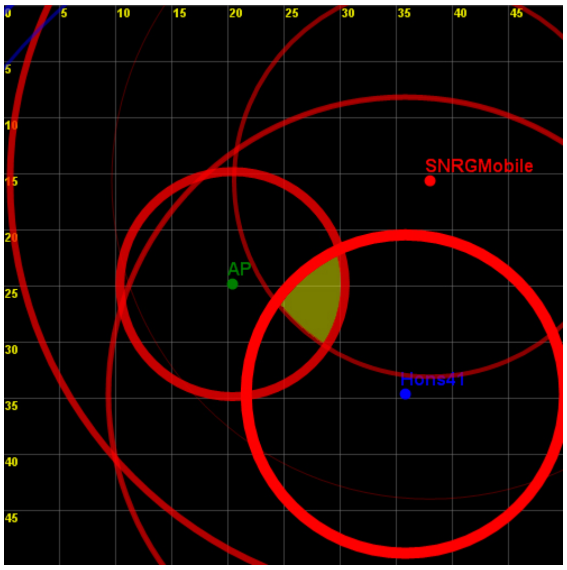


Figure 5.9: WiSpy SSM Compiler - Result 2

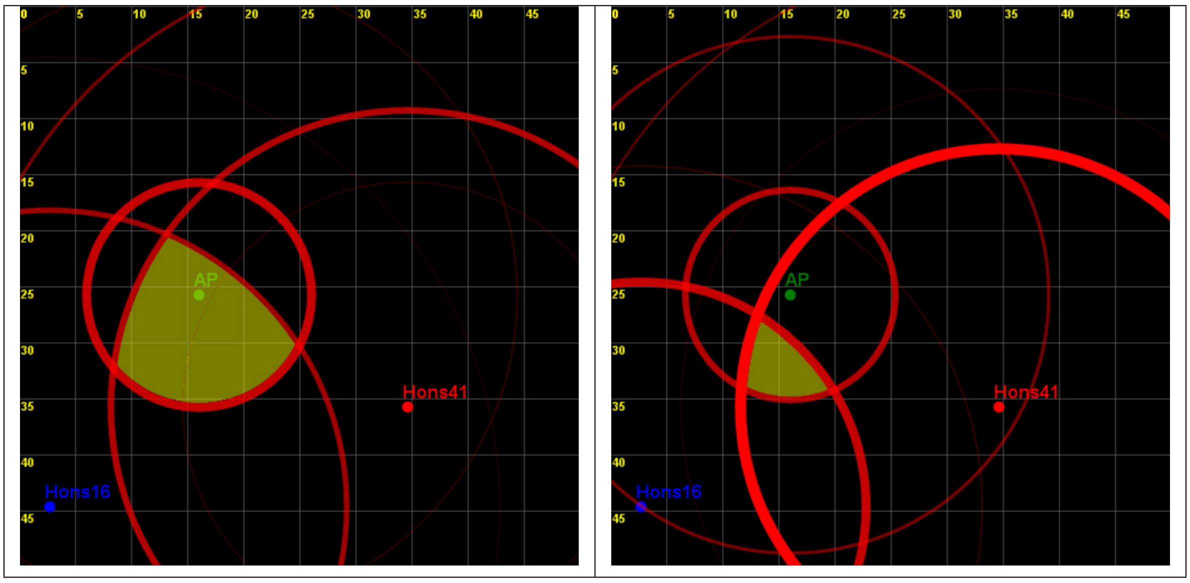
very accurate, as the AP was 2 meters away from the WiSpy SSM Collector at the 'AP' node.

If we take a closer look at Figure 5.8 we see smaller circles originating from the 'SNRGMobile' and 'Hons41' suggesting the signal is originating closer to them than where the AP is located. As both these circles are of a similar brush width, they appear to have been collected around a similar time, it is possible interference could have occurred or that the signal propagated along a shorter path.

5.3.2.2 Result 2

Figure 5.9 displays the visualisation obtained for Result 2, which was obtained with the same layout of WiSpy SSM Collectors as conducted for Result Set 1 and still investigating a channel 6 Wi-Fi network. Result 2 provides less accuracy than Result 1 but provides an area (highlighted in yellow) of approximately $15m^2$ where the signal may be originating from. The AP is located approximately 4 meters away from the area specified in Result 2.

