Estimating Passenger Flow & Occupancy on Board Public Transport Buses Through Mobile Participatory and Opportunistic Sensing

by

Stephen Brandon, B.Sc.

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfillment

of the requirements

for the Degree of

Master of Science in Networks & Distributed Systems

University of Dublin, Trinity College

September 2015

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Stephen Brandon

August 27, 2015

Permission to Lend and/or Copy

Acknowledgments

I wish to thank my project supervisor Prof. Siobhán Clarke for her support, constructive suggestions, and feedback which helped guide me throughout the course of my work on this dissertation. I also wish to extend my sincere gratitude to my co-supervisor Dr. Mauro Dragone who kindly offered technical advice and suggestions to any problems I could not solve alone.

I would also like to thank the many people that contributed to the production of this dissertation especially to my friends that volunteered to test the application.

Stephen Brandon

University of Dublin, Trinity College

September 2015

Estimating Passenger Flow & Occupancy on Board Public Transport Buses Through Mobile Participatory and Opportunistic Sensing

Stephen Brandon, M.Sc.
University of Dublin, Trinity College, 2015

Supervisor: Siobhán Clarke

Estimating passenger flow and occupancy on public transport buses usually involves dedicated hardware or conducting surveys, both of which can be expensive and time consuming. Bus occupancy data is useful for a variety of reasons. Bus operators can reduce running costs and environmental impact by understanding how to optimise the way they use their fleet. Passengers feel more comfortable and are more likely to use public transport when it is not congested.

Recent research has addressed various methods of collecting bus occupancy data. Much of this research has focused on hardware that needs to be installed on board the bus such as cameras and Wi-Fi Scanners. Crowdsensing applications have proved successful in other smart city projects including public transport applications such as estimating bus arrival time.

In this project we propose a novel solution to estimate passenger occupancy and flow on board buses. A mobile crowdsensing application capable of detecting other mobile devices through Bluetooth and Wi-Fi was created. Using a binary classification model we found that it is possible to differentiate if a bus is full or not based on the number of Wi-Fi users.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter we will introduce the project and discuss the motivation behind it as well as our objectives and expected results. Finally we will outline the structure of this document.

1.1 Background

The ability to calculate passenger flow and occupancy on board a bus in real time has many benefits for both the passengers and the operators providing the bus service. Buses provide a sustainable alternative to ease traffic congestion in cities, however to encourage people to use buses the provision of real time information such as expected arrival time and bus occupancy is important. The same real time data used to assure passengers that the service is running normally can be used by the bus operators for future planning, scheduling services and timetables, and reducing costs and environmental impact.

Much of the recent research into estimating passenger flow on board buses has involved using cameras [41] [15] or pressure sensors [43] to detect passengers entering or exiting the bus. While these methods have a high accuracy rate they have a number of problems that make them difficult to implement. Not only is the cost of installing hardware across a fleet of buses high but the on going maintenance and upgrading of these systems can be expensive.

In 2009 Dublin bus Implemented their RTPI (Real Time Passenger Information) system. This system provides real time bus location and schedule information to Dublin Bus and its passengers. The cost of installing the hardware on the buses was 5.4 million euro and an additional 3 million needed to be spent to install infrastructure to display this information at bus stops [10]. Despite this cost Dublin Bus still treated the implementation of RTPI as a high priority demonstrating the the value of real time information to themselves and their passengers.

In 2013 there were over 112 million journeys taken on the Dublin Bus fleet of over 900 buses [3]. It is clear that when operating a bus fleet on this scale that there would be significant savings to be made by having access to this type of real time data without the need to install additional hardware on each bus.

Advancements in mobile phone technology such as the number of sensors available in a typical smart phone make crowdsensing applications a realistic solution. In the case of real time information for public transport the participants are likely to engage in participatory sensing because the real time information it produces is useful to them.

The demand for real time information and advancement in mobile phone technology creates an exciting gap in the state of the art that crowdsensing applications can fill.

1.2 Motivation

This project is motivated by the benefits that can be gained by providing real time passenger information to public transport users and operators as well as the opportunity to contribute to the state of the art in crowd sensing for confined spaces such as a bus.

In order for participatory sensing applications to be useful both the data producers and consumers must benefit in some way from the data produced or else there would be no motivation to continue using the application. This project presents a real opportunity to use participatory sensing to its full potential as both data producers and consumers can potentially benefit from participating which represents the perfect scenario for crowd-sensing.

1.3 Hypothesis

The aim of this study is to investigate if a relationship exists between the number of discoverable Bluetooth devices on board a bus, the number of passengers connected to the buses Wi-Fi and the actual number of passengers on board a bus.

We propose that a relationship can be established between Wi-Fi users or discoverable Bluetooth devices and the actual number of passengers on board a bus. This relationship will allow us to infer the current occupancy of the bus where the data is collected.

1.4 Research Approach

1. Review current solutions to estimating passenger occupancy and flow on board buses and determine if these solutions are viable and if they could be implemented in a crowd sensing application.

2. Based on information learned during the review design a suitable crowdsensing application that is capable of collecting ground truth values from participants which can be used as benchmark values during evaluation.

3. Collect data by having a team of volunteer participants use the crowdsensing application during their daily commute. Participants will provide ground truth data along with the data that the application will produce.

4. Test the hypothesis by determining if a relationship between the ground truth values and the data collected by the application can be made.

5. Evaluate the effectiveness of the application based on the test results and investigate where the application could be improved.

1.5 Document Structure

Chapter 2 introduces the state of the art in participatory and opportunistic crowdsensing applications. In Chapter 3 we describe the design for our crowdsensing application and the RESTful web service. Chapter 4 will examine the implementation of our application. We will evaluate the data collected by our application to test our hypothesis in Chapter 5. Finally we give an overview of the project and our contribution in Chapter 6.

Chapter 2

State of the Art

In Chapter One we introduced the research objectives, motivations and the approach to conducting this research. In this chapter we examine the background for the research and review the state of the art in participatory and opportunistic sensing as well as peer to peer networking for mobile devices.

2.1 Participatory Sensing

Participatory Sensing involves the collection of sensor data from sensor nodes or participants in a wireless sensor network type of environment. In recent times smart phones have become popular devices to use as they have many sensors and are highly mobile and relatively inexpensive. Large numbers of smart phones can be tasked to create sensor networks where members of the public can participate in collecting and distributing data. Such applications have a wide range of uses such as in healthcare and urban planning, for example smart cities projects. [4]

Smart phones are suitable devices for participatory sensing for a number of reasons. Most modern smart phones have many sensors built in, to name a few: accelerometer, gyroscope, microphone, camera, GPS, Bluetooth, Wi-Fi, and light sensors. The combination of sensors and widespread use of these devices as well as simple programming interfaces such as the Android SDK make smart phones great participatory sensor nodes [24]. The use of smart phones in participatory sensing is much more economically feasible also as existing devices and communications networks and infrastructure are used [20].

Participatory sensing applications can be categorised as either people centric or environmental centric [25]. People centric applications are used to gather data about individuals, detecting patterns of activity, which can be used to help that individual improve some aspect of their life. There are many examples of people centric applications across a variety of domains including health care, lifestyle, and sleep monitoring to name a few. Environment centric participatory sensing applications are used to collect data about the environments that the participants are in for example air or noise pollution data can be collected in this manner.

2.1.1 Opportunistic Sensing

Opportunistic sensing is a complementary method of collecting data to participatory sensing. With opportunistic sensing the participant or subject does not have to actively participate in the data collection, they may not even be aware of the active applications. Opportunistic sensing occurs when the individuals smart phone meets certain criteria, for example location or environmental noise thresholds [23].

As with participatory sensing there are a number of concerns regarding privacy and resource usage of the sensors but the concerns are heightened as an opportunistic sensing application is designed to run without the user being aware of it running. Additional care must be taken not to leak users private data or hog resources that would affect the performance of the mobile device, for example heavy usage of some sensors such as GPS can have a negative effect of the battery life of the device [31]

2.1.2 Crowdsensing

Crowdsensing was defined in [28] as:

> individuals with sensing and computing devices collectively share data and extract information to measure and map phenomena of common interest.

Both participatory and opportunistic sensing are methods of collecting and distributing data in a Crowdsensing architecture. As the name suggests crowdsensing involves using a crowd of people who possess smart phones to gather and share data rather than sensor nodes as in a wireless sensor network for instance. Crowdsensing has evolved very fast as mobile phones become more powerful and packed with more sensors. In the following related work section we will review the state of the art in crowdsensing applications and investigate how this related work is of interest to my research.

2.1.3 Summary

We have introduced Participatory Sensing, Opportunistic Sensing, and Crowdsensing which are all active areas of study and central to this research project. There are a number of concerns and limitations to these approaches that must be considered during implementation of these methods. The authors of [23] note that participatory sensing applications are limited in their usefulness and sometimes scalability because they require a critical mass of users to participate in order for the application to work correctly. Privacy, Security, and Data Integrity remain some of the top concerns and these must be addressed in any implementation so that participants private data is protected.

2.2 Related Work

In this section we will review previous works in the areas of participatory and opportunistic sensing with particular reference to works that are used in an urban or public transport context. This section aims to look at research that has been carried out that could help overcome the main challenges of realising my research objectives which are:

1. Bus Detection: Recognising when a user is on a bus.
2. Route Classification: Determining what bus route the user is on.
3. Bus Occupancy: Estimating the occupancy of the bus and the remaining capacity.
4. Peer to Peer Communication: Collaborative task distribution.
5. Adaptable Solution: It should be possible to deploy the application on any bus that provides Wi-Fi.
6. Privacy and Security: Maintaining the users privacy and mitigating risks of attack.

2.2.1 Bus Detection/Route Classification

Previous work involving public transport buses [42] identified and provided solutions to the challenges of participatory sensing in this context. These challenges included bus detection, route classification and dealing with the data generated in a timely manner. By identifying actors in their system the authors were able to produce a solution that would satisfy each actor. The authors found that it was possible to accurately predict bus arrival time in a participatory sensing environment using only android enabled smart phones where participants used an application and a server that collected and distributed data. The authors used a variation of techniques such as cell tower signals, accelerometer readings and audio recordings to determine that the user was on a bus and what route they were on. While the authors findings support my research some of the techniques used such as audio recordings, which detected the sound of RFID tickets being validated, would not be suitable in this research. This is because not all buses use the same types of sound for validating tickets which would make the application difficult to implement on any type of bus.

Recent research explores how smart phones, using GPS and a 3-axis accelerometer can be used to determine the users mode of transport with high accuracy [32]. In this paper the authors investigated the use of various sensors in order to determine what mode of transport a person was using e.g. walking, running, bus, or car. The sensors used included Cell tower, Wi-Fi, Bluetooth, GPS, and accelerometer. The author found that while it was possible to determine the mode of transport with most of the sensors the data was too coarse grained to differentiate between some modes of transport e.g. car or bus. Various classification techniques were considered such as the Hidden Markov Model. While these results are promising more research into differentiating between motorised modes of transport will be required in my implementation. An important part of my implementation will involve the application being aware when the user is on a bus as opposed to a car or train rather than having to ask the user, meaning opportunistic sensing methods can be used which are much less disruptive for the user.

Bus route classification involves knowing the participants location. With mobile phones location is usually attained with GPS or Cell Tower ID or both. GPS is highly accurate but is heavy on battery [31]. Although getting the cell tower ID is less power hungry it is not as accurate and so a trade-off has to be made. Related work in the area of energy efficient positioning for smart phones found that using a mixture of cell tower sequence id matching and occasionally GPS to retain accuracy was the most efficient use of the sensors [31]. It was found that using cell towers location accuracy can be within 500 metres which is not accurate enough to determine what bus route a person may be on as bus routes are nearly always close together in cities. The authors developed a system that recorded the cell towers a mobile phone passed and the direction known as cell ID transitions and then refined the results using GPS data. This meant the GPS was only turned on occasionally rather than constantly. While the authors results are interesting I noted that there is still a median of error in the location accuracy and that some system learning was required to make the results more accurate. For my implementation this method could be adopted to be used with other methods to insure accuracy because of how close together some bus routes run.

2.2.2 Crowd Detection

Work in the area of crowd sensing/detection has been carried out on board busses [21]. The authors investigated the use of RFID bus tickets and Bluetooth scanning to create an origin/destination (OD) Matrix. While the aim of their research was not to estimate the occupancy of the bus the authors reached conclusions that will be significant in my research. It was stated in earlier research that 7.5% of people had Bluetooth in discoverable mode [29]. The authors found this number to be 9.7% during their implementation suggesting this number could be growing. A correlation of bus occupancy detected using Bluetooth and RFID found the two were related. The author noted that Bluetooth devices were detected in traffic and at bus stops that were not on the bus and this created noise in the data which needed to be removed. Despite this the findings in this paper support Bluetooth as a promising method of estimating occupancy of a bus. It should be noted however that the solution the author provided required hardware, a Bluetooth scanner, to be installed in the bus which would not be suitable for my implementation but the results are still relevant.

Recent work in crowd detection involves estimating queuing time for supermarkets [39]. In this paper the authors developed a participatory sensing application where the user uploads their shopping preferences and the application can tell them the queuing time in that supermarket. The mobile phone can recognise the supermarket based on Wi-Fi fingerprints or GPS data. The authors found that using cellular tower data was too coarse grained for this type of application and this finding applies to my research as a bus is a smaller space than a supermarket. These findings mean that the use of cellular tower data would need to be refined with another location method such as GPS, as suggested in [31], in order to be useful. The results of this work, while in the area of crowd detection, are actually better suited to bus and route classification in my research.

Recent research in crowd density estimation using Bluetooth [40] shows that the relationship between crowd density and discoverable Bluetooth devices can provide accurate crowd density estimations with up to 75% accuracy. The authors conducted a test in a large area and used an android application installed on a number of different devices to collect results collaboratively. While the authors approach is similar to the problem I am researching their implementation is quite different. It was also noted that the use of "features" is required to prevent results from being skewed by a demographic that is more or less likely to have a Bluetooth discoverable device. Features involves using averages rather than absolute count of devices. This should be possible in my implementation as there could be more than one device taking measurements. There were two considerations outlined by the authors that are particularly relevant to my research. The first is that mobile devices have a scanning area of about ten meters (300m^2 approx) which is more than sufficient for a bus environment. The second is that the transmission frequency of Bluetooth (2.4GHz) can be absorbed by humans in a crowded area which can affect results, this was also mentioned by the authors of Towards Sustainable Transport: Wireless Detection of Passenger Trips [21]. Considering both authors found the same problem I believe that our design should not rely on Bluetooth alone to estimate occupancy of busses but a combination of different approaches where the results will complement each other.

Research into estimating bus occupancy and passenger flow at bus stops has been carried out recently using Wi-Fi packet monitoring [30]. The authors used a Linux based laptop to monitor the packets sent by mobile devices as they probe for networks. These packets were logged along with MAC (medium access controller) addresses and timestamps in order to determine passenger flow an occupancy. While the authors concluded that there was a correlation between the detected Wi-Fi packets and the bus occupancy there were a number of issues with their results. Firstly their approach and experiments required a person with a laptop to be on the bus to take the measurements. Secondly the authors experienced problems with noisy data at bus stops, similar to the problem experienced by the authors of [21]. However it was noted that this noisy data could be due to a number of factors such as bus stops being close together which we believe will be common in all major cities. Interestingly the authors note that they ran similar experiments using Bluetooth but they found their results were not as accurate as using Wi-Fi. We believe that using a combination of the two will help validate and make the results more accurate which is something the author mentioned also but they did no further work on this. While the authors research aims are similar, again the approaches to the problem are different.

2.2.3 Peer to Peer Collaboration

Peer to peer networking is a distributed system architecture where each peer or node on the network is equal. Messages or data are routed from the origin node to the destination node via other nodes in the network, this is usually achieved using routing tables for example a distributed hash table. Unlike the client/server paradigm there is no one node in control of the network in pure peer to peer. Peer to Peer networking is a very active area of research, early work includes Chord [37] and Pastry [34] which are both protocols using distributed hash tables. Peer to peer networks are of interest to researchers because they are highly scalable, more economical than traditional networks, and very resilient to attacks. However security and privacy issues pose a large challenge to peer to peer networking for example routing attacks and malicious nodes are common security concerns. Skype [36] is a well known example of an application that is implemented using a peer to peer architecture.

Early work on mobile Peer to Peer communication focused on creating a peer to peer overlay network built on the IP layer [38]. Other early work focuses on peer to peer architecture [16] introducing variations on pure peer to peer which can help with scalability and network overhead such as routing. Most of the early work in mobile peer to peer pre-dates the now popular mobile operating systems such as Android, IOS, and Windows mobile which have made it easier for researchers to create peer to peer communications frameworks.

Android in particular has become a popular platform for research [17]. In this paper the authors propose an Android based middleware framework to allow peer to peer communication over Bluetooth or Wi-Fi. The authors outline how android, due to being open source, is a good platform for this type of development. Android provides a Rich, well documented, SDK (Software Development Kit) to developers.

More recently both Android [13] and Apple IOS [2] have provided libraries for peer to peer communications in their SDK making development even easier for developers. However as with crowdsensing much work remains to be done with regard to privacy and security of nodes and this is currently one of the most limiting factors of implementing a peer to peer solution.

2.2.4 Privacy

As mentioned at the beginning of this chapter privacy is very important in participatory sensing because as outlined in [9] users will not participate in sharing their data if they believe that their privacy could be compromised. In this paper the authors propose an architecture for privacy in participatory sensing applications. The architecture introduces an intermediary between queries for sensor data in the network and the mobile nodes that receive these queries. The intermediary is called a registration authority. All communications in the architecture are encrypted using identity based encryption. The authors address the main problems with privacy and security in participatory sensing however their architecture is not based on a collaborative or peer to peer network design, rather a centralised system which would require extra work to be done to make it suitable for a peer to peer architecture. The authors acknowledge that significant work still needs to be done to improve security and privacy in crowdsensing.

There are many different solutions available that claim to protect the privacy of both data producers and consumers in participatory sensing systems. It appears to be widely agreed that anonymity is required as well as secure communication channels for any sensitive participant data. Research carried out earlier this year showed that it is possible to protect user data without the need for third parties [22]. This result is good news for participatory sensing applications but the architecture assumes that a data consumer and producer are separate people/devices which is not the case in many participatory sensing applications. The paper did not address peer to peer network architecture either but I believe that the conclusions and methods used will still be very useful.

Privacy and security remain active areas of research for peer to peer communication also [35]. In this paper the authors describe how both the network and the individual peers can be vulnerable to attack and how malicious peers can exploit weaknesses in the architecture in order to carry out these attacks. This is very useful research because it highlights the areas where care must be taken with users private data. The idea of reputation management was also introduced where peers can become trusted over time. While these trust management approaches are common in peer to peer it was noted that it can become complicated and does not prevent attacks but rather mitigates against them.

Chapter 3

Design

Building on our findings during the Review of the State of the Art in the previous chapter this chapter sets out to introduce the chosen design of our crowd sensing application. First we will discuss the requirements for such an application then discuss the various components required to satisfy those requirements before introducing the overall design in detail. We will also discuss factors that influenced the design of the application. We have decided to call the mobile application "PassengerSense" making reference to the passengers on the bus and the crowd sensing nature of the application.

3.1 Requirements Engineering

During our review of the state of the art we identified some viable options for crowd sensing on board of a bus. Crowd sensing in such a confined space brings new challenges, which will be outlined in this chapter, and these challenges must be dealt with in the design phase of the project. We identified that Bluetooth and Wi-Fi represented the best chance of relating to the number of passengers on board the bus. The following sub sections will outline the functional and non functional requirements that will shape the architecture and design of our system. Each requirement will be justified with a brief background but we will save the technical details for Chapter 4.

3.1.1 Bluetooth Scanning

- The application must be capable of scanning to detect discoverable Bluetooth devices.

One solution to detect passenger flow and occupancy with the application will be to continuously scan for discoverable Bluetooth devices and collect the unique MAC address of the devices it detects. Using the MAC address the application can determine if the unique device has been previously seen which will give an indication to passenger flow.

Most mobile phones are equipped with class two Bluetooth radios meaning that they have an effective range of about ten metres. While this range is sufficient to cover the area of a bus it could potentially detect devices that are not on board the bus, at traffic lights and bus stops for example. Steps will need to be taken to reduce the noise in Bluetooth data and determine the difference between devices not on the bus and normal passenger flow. Implementing timestamps on all collected data should help to minimise noise in the data by eliminating results that last only a couple of seconds e.g. detected Bluetooth devices at a bus stop.

3.1.2 Wi-Fi Scanning

- The application must be capable of scanning to detect the number of devices connected to a buses Wi-Fi router.

Many bus operators, such as Dublin Bus, offer free Wi-Fi to passengers. The number of passengers connected to the network on board the bus will potentially provide an indication of the total number of passengers on board as well as indicate passenger flow.

Similar to the approach taken with Bluetooth scanning the application will need to continuously ping other devices on the Wi-Fi network and collect a unique MAC address to determine passenger flow and occupancy.

From our review of the state of the art we learned that only a sample of the population (between 7.5 - 9.7 %)[21][29] have their devices in Bluetooth discoverable mode which is why scanning for Wi-Fi and Bluetooth users will provide more reliable data to make estimates with. The results from the Wi-Fi scan may be useful on their own but may also provide a useful validation metric for the results from the Bluetooth scan as these results on their own would provide fairly coarse grained estimates of passenger flow and occupancy.

3.1.3 Ground Truth

- The application must be capable of collecting ground truth values for the number of passengers on board a bus.

To properly test our hypothesis we will require ground truth data about the number of actual passengers on the bus and the route number. The application will need an interface where users can enter the results of the actual number of passengers.

The ground truth data will be compared with the data collected from both Wi-Fi and Bluetooth Scanning to investigate if a relationship exists and if it does what is the nature of that relationship.

3.1.4 Cell Tower

- The application must be capable of detecting the RSSI of the cell tower it is connected to as well as neighbouring cell towers.

While collecting RSSI values for nearby cell towers is not useful towards determining passenger flow or occupancy the data will be collected by the application for use in other future cities projects. For example RSSI can be used to accurately predict arrival times of buses through a method known as cell tower sequence matching [42].

In future work the passenger occupancy and RSSI data may be used to tell users when the next bus is coming and how many seats are likely to be left on the bus when it arrives.

3.1.5 Location

- The application must be capable of recording the GPS Location of the device it is running on.

GPS data may be used to determine the location of a bus which can be used as a filtering value when evaluating data about individual journeys that will be returned by the application. GPS data may be used together with RSSI to provide highly accurate location services that require less battery to run [31]. The GPS data also acts as a way of validating any results reached through the use of RSSI data. RSSI in combination with GPS samples may be used as an energy efficient alternative to pure GPS to infer what bus route a person is currently on.

3.1.6 Storage

- The application must be capable of storing collected results locally and remotely.

Persistent storage of scan results will be necessary in order to evaluate our application later. The application must be capable of storing these scan results in its own storage as well as posting them to a remote web service. The web service must be platform independent meaning any device with an internet connection could potentially use it. A platform independent web service will help keep the mobile application loosely coupled with the web service which will save a lot of effort if the application needs to be implemented on other platforms such as IOS or in the event that a platform update would otherwise require an update to the application code.

There are a number of benefits to implementing a web service as well as local device storage. Firstly if the phone cannot contact the web service due to having no network available or some kind of error with the web service then the results are not lost. Having data stored in more than one location is good practice as it builds redundancy into the system giving it a higher rate of availability even in the face of errors.

The trade-off for this high availability and redundancy is that a web service will need to be built as well as the mobile application. The web service adds additional security considerations to the system which are explored in Appendix D. In this case the trade-off is made worth while because when testing the application it will not be necessary to retrieve the participants phone to collect the data as it will have been sent to the web service.

3.1.7 Ethical Considerations

- The application must comply with ethical standards for research.

Any research project involving human participation must be approved by the Research Ethics Committee. For a project to be approved it must adhere to ethical guidelines. Participants taking part in the study must do so on a voluntary basis and are free to opt-out at any time without reason or penalty. Anonymity should be provided for any data collected about participants direct or indirect. In this study no private data will be collected or stored on any device.