From analysis of Result 2, the area where the signal is originated from is fairly accurate, a person walking around the area could potentially see the AP unless it is well hidden. This result could be more accurate with more WiSpy SSM Collectors. Unfortunately in our experimentation we also didn't have full control over the test environment and overlapping signals from interference sources could have skewed our results.



Result 3A	Result 3B

Figure 5.10: WiSpy SSM Compiler - Result Set 3

5.3.2.3 Result Set 3

The Result Set 3 is shown in Figure 5.10, a different physical layout of WiSpy SSM Collectors was used to collect these results. Result Set 3 is also based on a channel 6 network. The area of intersection for Result 3A (highlighted in yellow) is larger than the previous two results (Results 1 and 2) but shows a fairly accurate display of where the AP may be. Result 3A provides an area where the AP is located and as with Result 2, a person walking around the area provided could potentially see the AP.

Result 3B is shown in Figure 5.10 and has been run under the same conditions as Result 3A, except that it is displaying a slightly different time. Result 3B shows a smaller intersection area (highlighted in yellow) than Result 3A but the AP is not contained within this area. Again, a person walking around this area could potentially see the AP.

5.3.2.4 Result Set 4

Result Set 4 can be seen in Figure 5.11. Two results were obtained under a new physical and test layout. Channel 11 was utilised in the Result Set 4. A WiSpy SSM Collector was not placed near the AP for these results. Instead the three collecting nodes where situated around the AP and all at equal distances from it. An almost equilateral triangle of WiSpy SSM Collector nodes, with the AP at the mid-point of the triangle. The results were promising.

In Result 4A in Figure 5.11, although the most current circles do not intersect, the dis-

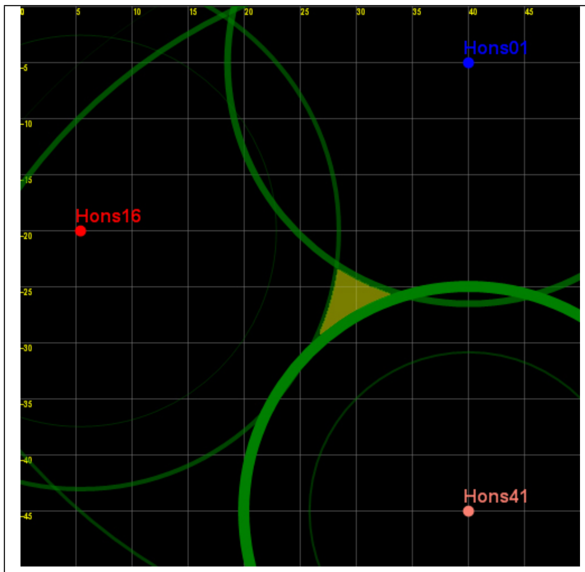
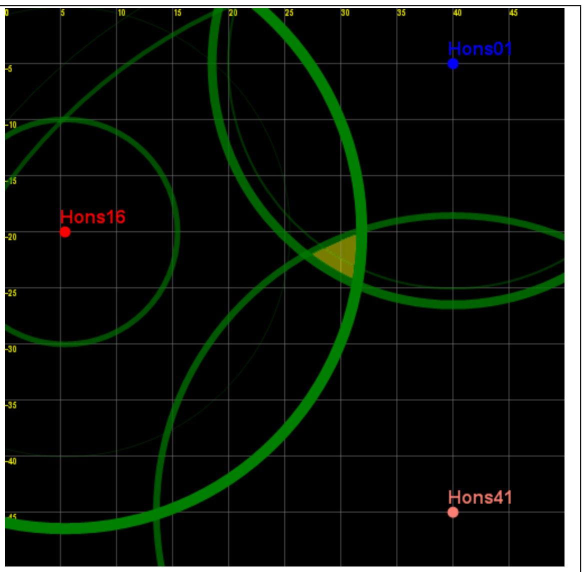
	
Result 4A	Result 4B

Figure 5.11: WiSpy SSM Compiler - Result Set 4

tances computed by the WiSpy SSM Compiler suggest the signals are being transmitted from near the highlighted location. Result 4A and 4B provide very similar areas of intersection, for the duration of this experiment most of our results suggested this highlighted area for the location of an AP.