When building this application the implications of the ethical guidelines will present a challenge that both passengers and application user data must be kept anonymous despite the need to be able to uniquely identify devices for counting purposes. The ethical guidelines are a non-functional requirement meaning they provide no functionality but still affect the overall design of the application and web service.

Passenger/Participant Privacy

- Any data detected, or collected by the application must be kept securely and privately.

As noted in the State of The Art Review in section 2.2.4 one of the concerns with crowd sensing is the privacy element. Users need to feel that their personal data is secure or they will be unlikely to participate. Steps must be taken during the design process to ensure that user data is kept secure.

To satisfy this requirement it will be necessary first to identify sources private information such as the MAC addresses of discovered devices. Next we will need to identify the risks associated with this data for example how an attacker could potentially track a passenger even off the bus using their MAC address. Finally steps need to be taken to properly secure that data and mitigate the security and privacy risk to the user. Here there will be some overlap with the ethical consideration requirement as outlined in section 3.1.7. Similar to the ethical considerations passenger privacy is another non-functional requirement but still affects the overall design of the application and web service.

3.2 Design Choices

In this section we will introduce the overall design and architecture of the mobile application by presenting a UML activity diagram before describing the main components of the design in greater detail. We will discuss various design challenges as well as what impact or trade off the chosen design had on the application.

3.2.1 Mobile Application

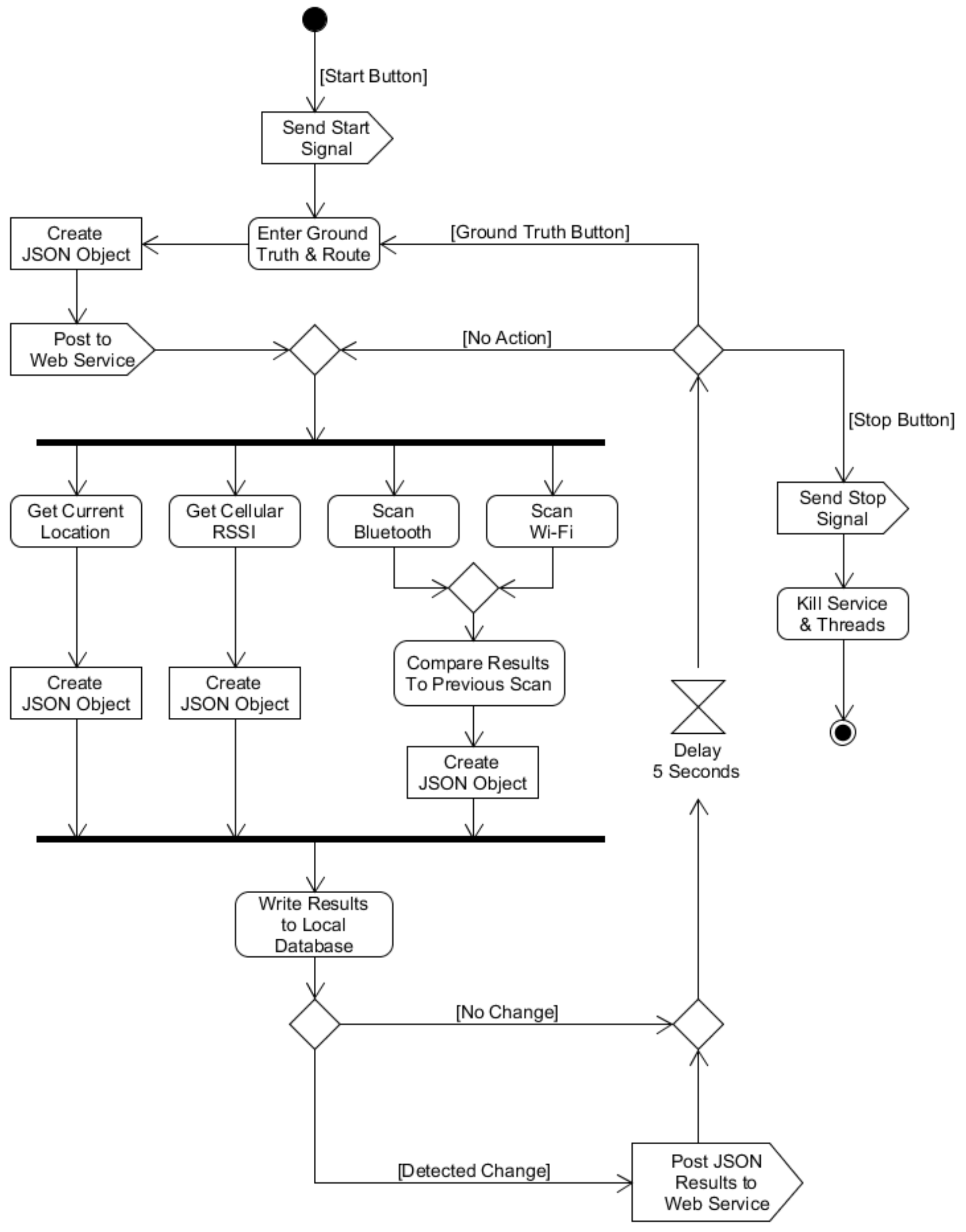


Figure 3.1: UML Activity Diagram for Mobile Application

Figure 3.1 shows that the proposed design of the mobile application satisfies the functional requirements as set out in section 3.1. Additionally from the diagram we can identify the two main components of the design which would need to be implemented in order to collect data in a way that will be useful to this study:

1. Crowd Sensing Component
2. Persistence Component

Crowd Sensing Component

The crowd sensing component includes all the parts of the application that are used to determine bus occupancy and passenger flow as well as location and cellular components.

Figure 3.1 demonstrates how the crowd sensing component of the application which includes the Bluetooth and Wi-Fi scanning happens concurrently. Concurrency in the application is necessary as the scan data needs to be collected from multiple sources at the same time in order to provide meaningful data to compare with the most recent ground truth value. For example it is desirable to know how many passengers are connected to Wi-Fi and how many discoverable Bluetooth devices are present at the same time and how any changes in one might affect the other. However concurrency will be a design challenge as our application will have to be programmed using threads which makes it more complicated to program, test, and debug for errors.

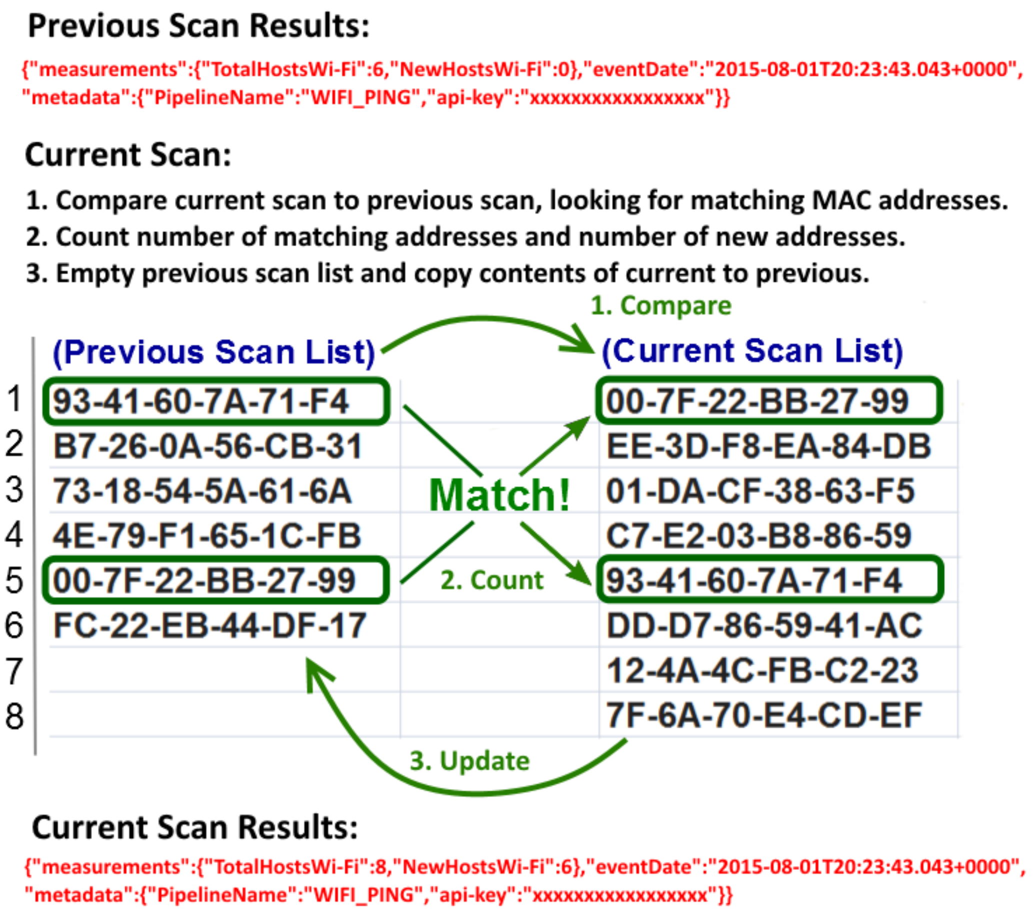
Crowd sensing in this application can be further broken down into inputs and functions. There are five inputs:

1. Location
2. Cellular Data
3. Bluetooth
4. Wi-Fi
5. Ground Truth

These inputs provide raw data, which may be noisy, and needs to be presented in both a meaningful and useful way to the persistence component in order to properly test our hypothesis. The functions of the crowd sensing component are to aggregate the data from each input and pass it to the persistence component.

As per our ethical considerations special care is needed in the crowd sensing component to ensure that no user or passenger could by identified through use of the application. For counting purposes however it is necessary to identify devices uniquely. The MAC address was chosen as the most reliable indicator of a unique device. IP addresses were considered but routers typically use NAT meaning that addresses are assigned to hosts as they come and go resulting in an IP address that may not be unique to a device during our scanning period. The negative of using a MAC address is that MAC addresses could be potentially used to track a persons device and so an algorithm must be designed to count the phones using their MAC addresses without the need to store the MAC address or reveal the address to the persistence component or any user in the system.

We designed an algorithm to be implemented on the inputs from both Bluetooth and Wi-Fi. Each detected device will have its MAC address added to a list called *current-scan-results* and when the scan is complete the contents of *current-scan-results* are moved in to a new list called *previous-scan-results* and *current-scan-results* is emptied. When the next scan concludes the results of *current-scan-results* are compared to *previous-scan-results* and the cycle continues. The algorithm will only return the total number of devices and the count of new devices.



Previous Scan Results:

{"measurements":{"TotalHostsWi-Fi":6,"NewHostsWi-Fi":0},"eventDate":"2015-08-01T20:23:43.043+0000",
"metadata":{"PipelineName":"WIFI_PING","api-key":"xxxxxxxxxxxxxxxx"}}

Current Scan:

1. Compare current scan to previous scan, looking for matching MAC addresses.
2. Count number of matching addresses and number of new addresses.
3. Empty previous scan list and copy contents of current to previous.

	(Previous Scan List)	(Current Scan List)
1	93-41-60-7A-71-F4	00-7F-22-BB-27-99
2	B7-26-0A-56-CB-31	EE-3D-F8-EA-84-DB
3	73-18-54-5A-61-6A	01-DA-CF-38-63-F5
4	4E-79-F1-65-1C-FB	C7-E2-03-B8-86-59
5	00-7F-22-BB-27-99	93-41-60-7A-71-F4
6	FC-22-EB-44-DF-17	DD-D7-86-59-41-AC
7		12-4A-4C-FB-C2-23
8		7F-6A-70-E4-CD-EF

Current Scan Results:

{"measurements":{"TotalHostsWi-Fi":8,"NewHostsWi-Fi":6},"eventDate":"2015-08-01T20:23:43.043+0000",
"metadata":{"PipelineName":"WIFI_PING","api-key":"xxxxxxxxxxxxxxxx"}}

Figure 3.2: Algorithm designed to protect passengers private data

By implementing the algorithm no MAC address is ever persistent in the application and the information that matters i.e. detecting passenger flow is still collected in a manner that is easy to work with and secure. This method satisfies both the ethical requirement in section 3.1.7 and the privacy requirement in section 3.1.7.

Persistence Component

The design of the persistence component is influenced by a number of different factors. Both non-functional requirements for passenger privacy and ethical concerns would dictate that the device cannot store any private data, as outlined in the previous section this data must be consumed then discarded.

In order to minimise read and write times to the devices database, which is important due to the concurrent processes that will require access to the database, there will be a single table where each input can store its data. This data can be then be related to the input that produced it by using a unique input name.

The persistence component will need to be able to send and receive data from a remote web service. To communicate with the web service it will be necessary to be able to package and parse data in a format that the web service understands. An internet connection to communicate with the web service will be required but this will be provided by the Wi-Fi on board the bus which the application should be connected to anyway.

3.2.2 Web Service

Web services have become an increasingly popular feature of web applications as they allow for easy exchange of data over the internet making them ideal for most kinds of Distributed system. Implementing a web service in this project has many benefits as those noted in section 3.1.6.

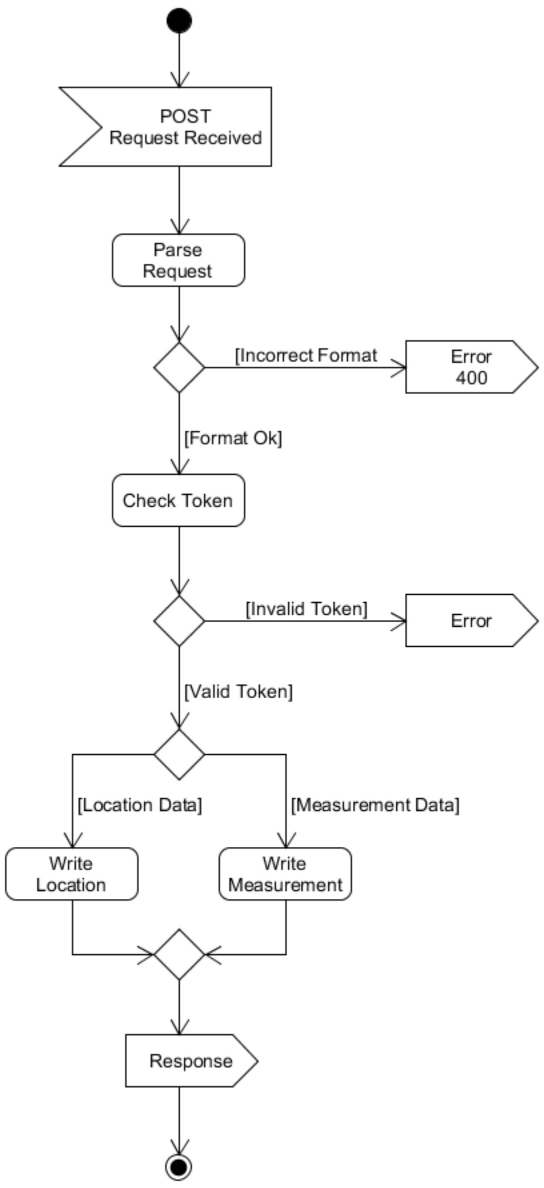


Figure 3.3: UML Activity Diagram for the Web Service

Web services typically receive HTTP requests and then generate a response after dealing with the request. Figure 3.3 shows that this web service will need to be able to parse post requests and determine if the data is valid. Invalid data will generate an error in the response while valid data will be written to a database and a success response will be sent. Another interesting feature of the design of this web service is the use of tokens. A token is a string that is unique to each device permitted to use the service. Any device that sends requests to the service will need to include their token otherwise the service will not process the request. Using a token helps to prevent accidental or malicious poising of data in the database.

Tokens will be generated by the web service and their value has no relation to any private user data. For security reasons it's important to authenticate devices sending data to the service however as per our ethical and privacy considerations it's also important not to identify the user as part of this authentication process.

3.3 Design Summary

In this chapter we introduced the design for PassengerSense, a crowd sensing mobile application designed to test our hypothesis by collecting data about Bluetooth and Wi-Fi usage on board of a bus and comparing these values with ground truth values provided by users of the application.

The proposed design satisfies all requirements both functional and non-functional and addresses the design challenges posed by the requirements. The proposed design has a number of merits including the loosely coupled web service which allows the application to scale comfortably even moving to different platforms. The design deals with security and privacy which are usually top concerns in crowd sensing applications. Redundancy is built into the design by replicating data sent to the web service in the devices local database, any data not sent to the web service can be queued and sent at a later time. Overall the design conceptualises a crowd sensing application capable of collecting the data required to test our hypothesis.

Chapter 4

Implementation

Following on from Chapter 3 where we outlined the design for our application this chapter aims to examine the implementation of that design. First we will introduce the chosen technologies and justify their inclusion. Next we will present the completed application that we developed as well as other resources related to the implementation of the application. Finally we will discuss the completed implementation outlining how it satisfies the motivations for this study and how it contributes to testing our hypothesis.

4.1 Chosen Technologies

PassengerSense, our mobile application, was implemented in Android using the Java programming language. Android is very suitable for this type of application because it features a rich SDK that allows us to access and manipulate all of the hardware features that we require on the phone such as GPS, Wi-Fi, and Blutooth. An Android operating system also allows users to install applications from any location which is very useful when asking volunteers to test the application.

Much work has previously been done in the area of Crowd Sensing using android devices, mobile crowd sensing is becoming an increasingly popular way of collecting data in urban environments due to the availability of mobile devices and the advancement of sensors and technology in those devices. The previous works were useful in both the design and implementation phases of PassengerSense as reference models but also for best practices for dealing with issues such as persistence, privacy, and battery usage. The previous works served as reassurance that this type of crowd sensing application was an achievable goal using Android Technology.

By default Android applications use an SQLite database for persistence. Although it is possible to use other databases SQLite is suitable for our requirements and is already set up on every Android device. In the following sections we will describe how PassengerSense uses the SQLite database.

JSON was chosen as a way to send and receive data from the RESTful web service that we will use to collect data from PassengerSense. RESTFul web services are becoming increasingly popular due to their relatively straight forward implementation compared to SOAP based web services. While both types of web services have their merits studies have shown that RESTful performs better in terms of network overhead [27]. The web service implementation will be described in greater detail in section 4.4. JSON is a great format for exchanging data across networks because it is human readable, lightweight compared to XML, and can be parsed easily in many different programming languages helping to keep components of the application loosely coupled.

4.2 Chosen Libraries

As a result of Android being a popular platform for research and development of crowd sensing as well as participatory and opportunistic sensing applications there are a number of good libraries available for developers to use. While developing PassengerSense some libraries and open source projects were integrated with the application to provide the most reliable implementation based on previous research. In the following sections we will examine these libraries and what role they played in the implementation of our application.

4.2.1 Mobile Sensing Technology (MoST)

Rather than re-invent a crowd sensing implementation in Android we decided to implement a mobile sensing library built for android devices called Mobile Sensing Technology (MoST) [6] [7]. MoST is an open source comprehensive crowd sensing library that is built to cater for new and old android devices making it suitable to use in a study where volunteers may use their own phones. The developers of MoST have researched an alternative to their library, the Google Play Services Library they found that while it had a lot to offer MoST performed very well and better in some cases [5].

The library introduces a list of input 'pipelines' where each pipeline provides raw data from the sensor to the devices SQLite Database. There are 22 pipelines in total for example light sensor, accelerometer, cell information, and battery information to name a few. Each pipeline can be can be activated and de-activated individually and runs in a background service.

The notion of Android Services are central to the implementation of our application. Android services represent some component such as a sensing activity e.g. scanning for Bluetooth devices that runs in the background without any user interface [14]. Using Services is an effective method of minimising disturbance to the user when running crowd sensing processes.

Integrating MoST into our application certainly saved a lot of development time as some of the pipelines contained exactley the data we needed. However MoST was not quite ready to use out of the box and was certainly a challenge to integrate as the lack of documentation on the library meant integration was quite time consuming. MoST itself used an older version of the Google Play Services Library which is no longer supported. This meant that any pipeline such as location that depended on deprecated libraries could not be implemented and in fact was easier to re-develop the pipeline from scratch due to the complex nature of MoST.

Overcoming the development and integration challenge, in our implementation of

MoST we would require only two pipelines to be used. Below we show what data these two pipelines originally collected and send to the database.
For each detected Bluetooth device:

Time Stamp A Unix time stamp.

MAC The detected devices Bluetooth MAC address.

Name The detected devices Name.

Class The class of detected devices Bluetooth radio.

For the nearest cellular tower:

Time Stamp A Unix time stamp.

GSM Cell ID A unique numerical identifier for the cell tower.

GSM LAC A unique numerical identifier for the location area of the cell tower.

Starting with the Bluetooth implementation, the way in which the data was collected was not suitable for our application. By default MoST stores each device it detects in a table row for detected bluetooth devices with the details of that device. Due to our Ethical requirements we could not store any data that we did not need or any data that could be considered private such as the devices MAC address. Next the cellular implementation does not detect RSSI and this was one of our requirements. Also by default MoST cannot POST its data to a remote server. Clearly MoST needs some modifications to be successfully implemented in our application.

Modifications to MoST

While MoST was a great starting point some modifications needed to be made to bring the library in line with some of our requirements from Chapter 3. This section will outline the modifications that were made to the MoST library during our implementation.

One of the significant changes we made to the MoST library was the ability to communicate with a RESTful web service. To implement this we created a new class that could be instantiated using the singleton pattern. This class accepted JSON strings and posted them to the RESTful web service in a process that was run in a separate thread. Using threads to communicate over the network is required in Android but also allows multiple calls to the same object to be queued in a thread pool which was advantageous for our implementation.

The way in which MoST stored data was not suitable for our ethical requirements but also would have generated a lot of network overhead when sending data to our web service due to the amount and frequency of data recorded. Initially each pipeline had its own table in the database. In our implementation there is a single table that keeps a copy of JSON data posted to the web service.

JSON strings needed to be stored in the database and then sent to the web service but by default MoST did not use JSON so each pipeline that we intended to use needed to be modified to produce JSON strings rather than the Content Values that it used by default. These JSON strings were then stored in the database and sent to an instance of our new class that posted them to the web service.

An algorithm was developed to reduce the network overhead and stop private data from being stored or transmitted to the web service. This algorithm was implemented within the MoST library for the scan results of Bluetooth. The algorithm as outlined in Chapter 3 section 3.2 takes the list of detected devices and compares them to the previous scan results checking for MAC address that match. The results of this algorithm show how many devices were and were not present in the previous scan allowing easy calculation of bus occupancy and flow without exposing private data. The Wi-Fi scanning process would need to use this algorithm but MoST does not have this function so we will discuss it in the next section.

Lastly while MoST did have pipeline for cellular data it did not capture the RSSI of nearby cell towers and this was one of our requirements. The pipeline for cellular data was modified to get the RSSI of nearby cell towers. RSSI or Received Signal Strength Indication was not straight forward to capture and this was made more difficult that the application needed to support phones from Android API 8 right up to the newest Android devices. RSSI can be read in a number of different ways depending on what type of network the user is connected to e.g. GSM, 3G, 4G LTE. This part of the application needs more work because phones connected to 4G/LTE networks often return 99 as an RSSI value meaning that signal is not available when in fact it is [12]. To solve this issue a method that could accurately detect the type of network the user was connected to and apply the appropriate sensing process would need to be developed unfortunatley we did not have enough time to address this issue properly but as a work around it was possible to disable the phones data when connected to the Buses wi-fi which forced the phone to use GSM and this in most cases gave an accurate RSSI value.

In summary MoST is a powerful and very rich mobile sensing library however as outlined in this section it did require significant modifications to be useful in our application. While the learning curve was initially steep to understand this complex library without documentation it was worthwhile because using this library saved time rather than trying to reinvent crowd sensing techniques that are tried and tested and already developed.

4.2.2 Network Discovery

As per our requirements we needed to be able to count the number of passengers connected to the buses Wi-Fi during each scan. Unfortunately this was not a feature that MoST could provide so we began to look for libraries that could. We decided to implement a heavily modified version of the open source Network Discovery application, available on the Google Play Store [18]. The source code for this application is made available on GitHub [19]

Network Discovery is a full Android application rather than a library so we only needed a few functions from the many it had. Most of the applications functions were striped away as we only required the ability to get a list of hosts connected to the wireless network.