5.4 Chapter Summary and Conclusion

The results produced by the WiSpy SSM Compiler were fairly accurate providing either the exact location of the RF source or a very close approximation there of. These results would be adequate in aiding administrators in locating rogue sources and also better planning their wireless networks prior to deployment.

Chapter 6

Conclusions

With the rise of mobile users utilising the 802.11b/g/n Wi-Fi technologies, performance needs to be maintained and security needs to be infallible as much as possible. Network administrators need to keep a look out for rogue APs that are not authorised to be on the secure network. An administrator utilising a low cost 2.4 GHz spectrum analyser can detect interference sources and choose the least cluttered channel for use in their Wi-Fi network. The WiSpy SSM Collector application was developed to reach these goals. By combining multiple WiSpy SSM Collectors around the Wi-Fi network, the network administrator can locate where particular Wi-Fi devices are physically situated by using the WiSpy SSM Compiler.

6.1 Summary

Chapter 2 discussed previous findings and work in the IEEE 802.11 Wi-Fi standard. Trilateration was found to be important to this project meeting its goals as it aided us in locating a point in space by using three other known points. By using trilateration and signal strength received from our WiSpy spectrum analysers we discovered we could locate Wi-Fi devices. Finally, an overview of Wi-Fi security protocols/standards were discussed.

Chapter 3 went on to discuss the design of our solution for discovering Wi-Fi devices using WiSpy spectrum analysers. The requirements were analysed to specify a three part solution, with the overall design being disconnected client-server architecture. The solution was named the WiSpy SSM Tool. The first part of the solution was composed of the WiSpy SSM Collector, which enumerates signal data from a connected WiSpy spectrum analyser and transmitted the signal data combined with dynamic location details to the webservice. The location details were automatically updated by interfacing with a GPS. The WiSpy SSM Compiler would retrieve data from the webservice and display it to the user, which provided a visualisation of the Wi-Fi network in which the WiSpy SSM Collectors were situated.

Chapter 4 used the design to implement the WiSpy SSM Tool. Development was conducted in Microsoft Visual C#. The webservice was an ASP.NET Webservice and was deployed on Microsoft IIS. The webservice was an interface to a SQLite lightweight database where all the signal data was stored and retrieved.

The WiSpy SSM Collector could dynamically update its location information using a connected GPS device which allowed roaming listening nodes. A connection to the webservice was only necessary when transmitting data, signal data could be collected without an Internet or network connection. The signal data could be transmitted from the WiSpy SSM Collector when the connection was re-established. A line graph was provided on the WiSpy SSM Collector interface to provide feedback and basic analysis of the 2.4 GHz Wi-Fi spectrum.

The WiSpy SSM Compiler would retrieve signal data from all stored node data in the SQLite database through the webservice. The signal data would be sorted chronologically and graphically displayed for analysis. The signal data was separated into Wi-Fi channels and each channel was displayed separately as a circle originating from the node. The circles would be larger if the overall signal strength for that particular channel was little, and the circles would be smaller (closer to the node) if the signal strength was high. A formula (see Algorithm 6) was used to convert signal strength to distance, this formula was for the ideal line-of-sight scenario and did not take into account multipath propagation effects or free space loss by obstructions.

Once the full solution was developed, it was evaluated for accuracy, Chapter 5 discusses results obtained. Results were gathered for the WiSpy SSM Collector and evaluated. These results were found to be accurate for the WiSpy spectrum analysers. The results from the WiSpy SSM Collector had to be accurate as their signal data was the basis for the WiSpy SSM Compiler generating correct results. Multiple experiments were conducted with the WiSpy SSM Compiler and different WiSpy SSM Collector location setups. Results obtained were fairly accurate in discovering the location of Wi-Fi APs.

6.2 Problem Statement Revisited

The WiSpy SSM Tool was developed to meet the project goals. This tool contains the WiSpy SSM Collector, WiSpy SSM Compiler and the ASP.NET Webservice. The WiSpy SSM Collector analysed RF spectrum (2.4 GHz range) from a connected WiSpy spectrum

	WiSpy v1	WiSpy 2.4x
Antenna	Internal PCB Trace	External, RP-SMA
Bandwidth	2400 to 2483 MHz	2400 to 2483.5 MHz
Frequency Resolution	1 MHz	328 KHz
Amplitude Range	-97 dBm to -50.5 dBm	-110 dBm to -6.5 dBm
Amplitude Resolution	1.5 dBm	0.5 dBm
Sweep Time	120 ms	165 ms
Price	$99 USD	$399 USD

Table 6.1: WiSpy v1 VS WiSpy 2.4x [21–23]

analyser and from the output a user could select the least cluttered Wi-Fi channel for utilisation in a Wi-Fi network. Interferences could also be detected using the WiSpy SSM Collector. Multiple WiSpy SSM Collector applications situated within a Wi-Fi network transmit data to the webservice to be stored in a SQLite database. The WiSpy SSM Compiler retrieved data from the database and combined the frequency VS signal amplitude from three (or more) of the collecting nodes it was possible to locate 2.4 GHz Wi-Fi devices. Devices were discovered by graphically displaying the Wi-Fi signal data per channel as different size circles originating from the node depending on signal strength. Wi-Fi devices are discovered where the circles overlap (intersect).

6.3 Future Extensions

MetaGeek has released the WiSpy 2.4x Spectrum Analysers [23], Table 6.1 compares the new release to the WiSpy v1 that was used in this project. Firstly, the newer WiSpy has an external antenna allowing any type of antenna to be connected to it (dipole or directional Yagi antenna). The new device also operates on the 2.4 GHz band and the resolution of the frequency is improved to 0.328 MHz compared to 1 MHz. The amplitude range has dramatically improved, which would provide a higher accuracy in calculating distances. The amplitude resolution is improved to 0.5 dBm which would further increase accuracy. The minimal increase in sweep time would not affect the results. Although four times the cost, the WiSpy 2.4x spectrum analyser would impressively improve the accuracy of locating Wi-Fi device.

Templates for types of interferences could be implemented into the WiSpy SSM Collector to automatically detect specific interference sources (whether they be BlueTooth devices, microwaves or cordless phones). The WiSpy SSM Compiler could be modified to display the signal data received from each node (similar to that of the WiSpy SSM Collector).

The WiSpy SSM Compiler could integrate an option for under laying an image of the area under investigation, for example an image of the Rhodes University Hamilton Building, or even a map of a town. A further extension would be for the WiSpy SSM Compiler to automatically calculate the graphical displays scale by giving three known GPS points. As we know, if you have three known points and each have a known distance to a fourth unknown point, you can calculate its exact position using trilateration.

Bibliography

[1] Aircrack-ptw. Cryptography and computeralgebra. Online: `http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/`, Accessed: 28/05/2007, 2004.

[2] Arbaugh, W. A., Shankar, N. and Wan, J. Your 802.11 wireless network has no clothes. *Department of Computer Science, University of Maryland* (2001).

[3] Axelson, J. The HID Page: Resources for Developers of USB Devices in the Human Interface Device Class. Online: `http://www.lvr.com/hidpage.htm`, Accessed: 27/06/2007, 2007.

[4] Bahl, P., Padmanabhan, V. N. and Balachandran, A. Enhancements to the RADAR User Location and Tracking System. *Microsoft Research* (2000).

[5] Balfanz, D., Durfee, G., Grinter, R. E., Smetters, D. K. and Stewart, P. Network-in-a-box: How to Set Up a Secure Wireless Network in Under a Minute. *Palo Alto Research Center* (2004).

[6] Bardwell, J. *I'm Going To Let My Chauffeur Answer That: Math and Physics for the 802.11 Wireless LAN Engineer.* 2003.

[7] Borisov, I., Goldberg, I. and Wagner, D. *Intercepting Mobile Communications: The Insecurity of 802.11.* 2001.