In our implementation the network discovery process is run in a background service and is run in a loop with a five second delay after each scan until the user presses the stop button. The class responsible for scanning the network sends a ping message to each IP address in the routers range and listens for the reply which indicates there is a device at that address. The MAC address of the detected device is added to a list where it will be used in our algorithm, which is outlined in Chapter 3 section 3.2. This algorithm will tell us how many of the detected hosts were present in the last scan and how many are not. The algorithm is implemented when the Wi-Fi scanning process is complete and results are passed to the class that we created in the MoST library that handles communication with the web service and data persistence. Like with MoST we needed to change the data format to JSON.

To summarise our use of the Network Discovery application code I think it was good to find an open source project like this because it is not a straight forward task to implement on Android. Although our implementation does not resemble the original application at all it functions perfectly accurately counting the number of hosts connected to the buses Wi-Fi.

4.2.3 Location

To obtain the devices GPS position we used the standard Android LocationManager and created a background service that would get a GPS position when the location changed by more than 5 meters or 5 seconds had elapsed. These constraints stop constant updates to the web service if there is no change in sensed data.

We noted during testing that the older phones took longer to get a fix on their GPS positions but in general the location service class was efficient.

4.3 Application Overview

In this section we will describe the complete implementation of PassengerSense using screen shots of the user interface to demonstrate how the application appears to users as well as discussing the processes that run in the background. We will begin by examining the typical sequence of events that a user goes through when using the application. Next we will outline how the application appears to users and what features it has. Finally we will examine the data that is produced from using the application and outline how this will contribute to answering our research question.

4.3.1 Using the Application

In line with our ethical considerations participants using PassengerSense will have been briefed on what the application is doing, how to use it, and how to remove it if they want to stop the study. During the briefing the participant is instructed to take an initial count of people on the bus before starting the application. Due to the layout of a typical Double Deck Bus, as shown in figure 4.1, we found the easiest way to count passengers was to enter the bus and count downstairs before moving upstairs, counting the passengers there, and finally sitting in a place where you can then observe how many people are getting on and off the bus. Most of the Dublin Bus fleet only have one door so it is straight forward to keep track of passengers getting on and off the bus.

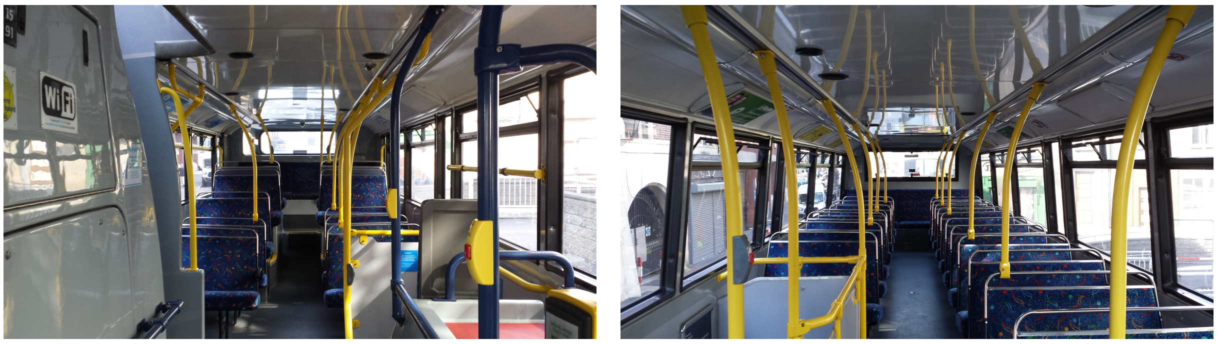


Figure 4.1: Typical layout of Dublin Bus. Left: Downstairs, Right: Upper Saloon

Next participants are instructed to connect to the Wi-Fi, provided by Dublin Bus. Once connected they can start the application. When the application is opened the user can select 'Begin Sensing', the user is then asked to input their current bus route as well as the ground truth value. The application then repeatedly scans for discoverable Bluetooth devices and devices connected to the buses network posting any changes to the web server. The user is asked to update the passenger count at each bus stop where there is a change in occupancy, each ground truth update is posted to the web service.

4.3.2 User Interface and Features

PassengerSense was designed to be as user friendly as possible in order to make it as easy as possible for our volunteer participants to use the application without problems. Figure 4.2 shows the three main screens that the application has.

Home Screen

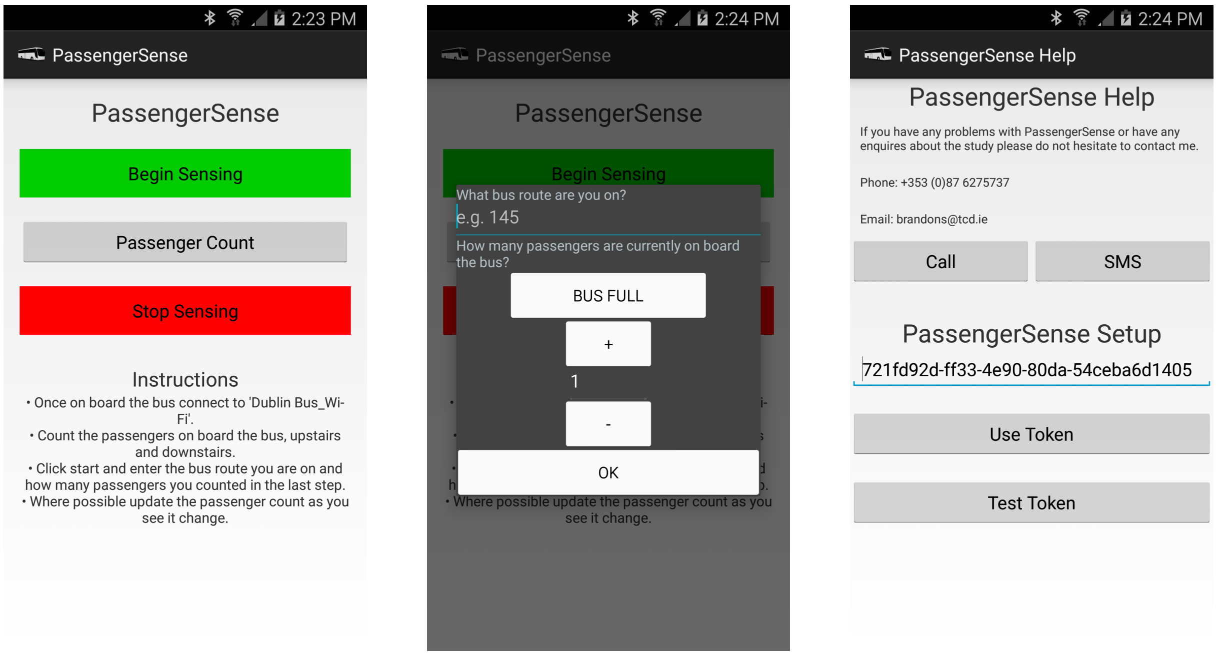
The first screen on the left is the applications home screen, this is what the user sees when they open the application. There are three buttons: "Start Sensing", "Passenger Count", and "Stop Sensing". Below the buttons there is a set of instructions for the user.

Ground Truth Dialog

The middle screen shows the "Ground Truth Dialog" this is where the user tells the application what bus route they are on and how many passengers are on the bus. This dialog is shown when the "Start Sensing" button is pressed. Users can bring the Ground Truth Dialog back up by pressing the "Passenger Count" button.

Help Screen

The screen on the right is the Help section. This screen has two functions; The first is to input the web service token, which is unique to each device, so that it can communicate with the web service. The token is saved using the Android "Shared Preferences" feature and can be tested with the "Test Token" button. The second function of the help screen is to provide a contact method for the participant to contact the lead researcher in the event they are having any problems. The second function is part of our ethical considerations.



Instructions

- Once on board the bus connect to 'Dublin Bus_Wi-Fi'.
- Count the passengers on board the bus, upstairs and downstairs.
- Click start and enter the bus route you are on and how many passengers you counted in the last step.
- Where possible update the passenger count as you see it change.

What bus route are you on?

e.g. 145

How many passengers are currently on board the bus?

If you have any problems with PassengerSense or have any enquires about the study please do not hesitate to contact me.

Phone: +353 (0)87 6275737

Email: brandons@tcd.ie

721fd92d-ff33-4e90-80da-54ceba6d1405

Figure 4.2: PassengerSense User Interface Screen Shots.

4.3.3 Data Collection/Persistence

Participants volunteered to have PassengerSense installed on their phones or used phones provided by the researchers. The participants were people who used Dublin Bus service regularly, usually for commuting to and from work. They agreed to use the application for a week each, in some cases participants were happy to use it for longer.

When PassengerSense is running it saves collected data in its local SQLite table as well as posting data to the web service. In the local database there is only a single table where the JSON strings are stored before being sent to the web service. During testing it was noted that if the application lost connectivity that the data was still recorded in the local database.

There are four types of JSON string that are created by the application. Each JSON string has a time stamp as well as an API key and Pipeline Name. The API key is generated by the web service when a new device is added. The key is an alphanumeric string that is unique to each device, without a valid key the web service will reject data sent to it. The Pipeline name indicates what sensor generated the data. An example of the four types of JSON strings have been included in Appendix B.

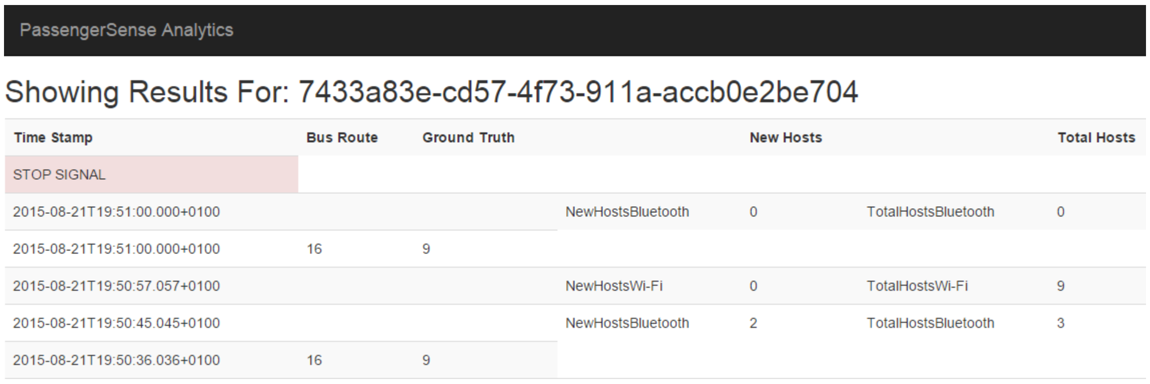
4.4 Web Service

Data collected by the application is sent to a RESTful web service. We were provided with the use of the Future Cities web service in Trinity College. This existing web service allowed us to set up a new project where our data would be stored. Once this project was set up, individual devices could be added to the project and the web service generated a unique alphanumeric key for each device.

The web service has an administrators user interface that can be used to add new devices and generate device keys as well as view the data that has been collected. The ability to log in remotely and view data without queries was very useful during testing of the mobile application.

Data is sent to the web service using a HTTP Post request where the contents is a JSON string. The web service uses a no SQL database, MongoDB, to store data. To retrieve our collected data from the web service it's possible to submit a query in a URL which returns raw JSON data.

In order to be able to understand and work with the raw JSON data we created a small web application using PHP that was able to parse the raw JSON data into much more user friendly tables. The data in these tables could be filtered which was helpful during our evaluation.


PassengerSense Analytics

Showing Results For: 7433a83e-cd57-4f73-911a-accb0e2be704

Time Stamp	Bus Route	Ground Truth		New Hosts		Total Hosts
STOP SIGNAL						
2015-08-21T19:51:00.000+0100			NewHostsBluetooth	0	TotalHostsBluetooth	0
2015-08-21T19:51:00.000+0100	16	9				
2015-08-21T19:50:57.057+0100			NewHostsWi-Fi	0	TotalHostsWi-Fi	9
2015-08-21T19:50:45.045+0100			NewHostsBluetooth	2	TotalHostsBluetooth	3
2015-08-21T19:50:36.036+0100	16	9				

Figure 4.3: Web application to query and display PassengerSense data.