[8] Button, D. Tech articles: Effect of obstructions on RF signal propagation. Online: `http://www.emswireless.com/english/Tech_Articles/tech_art03.asp`, Accessed: 19/03/2007, 1999.

[9] Cisco Systems. Cisco Wireless Location Appliance. *Data Sheet* (2006).

[10] Editors of IEEE 802.11. Part II: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, High Speed Physical Layer in the 5 GHz Band. Tech. rep., Institute of Electrical and Electronics Engineers, Inc., New York, IEEE 802.11a-1999 edition, 1999.

[11] Editors of IEEE 802.11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Higher Speed Physical Layer Extension in the 2.4 GHz Band. Tech. rep., Institute of Electrical and Electronics Engineers, Inc., New York, IEEE 802.11b-1999 edition, 1999.

[12] Editors of IEEE 802.11. Part II: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Further Higher Data Rate Extension in the 2.4 GHz Band. Tech. rep., Institute of Electrical and Electronics Engineers, Inc., New York, IEEE 802.11g-2003 edition, 2003.

[13] Farpoint Group. Evaluating interference in wireless LANs: Recommended practice. *Fairpoint Group Technical Note* (2006).

[14] Ferrara, A., and MacDonald, M. Programming .NET Web Services. Online: `http://www.oreilly.com/catalog/prognetws/chapter/ch02.html`, Accessed: 27/08/2007, 2002.

[15] Finisar.SQLite. An ADO.NET Data Provider for SQLite. Online: `http://adodotnetsqlite.sourceforge.net/`, Accessed 01/09/2007, 2006.

[16] Franson Technology. Franson GpsTools - GPS/GIS .NET C, VB.NET. Online: `http://franson.com/gpstools/`, Accessed 10/10/2007.

[17] Franson Technology. Franson GpsGateGPS. Splitter, simulator, logger and serial port splitter. Online: `http://franson.com/gpsgate/`, Accessed: 10/10/2007, 2007.

[18] Geier, J. Performing radio frequency site surveys to effectively support VoWLAN solutions. *Helium Networks* (2006).

[19] Higgins, T. MetaGeek Wi-Spy 2.4 GHz Spectrum Analyzer. Online: `http://www.smallnetbuilder.com/content/view/24766/96/`, Accessed: 04/03/2007, 2006.

[20] MetaGeek. Wi-Spy Hardware Interface Specification. Online: `http://www.metageek.net/products-wi-spy-24x/development-specifications`, Accessed: 05/06/2007, 2006.

[21] MetaGeek. WiSpy V1 Spectrum Analyser. Online; `http://www.metageek.net/products/wi-spy`, Accessed: 04/03/2007, 2006.

[22] MetaGeek. MetaGeek Store. Online: `https://www.metageekstore.com/`, Accessed: 04/03/2007, 2007.

[23] MetaGeek. WiSpy V2.4x Spectrum Analyser. Online: `http://www.metageek.net/products/wi-spy-24x`, Accessed: 20/10/2007, 2007.

[24] Microsoft Corporation. Internet Information Services 5.1. Online: `http://www.microsoft.com/windowsxp/evaluation/features/iis.mspx`, Accessed: 10/10/2007, 2002.

[25] Microsoft Corporation. Microsoft Visual C++ 2005 SP1 Redistributable Package (x86). Online: `http://www.microsoft.com/downloads/details.aspx?familyid=200B2FD9-AE1A-4A14-984D-389C36F85647\&displaylang=en`, Accessed: 10/10/2007, 2007.

[26] MICROSOFT DEVELOPER NETWORK. Building web services. Online: http://msdn2.microsoft.com/en-gb/webservices/aa740648.aspx, Accessed: 27/08/2007, 2007.

[27] MICROSOFT DEVELOPER NETWORK. MSDN Home Page. Online: http://msdn.microsoft.com/, Accessed 09/06/2007, 2007.

[28] MURPHY, W. S. AND HEREMAN, W. Determination of a position in three dimensions using trilateration and approximate distances. *Colorado School of Mines* (1999).