Figure 4.3 shows a screen shot from our web application. To address our research question it is not necessary to look at the location or cellular data so our web service excludes this which makes the data a lot easier to work with. The web application outlines bus journeys by displaying a green start tag and red stop tag which indicates a full bus journey. Ground truth data is displayed separate to data collected through opportunistic sensing methods. This way of organising our data is important as it will be nesecary to identify noisy data during our evaluation. Noisy data could include detecting Bluetooth devices not on the bus i.e. they will only appear for a couple of seconds.

4.5 Ethical Implementation

Throughout the design and implementation chapters we have referred to features and functions that have been developed to satisfy our ethical requirements such as the help page on the application, outlined in section 4.3.2 and the algorithm designed to protect passenger privacy as described in section 3.2.1.

While these features ensured that the application itself was in keeping with ethical guidelines there are additional requirements to satisfy. Each participant is briefed about the study and provided with a participant information sheet. Once the participant is happy to contribute to the study they sign a consent form which they receive a copy of. A copy of the participant information sheet can be found in Appendix C.2.

Realising that participants may attract attention while trying to count other passengers on the bus they are provided with information leaflets to give to enquiring members of the public. A copy of this information leaflet can be found in Appendix C.1. The information leaflet was designed to inform members of the public in plain English what the application is doing and what data it collects. The leaflet has an email address for the lead researcher if the reader has further questions and there is web site address where the user can go to the project blog for more information.

In addition to the information leaflet we created a blog where the project is explained in greater detail. From this page there are links to copies of all the ethical resources as well as the ethical application submitted to the Ethical Committee. There is also a link where users can view the source code of PassengerSense on GitHub.

4.6 Implementation Summary

In this chapter we have outlined how we choose android libraries and dependencies to include in our application as well as detailing their implementation. Much of the difficulty in developing our application came from the modifications to these libraries and dependencies that were necessary to satisfy our requirements as outlined in the previous chapter.

Next we introduced the developed application demonstrating what the participant will have to do to use the application before showing screen shots depicting how the application appears to participants. The RESTful web service that collects data from the mobile application is introduced and the JSON format for communication with the application is explained. We also introduce our own web application that will help evaluate the results and manage queries to the web service.

Finally we discussed additional resources that were needed to fully implement the application such as the ethical documentation and web site. To summarise the implementation we are satisfied that the developed application is fit for purpose as a means of collecting data to help prove or disprove our hypothesis. We are also satisfied that the implemented application satisfies the requirements set out in the previous chapter.

Chapter 5

Evaluation

In the previous chapter we examined the implementation of our design for a crowdsensing application that was used to collect data which will test our hypothesis. In this chapter we will evaluate the performance of this application as well as analyse the data that was collected. We will begin by examining our collected data set and using statistical analysis to see how this data set can test our hypothesis. Next we will discuss any issues we had with the data set before concluding with a discussion on how our data could be affected by demographics.

5.1 Quantitative Evaluation

Our mobile application was designed to collect a data set that could be used to determine the answer to our research question. In this section we will statistically evaluate our data set to test our hypothesis and evaluate our results.

Our data set was collected by five volunteers who used the application during their normal commute for a period of seven days each. A summary of the data collected by the volunteers is outlined below in Table 5.1.

5	Volunteers
7	Bus Routes
31	Individual Journeys
368	Ground Truth Samples

Table 5.1: Data Set Summary

The collected GPS data made it possible to map individual bus journeys which was also useful as a way of confirming that the values are from a bus journey and that the application is working correctly. Figure 5.1 shows the GPS traces collected from two participants on two different bus routes. Using the Future Cities web service each bus journey could be displayed in this way.

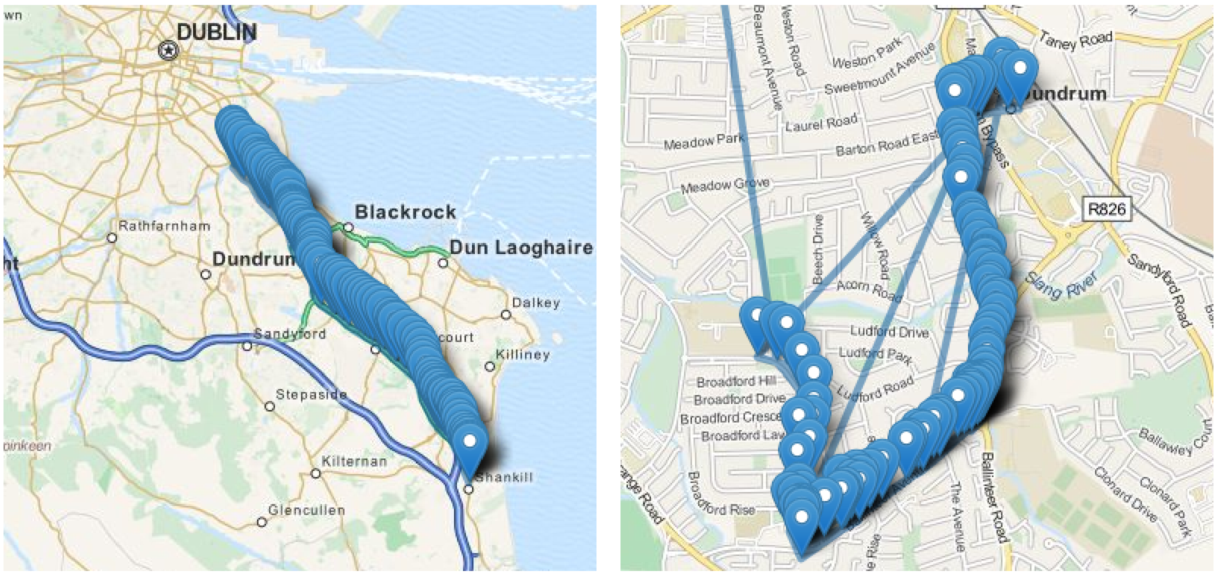


Figure 5.1: GPS traces from participants phones

5.1.1 Correlation

Our hypothesis seeks to establish a relationship between the number of Wi-Fi users, discoverable Bluetooth devices, and the actual number of passengers on board a bus. To investigate this relationship we begin by using statistical correlation to determine how strong the relationship will be if one exists.

To calculate the correlation we used the Microsoft Excel "Correl" function. Excel uses The Pearson Product-Moment Correlation Coefficient [26]. This formula measures the linear correlation or dependence between two variables and returns a value between +1 and -1. These values indicate the strength of the correlation where 0 means no relationship exists. Below is the actual equation used by Excel:

$$Correl(X,Y) = \frac{\Sigma(x-\bar{x})(y-\bar{y})}{\sqrt{\Sigma(x-\bar{x})^2\Sigma(y-\bar{y})^2}}$$

5.1.2 Correlation Results

When we correlated all of our sample ground truth values with the opportunistic data we collected we found that there is a relationship between the number of Wi-Fi users and the actual number of passengers on board the bus. However the same is not true for Bluetooth. The relationship between Bluetooth and the ground truth was negligible. Table 5.2 shows the results of our correlation.

Correlation	**R**	**Rˆ2**
Ground Truth / Wi-Fi Users	0.614	0.377
Ground Truth / Bluetooth Detected	0.024	0.000
Wi-Fi Users / Bluetooth Detected	-0.015	0.000

Table 5.2: Correlation Results (All Data)

The results of our correlation (R) in table 5.2 shows a reasonably strong positive relationship between the number of Wi-Fi users and the observed ground truth (0.614). Having established that a relationship exits we then used the coefficient of determination (Rˆ2) to determine the strength of that relationship. The Rˆ2 value of 0.377 suggests that the correlation relationship is not a good fit for our statistical model or that much of our data cannot be explained by this relationship. The value for Rˆ2 indicates that this relationship would not be accurate method of estimating bus occupancy due to the unknown part of the relationship.

We believed that this relationship would be stronger during rush hour as our data showed more devices than average connected to the buses Wi-Fi during rush hour, even when buses were not full. we reran the correlation tests using rush hour data only. This meant selecting all bus journeys that took place between 07:00 - 09:00 and 16:00 - 19:00. Our data set contained 123 ground truth samples within these times with ranging occupancies from only 9 people to full buses. The decision to isolate rush hour data was based on an assumption that rush hour commuters are more likely to have smart phones and use them during their commute. Table 5.3 shows the results of correlating our rush hour only data.

Correlation	**R**	**R^2**	**R^2 Change**
Ground Truth / Wi-Fi Users	0.796	0.634	+0.258
Ground Truth / Bluetooth Detected	0.062	0.004	+0.004
Wi-Fi Users / Bluetooth Detected	0.025	0.001	+0.001

Table 5.3: Correlation Results (Rush Hour Data)

The "Change" column was added to table 5.3 to show how the coefficient of determination value increased for each test when rush hour only data was used as opposed to our entire data set in table 5.2. The most significant change in the results is the increase in correlation between Wi-Fi users and observed ground truth. The R^2 value has increased to 0.634 which means that the relationship is much stronger when rush hour only data is observed. The relationship between Bluetooth and observed ground truth remains too weak to be useful for predicting bus occupancy.

Despite finding that during rush hour the relationship between Wi-Fi users and observed ground truth is much stronger this relationship is still too weak to make accurate estimates of bus occupancy. Initially it was hoped that by collecting both Wi-Fi and Bluetooth one of the results could act as a validation metric for the other but this turned out not to be possible due to the weak relationship we found between discoverable Bluetooth devices and observed ground truth.

5.1.3 Correlation Findings

Calculating the correlation between our detected Bluetooth devices, Wi-Fi users, and observed ground truth we found that while a relationship existed only between Wi-Fi users and observed ground truth values it was too weak to make reliable estimates about bus occupancy based on Wi-Fi data.

Next we looked at our rush hour only data and ran the same correlation tests which revealed that our discovered relationship between Wi-Fi users and observed ground truth became stronger during rush hour. This result indicated that demographics play a role in this relationship, we will discuss how demographics could affect our results later in this chapter in section 5.2.2. Despite the stronger correlation the relationship remained too weak for making reliable estimates.

Following the results of our correlation tests we continued to look for ways that our data set could be useful in estimating bus occupancy, we now turn our attention to Binary Classification.

5.1.4 Binary Classification

A binary classification task involves separating data into two distinct categories based on features of that data. Binary Classification is commonly used in medical statistics to determine if a patient has a disease or not based on symptoms they display. In this study binary classification will be used to determine if a bus is full or not based on the number of people connected to the buses Wi-Fi.

For the purposes of our binary classification model we will say that a bus is full when $\geq 75\%$ of seats are occupied. Most of the double deck Dublin Bus fleet have 76 seats [11]. We choose this value of 75% because during rush hour buses can fill up fast so any passengers enquiring about bus occupancy could know how likely they were to get a seat on this bus or not.

There are a number of different methods available for carrying out binary classification and there is also much debate as to which is the most accurate. In creating a binary classification model for our data we choose to use logistic regression. We made this decision based on the resources and support that were available for logistic regression calculation.

Logistic regression measures the relationship between a dependent variable and one or more independent variables by estimating probabilities that the independent variable and dependent variable(s) are in some way related. These probability values allow us to create a classification model where we can use probability to determine if a bus is full or not. In our case the dependent variable is our observed ground truth and our independent variable is our corresponding count of Wi-Fi users.

Adding Bluetooth as a secondary independent value had a negative effect on the accuracy of our classification model so it was decided to omit Bluetooth altogether for our binary classification evaluations.

Logistic regression calculations were carried out using the "Real Statistics Resource Pack" for Microsoft Excel [8].

5.1.5 Binary Classification Approach

As in section 5.1.1 we will evaluate two binary classification models one using all of our data and the other using rush hour data only. To represent our binary classification as well as evaluate its accuracy we will create classification tables and examine the (ROC) Receiver Operating Characteristic.

A classification table is a way of evaluating a binary classification model. The table represents the four possible outcomes of binary classification. Where P is positive and N is negative the outcomes can be:

True Positive: actual value: P - prediction: P.

False Positive: actual value: N - prediction P.

True Negative: actual value: N - prediction N.

False Negative: actual value: P - prediction N.