[29] NAIR, J. Asynchronous Socket Programming in C: Part I. Online: http://www.codeguru.com/csharp/csharp/cs_network/sockets/article.php/c7695/, Accessed: 26/08/2007, 2005.

[30] NETSTUMBLER. Netstumbler v0.4.0 release notes. Online: http://downloads.netstumbler.com/downloads/netstumbler_v0.4.0_release_notes.pdf, Accessed: 05/05/2007, 2007.

[31] RASHEED, F. Implementing Design Patterns in C - Singleton Pattern. Online: http://www.c-sharpcorner.com/UploadFile/faraz.rasheed/SingletonPattern12052005063955AM/SingletonPattern.aspx, Accessed 03/07/2007, 2003.

[32] RAY S., CARRUTHERS J. B. AND STAROBINSKI, D. Evaluation of the masked node problem in ad hoc wireless lans. *IEEE Transactions on Mobile Computing 4*, 5 (2005), 430–442.

[33] RIVEST, R. L. RC4 Encryption Algorithm. *RSA Data Security, Inc.* (1992).

[34] ROSE, C., ULUKUS, S. AND YATES, R. Wireless systems and interference avoidance. *WINLAB, Department of Electrical and Computer Engineering, Rutgers University* (2000).

[35] SQLITE. SQLite Home Page. Online: http://www.sqlite.org/, Accessed 01/09/2007, 2007.

[36] THE SCHMOO GROUP. AirSnort homepage. Online: http://airsnort.shmoo.com/, Accessed: 05/05/2007, 2007.

[37] TROPOS NETWORKS. 802.11 Technologies: Past, Present and Future. Online: http://www.tropos.com/pdf/technology_briefs/tropos_techbrief_wi-fi_technologies.pdf, Accessed 22/10/2007, 2007.

[38] WALKER, J. R. Unsafe at any key size; an analysis of the WEP encapsulation. *Intel Corporation* (2000).

[39] WEISSTEIN, E. W. Law of sines. Online: http://mathworld.woflram.com/LawofSines.html, Accessed: 15/06/2007, 2003.

[40] Wi-Fi Alliance. Wi-Fi Protected Access: Strong, standards-based, interoperable security for today's Wi-Fi networks. Online: `http://www.54g.org/pdf/Whitepaper_Wi-Fi_Security4-29-03.pdf`, Accessed: 04/04/2007, 2003.

[41] Yang, X., and Petropulu, A. P. Joint statistics of interference in a wireless communications link resulted from a poisson field of interferers. *Electrical and Computer Engineering Department, Drexel University Philadelphia* (2001).

Appendix A

Wi-Fi Encryption, Cracks and Tools

A.1 WEP and WPA Encryption

WEP uses the RC4 cipher [33], which contains several major security flaws, giving rise to a number of attacks that directly violate the goals explained above. The protocol standard specifies use of a 40-bit key however several manufacturers have extended the key length to 104-bits, the latter rendering a brute-force attack on the key impossible. There are shortcuts to brute-force attacks in order to obtain the key, therefore not even 104-bit WEP keys are secure. The typical cipher strength is often advertised as 64-bit, with a 40-bit key and a 24-bit public initialisation vector (IV), or 104-bit key and 24-bit public IV.

For every packet sent, a new IV is generated and appended to the key, RC4 is used to generate a key stream of the length of the plaintext and this key stream is XORed against the plaintext data. The sender appends the IV into the frame header of the packet, and a tag is set indicating that this is a WEP packet [38]. The packet is then sent.

The IV is 24-bits long, appending to the shared key to form a family of 2^{24} keys, each frame transmission selects one of these 2^{24} keys and encrypts under the key. This leads to widespread key abuse as a single AP running at 11 Mbps with a typical throughput can exhaust the key space in about an hour [38].

Numerous attacks exist against WEP. In a TCP/IP network, every data frame contains an IP datagram containing large amounts of known plaintext. In such an environment, an attacker can recover a partial stream for every packet sent. Analysis can be conducted on the traffic to identify the type, and this information can be used to guess the values or other variables in the packet headers, revealing more of the data and the key stream. The known plaintext from a single DHCP exchange can provide enough information to decode almost the entire TCP/IP header of every IP datagram encrypted by the DHCP client. Any fully decrypted packet provides the context and hints that can be used to identify the plaintext of a still not fully decrypted packet. This process can continue until the key streams are revealed for almost every IV [38].