In our model we will look for accurate values for True Positives indicating the model has predicted the bus is full and it is full or True Negatives which indicate our model has predicted the bus is not full and it is not full. The classification table will also be used to calculate the accuracy of our classifier which will return a value between 0 and 1 where 1 is 100% accurate.

To further evaluate our results we will look at the receiver operating characteristic. The receiver operating characteristic will show the performance of our binary classification model by plotting the true positive rate against the false positive rate in a scatter chart.

5.1.6 Binary Classification Results

Beginning with all of our collected data, Table 5.4 shows the classification table. The table represents the observed outcome i.e. if the bus was actually full or not against the predicted outcome i.e if the classifier estimated that the bus was full or not.

	Full Obs	**Not Full Obs**	
Full Predicted	11	4	15
Not Full Predicted	41	311	352
	52	315	367
Accuracy	0.211538	0.987302	0.877384

Table 5.4: Classification Table (All Data)

The classification table shows that our classifier can accurately differentiate between a bus that is full and one that is not. Our Accuracy calculations show that the classifier can more accurately determine when a bus is not full when compared to one that is. Overall our accuracy for this classifier is high enough to be useful in estimating if a bus is full or not.

Running the same example with only rush hour data we produced the following classification table:

	Full Obs	**Not Full Obs**	
Full Predicted	15	5	20
Not Full Predicted	10	92	102
	25	97	122
Accuracy	0.6	0.948454	0.877049

Table 5.5: Classification Table (Rush Hour)

Interestingly table 5.5 shows that the accuracy for being able to determine if a bus is full or not has not changed by much using only the rush hour data. This may be because the entire data set also contains the rush hour data.

Further analysis of our binary classification models reveal the differences between the two data sets. Examining the Receiver Operating Characteristic plot chart in figure 5.2 we observe that the rush hour data has a higher sensitivity meaning its values come closer to (0,1) which is known as perfect classification. In figure 5.2 the X axis represents the false positive rate while the Y axis represents the true positive rate.

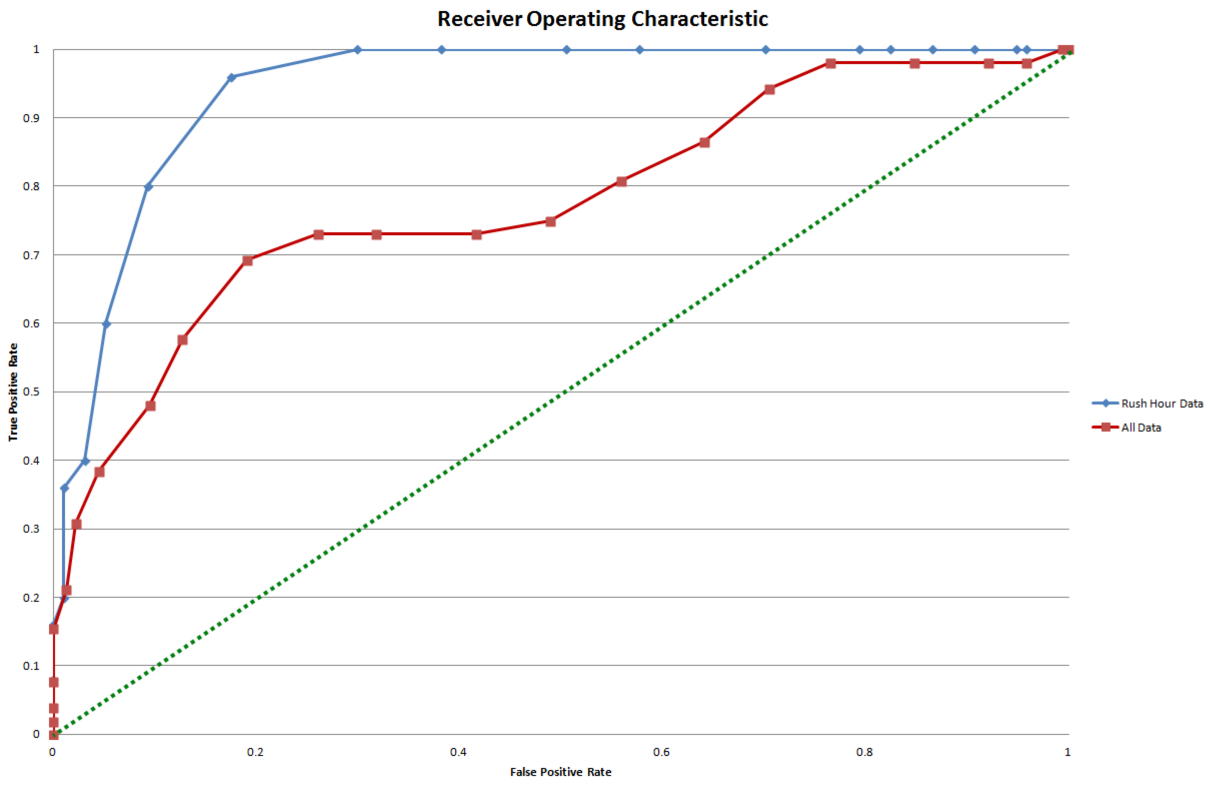


Figure 5.2: Receiver Operating Characteristic.

The reason for the greater sensitivity in rush hour data is most likely due to the demographics of passengers during rush hour. We established that the sensitivity is not simply caused by buses always being full during rush hour because the average occupancy for our 123 ground truth samples during rush hour is 40 passengers which is well below the 57 or 75% level that we consider full.

We have included the table used to carry out the logistic regression calculations for our entire data set in Appendix E which shows how the classification tables were determined. The receiver operating characteristic analysis is also included in this appendix and includes calculations for the area under curve (AUC).

5.1.7 Binary Classification Findings

Evaluating the results from our binary classification model we find that it is possible to estimate if a bus is full or not with close to 88% accuracy. It was also found that including the corresponding discoverable Bluetooth device count as another independent variable in our binary classification model had a negative effect on the accuracy of the overall system and for that reason it was omitted.

Finding that Bluetooth has a negligible relationship with our observed ground truth is an important finding because this goes against our initial belief derived from our state of the art where it was suggested that Bluetooth represents a strong relationship to population. It is worth noting that some of the studies citing that Bluetooth represents a portion of the population are up to 6 years old. Bluetooth may not be as widely used today as it was at the time of those studies, also for security many devices will not stay in Bluetooth discoverable mode for more than two minutes. From conversations with our participants appears that it is common practice to disable Bluetooth to prevent it from using battery.

It would appear that the use of binary classification is a good solution to this problem as most passengers just want to know if there is a seat available for them, not how many people are on the bus. For the bus operators they will want to know where and when there buses are full and empty. Binary classification can be used to answer all of these questions however I feel that more work is needed with a larger data set to properly evaluate if this is a problem that can be solved by crowdsensing alone.

5.2 Qualitative Evaluation

In this section we will evaluate our results from a qualitative point of view. First we will outline what noise we found to exist in our data set, how this occurred and how we dealt with it as well as any issues that arose as a result. Next we will discuss other factors that could have influenced our data.

5.2.1 Removing Noise From Data Set

Our mobile application continuously scanned for discoverable Bluetooth devices as well as the number of devices connected to the buses router while the user occasionally updated the ground truth value. This creates two main sources of noise in the data set. The first is that it was possible for the application to detect Bluetooth devices that were not on the bus. For example if the bus was stopped at a busy city centre junction the application may detect Bluetooth from pedestrians.

To deal with this noise in the data our privacy algorithm could determine how many devices were in our last scan which makes it easier to identify Bluetooth devices that come and go quickly by using the timestamp.

While our privacy algorithm, as described in section 3.2.1, is great for protecting passenger privacy it does prevent us from being able to identify individual hosts in the data set and this is a problem when trying to remove noisy data. In some instances it was easy to identify devices that were on the bus and in other cases it was more difficult.

As a solution to this problem we would recommend that our application temporarily hold detected MAC addresses for the duration of the bus journey rather than just one scan after they are detected as implemented in our application. This means for each scan it would be possible to identify devices that were on the bus at any time during the journey as opposed to devices that were detected in passing. The MAC addresses could be all removed once the user pressed stop.

Another source of noise in the data was produced by Wi-Fi devices connecting to the buses router that were not on the bus. For example a pedestrian that has "Dublin-Bus Wi-Fi" saved on their device will automatically connect to the bus if they are standing beside it for any length of time. The same solution suggested for removing noise from the Bluetooth data will work as a solution to this problem as both the Wi-Fi and Bluetooth use the same privacy algorithm to protect passenger privacy. However it was much easier to identify these extra Wi-Fi deivces as noise because usually more than one device would connect at a time and this made it easy to identify spikes in the data set that quickly disappeared.

Noise in the data set is an unavoidable problem but the process to identify and remove this data could be better. We don't believe that it had a serious effect on the quality of our data as it was possible to identify and remove data that did not belong in the data set. The process of identifying and removing noisy data was time consuming and the solution to keep mac addresses for the duration of the journey would make this process much quicker meaning it would be possible to deal with a bigger data set.

5.2.2 Discussion

During the study we observed different ways in which our detected data could be affected. The most obvious of these is demographics. For instance young children and elderly people are more unlikely to have a smart than the average person who uses a rush hour bus. If many seats on the bus are taken by people who don't have smart phones then they become more difficult to detect from a corwdsensing point of view.

Another trend that weaken the relationship between Wi-Fi users and passengers is the prevalence of 3G/4G data. Mobile data is becoming faster, more reliable, and cheaper meaning people may be less likely to connect to free Wi-Fi provided by bus operators as it may not be as good as their mobile data connections.

Three out of our five volunteers noted occasions where the Wi-Fi was not working on board the bus. This study took place over just seven days which suggests this problem could be quite common. On a bus where Wi-Fi was not working it would be impossible to make estimates on occupancy using our application.

5.3 Evaluation Summary

In this chapter we evaluated our data set which was created by a team of five volunteers who used our application during their commute for seven days each. Analysing this data we found that there is a relationship between the number of Wi-Fi users and the number of passengers on board a bus. This relationship becomes stronger during rush hour. We also found that there is no meaningful relationship between the number of Bluetooth devices and number of passengers.

Next we created a binary classification model through logistic regression to determine if our data set could be used to determine if a bus is full or not. We observed that our classifier was able to accurately predict if a bus was full or not in almost 88% of cases.

While we established that more work could be done to improve the accuracy of our classifier our findings support our hypothesis that there is a relationship between the number of Wi-Fi users and passengers on board. Discovering that Bluetooth has no relationship to the passenger count is a useful contribution as this goes against what was seen in our State of the Art in Chapter 2.

Finally we discussed how the data collected by our application had noise and how we dealt with this problem. A solution for dealing with this noise in future work was presented. We concluded our evaluation chapter by discussing the different ways in which our findings could be manipulated by demographics for example.

Chapter 6

Conclusions

In the previous chapter we evaluated the data set that was created by our volunteers using the crowdsensing application that we implemented and how this data supports our hypothesis. In this chapter we will summarise our project and our results as well as discussing future work.

6.1 Project Overview

This project aimed to provide an alternative approach to estimating passenger flow and occupancy on board buses. Many of the existing solutions to this problem involve installing hardware on the buses which can be costly to implement and maintain. We aimed to establish if a relationship could be drawn between Wi-Fi or Bluetooth and the actual number of passengers on board. Establishing a significantly strong relationship would allow accurate estimates to be made in real time about bus occupancy through a crowdsensing application.

In order to test our hypothesis it was necessary to collect data about bus passengers and the WI-Fi and Bluetooth usage in a bus environment. We designed and implemented a crowdsensing application that could continuously scan for discoverable Bluetooth devices as well as scanning to count how many devices were connected to the buses router.

The application also collects the actual passenger count or ground truth. The ground truth value is updated by the applications user when the bus occupancy changes. Our application posts the observed data back to a RESTful web service where it can be queried for processing.

We assembled a small team of volunteers who agreed to use our application during their commute for a period of one week. These volunteers would activate the application once on board their bus and do a head count and submit the ground truth values.

Next we created a small web application to help analyse and filter the data collected by our volunteers. This web application helped to remove any noise from our data set.