WPA uses the Temporal Key Integrity Protocol (TKIP) for encryption and employs 802.1X authentication with one of the standard Extensible Authentication Protocol (EAP) types available today. AP's and client network interface cards can be upgraded to use WPA by the means of a firmware upgrade. Enterprises will require an authentication server, and home users can utilise a special feature which allows the use of a shared password [40].

TKIP increases the size of the key from 40 to 128 bits and replaces WEP's single key with dynamically generated keys that are distributed by an authentication server. It uses a key hierarchy and key management methodology that removes the predictability which crackers relied upon to exploit WEP. After the authentication server has accepted a client, 802.1X is used to produce a unique master key or pair-wise key for that particular session. TKIP distributes this key to the client and the AP, which dynamically generates unique encryption keys for that session. On any given data packet in that session, 500×10^{12} possible keys exist.

A protection scheme against alterations to packets is the Message Integrity Check (MIC), which is designed to prevent an attacker from modifying data in individual packets and sending them on. This check involves a strong mathematical function in which the receiver and sender both compute and then compare a MIC, if they do not match, the packet is assumed to be tampered with and dropped.

WPA uses 802.1X authentication with EAP; EAP handles the presentation of users' credentials in the form of digital certificates, unique usernames and passwords, smart cards or any other secure identity. The framework allows clients to mutually authenticate with the authentication server, which prevents clients from accidentally connecting to rogue AP's and ensures that users who access the network are the ones who are supposed to be there.

A.2 Wireless Networking Tools and Cracks

AirSnort is a WLAN tool which recovers encryption keys, passively monitoring transmissions and computing the encryption key when enough packets have been gathered, this tool requires approximately 5 to 10 million encrypted packets to be gathered, and can guess the encryption password in under a second [36]. A newer, more advanced WEP encryption cracker is AirCrack-ptw, which can recover the key using as little as 40,000 captured packets which can be captured in less than a minute under good conditions [1].

Wireless networks can be discovered and more information can be gathered about them using a tool called NetStumbler [30], this tool is extremely versatile and provides many features. Typical information gathered about any AP is the MAC address, Service Set Identifier (SSID), channel used, type of wireless (802.11a/b/g/n), form of encryption used, estimated distance of transmission, IP address and subnet.

Appendix B

Included with CD Deliverable

A list of contents on the accompanying CD is as follows:

- Final Project Proposal
- Final Literature Review
- Final Poster
- Final Presentation
- Final Thesis
- All References included in Thesis
- WiSpy SSM Tool - Application Assemblies and Precompiled Webservice
- WiSpy SSM Tool Visual C# Source code, dependencies and handy utilities

Glossary

AES - Advanced Encryption Standard

AP - Wi-Fi Access Point

API - Application Program Interface

DHCP - Dynamic Host Configuration Protocol

DNS - Domain Naming System

GPS - Global Positioning System

GUI - Graphical User Interface

HID - Human Interaction Device

LAN - Local Area Network

NMEA - National Maritime Electronics Association

PC - Personal Computer

PDA - Personal Digital Assistant

PID - Product Identification

RF - Rafio Frequency

RSSI - Receive Signal Strength Indication

SOAP - Simple Object Access Protocol

TCP/IP - Transmission Control Protocol / Internet Protocol

Throughput - A measure of the amount of data transferred in a specific amount of time, usually expressed as bits per second (bps)

UDP	- User Datagram Protocol
USB	- Universal Serial Bus
VID	- Vendor Identification
VoFi	- Voice over Wi-Fi
VPN	- Virtual Private Network
WDoS	- Wireless Denial-of-Service Attack
WEP	- Wired Equivalent Privacy Protocol
Wi-Fi	- Wireless Fidelity
WLAN	- Wireless Local Area Network
WPA	- Wi-Fi Protected Access Protocol
XML	- Extensible Markup Language