Finally we evaluated our data set by first examining if there was any correlation between our collected Bluetooth and Wi-Fi results and the ground truth samples. Having found a relationship between Wi-Fi users and our ground truth samples we went on to develop a binary classification model that could estimate if a bus was full or not based on the number of Wi-Fi users on board. We found our binary classification model to be accurate in almost 88% of cases that we tested.

6.2 Contribution

Our project introduces a novel approach to estimating passenger flow and occupancy on board public transport buses. Our approach is validated through the data we collected which shows that there is a relationship between the number of Wi-Fi users and the number of passengers on board a bus. Further to establishing this relationship we find that there is no meaningful relationship between discoverable Bluetooth and the number of passengers which could be useful information for anyone considering future work in this area.

We offer solutions to the problem of maintaining passenger privacy while still being able to uniquely identify devices. This is a trade off that can make it difficult to filter the noise from a data set if it is not dealt with.

Finally we collected GPS and corresponding RSSI data which can be used in future projects to determine if it is possible to infer the bus route from RSSI which would be a more energy efficient solution than using GPS.

6.3 Future Work

While this project sets a good foundation for estimating passenger flow and occupancy through crowdsensing there is still plenty of room for improvement. Initially the idea behind collecting both Bluetooth and Wi-Fi data was that one could act as validation metric for the other increasing the accuracy of our classifier.

Future work could be done to introduce another independent variable that would help validate the Wi-Fi relationship and increase the accuracy of our classifier. Wi-Fi probing signals are emitted by smart phones that have their Wi-Fi radios turned on. These probing signals are looking for familiar wireless networks that the device can connect to. The probing signals contain the sending devices MAC address which makes it possible to uniquely identify devices for counting purposes. Its possible that there is a strong relationship between Wi-Fi probing signals and passengers because even when passengers don't connect to the Wi-Fi on board the bus but simply use mobile data, their device will still continuously emit a probing signal.

This project did not look at determining what bus the user was on other than collecting the route number directly from the user. Future work could look at integrating passenger occupancy and flow applications with a bus scheduling application creating a powerful crowdsensing application. This would allow passengers to know when the bus is coming and if there will be a seat available for them.

Making the application collaborative i.e. if more than one application user is on the same bus both of the applications can communicate in order to make more accurate estimations. Collaboration could be used to infer bus occupancy and also what route the application is on. A collaborative function would help the applications data become more reliable as well as more energy efficient as it may be possible to share crowdsensing tasks between multiple clients on the same bus.

If the application had the ability to differentiate between different modes of transport such as the bus, train, or a car using its accelerometer it would not be necessary to ask the user when they get on and off the bus. Future work could include creating this feature which would make the application much less intrusive for the user.

Appendix A

Abbreviations

Short Term	**Expanded Term**
RTPI	Real Time Passenger Information
GPS	Global Positioning System
SDK	Software Development Kit
RFID	Radio Frequency Identification
NAT	Network Address Translation
MAC	Media Access Control
RSSI	Received Signal Strength Indication
UML	Unified Modelling Language
HTTP	Hyper Text Transfer Protocol
MoST	Mobile Sensing Technology
JSON	JavaScript Object Notation
XML	Extensible Markup Language
LAC	Location Area Code
GSM	Global System for Mobile Communications
LTE	Long Term Evolution
SQL	Structured Query Language
ROC	Receiver Operating Characteristic

}

Appendix B

JSON Data

The following are examples of JSON strings stored by the application and sent to the web service.

B.1 Wi-Fi Count

```
{
        "measurements":
        {
                "TotalHostsWi-Fi":7,
                "NewHostsWi-Fi"5
        },
        "eventDate":"2015-08-01T20:23:43.043+0000",
        "metadata":
        {
                "PipelineName":"WIFI_PING",
                "api-key":"xxxxxxxxxxxxxxxxx"
        }
}
```

B.2 Discoverable Bluetooth Devices Count

```
{
        "measurements":
        {
                "TotalHostsBluetooth":0,
                "NewHostsBluetooth":0
        },
        "eventDate":"2015-08-01T20:23:38.038+0000",
        "metadata":
        {
                "PipelineName":"BLUETOOTH",
                "api-key":"xxxxxxxxxxxxxxxxx"
        }
}
```

B.3 Nearby Cellular Data

```
{
        "measurements":
        {
                "rssiTo15813":-63,
                "rssiTo15430":-85,
                "rssiTo36364":-77,
                "rssiTo65535":-93,
                "rssiTo15767":-91
        },
        "eventDate":"2015-08-01T20:23:41.041+0000",
        "metadata":
```

```
        {
                "PipelineName":"CELL",
                "api-key":"xxxxxxxxxxxxxxxxx"
        }
}
```

B.4 GPS Location

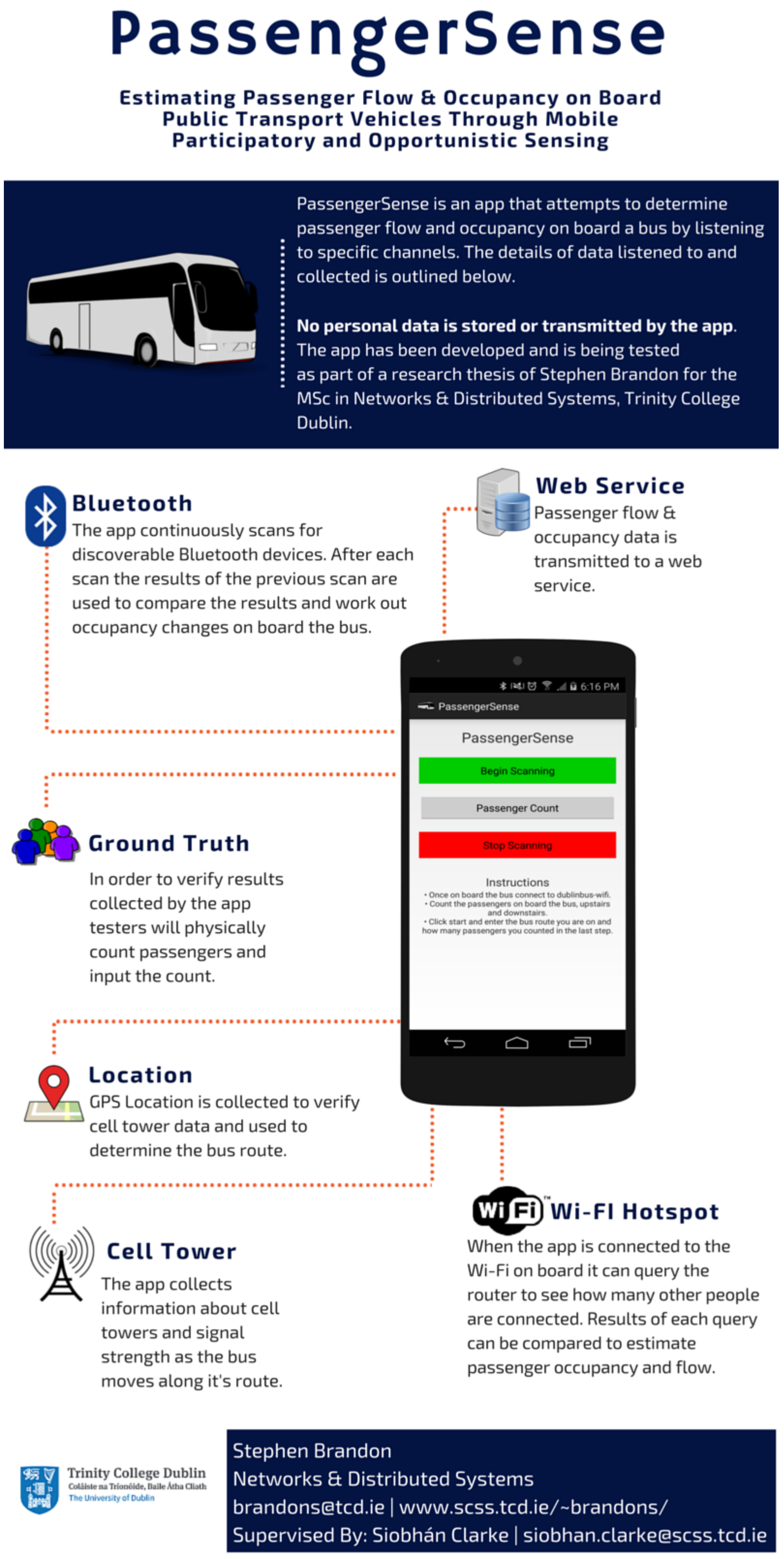
```
{
        "elevation":0,
        "longitude":-6.2090056,
        "latitude":53.304245,
        "eventDate":"2015-08-01T20:23:38.038+0000",
        "metadata":
        {
                "api-key":"xxxxxxxxxxxxxxxxx",
                "PipelineName":"LOCATION"
        }
}
```

Appendix C

Participant Information

C.1 Public Information Leaflet

This is a copy of the information leaflet that is supplied to participants to give to members of the public that may enquire about the application. The leaflet aims to inform the reader about what the application is doing and why it is doing it. There is contact information for the lead researcher as well as a web site address where more information is available.


PassengerSense

Estimating Passenger Flow & Occupancy on Board Public Transport Vehicles Through Mobile Participatory and Opportunistic Sensing

PassengerSense is an app that attempts to determine passenger flow and occupancy on board a bus by listening to specific channels. The details of data listened to and collected is outlined below.

No personal data is stored or transmitted by the app. The app has been developed and is being tested as part of a research thesis of Stephen Brandon for the MSc in Networks & Distributed Systems, Trinity College Dublin.

Bluetooth
The app continuously scans for discoverable Bluetooth devices. After each scan the results of the previous scan are used to compare the results and work out occupancy changes on board the bus.

Web Service
Passenger flow & occupancy data is transmitted to a web service.

Instructions
• Once on board the bus connect to dublinbus-wifi.
• Count the passengers on board the bus, upstairs and downstairs.
• Click start and enter the bus route you are on and how many passengers you counted in the last step.

Ground Truth
In order to verify results collected by the app testers will physically count passengers and input the count.

Location
GPS Location is collected to verify cell tower data and used to determine the bus route.

Cell Tower
The app collects information about cell towers and signal strength as the bus moves along it's route.

Wi-FI Hotspot
When the app is connected to the Wi-Fi on board it can query the router to see how many other people are connected. Results of each query can be compared to estimate passenger occupancy and flow.

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Stephen Brandon
Networks & Distributed Systems
brandons@tcd.ie | www.scss.tcd.ie/~brandons/
Supervised By: Siobhán Clarke | siobhan.clarke@scss.tcd.ie

Figure C.1: Copy of Information Leaflet Distributed to Participants

C.2 Participant Information Sheet

This is a copy of the information sheet that participants were given to read before they signed the consent form. The information sheet outlines what is expected of volunteers and what they can expect from the study.

C.2.1 Information Sheet for Prospective Participants

Background

Data about passenger usage of public transport is useful for both the passengers and the companies providing the transport. It is useful for passengers to know how full a bus or train will be before they board it. Companies providing the transport can use the same data to plan best use of their resources and minimise congestion which is uncomfortable for passengers.

Getting this type of passenger data usually involves using hardware which can be expensive for the bus companies to install and implement across their fleet. The data collected by this hardware is seldom made available to passengers.

In this study we aim to draw a relationship between bus occupancy and smart phone activity on board the bus. To do this we have developed an application that observes the environment on the bus using methods known as participatory and opportunistic sensing.

Participatory sensing describes a system where participants provide some sort of data. In this study participants will be required to count passengers on board the bus and observe passenger flow during their journey. This data will input into the mobile application and will create the ground truth or real data for comparison.

Opportunistic sensing describes a system where data is collected as it becomes available without any input needed from participants. In this study the mobile application will detect Wi-Fi, Cell Network, and Bluetooth activity on board the bus and this data will be used to estimate occupancy and passenger flow.

Once data has been collected from passengers on different bus routes at different times we hope to draw a relationship between the ground truth values collected by the participant and the data collected through opportunistic sensing. If a relationship exists passenger occupancy and flow can be estimated by observing smart phone activity on board public transport vehicles.

Procedure

Each participant will have an application installed on their phone and be briefed on how to use it and remove it. The application will be capable of collecting the following type of data once the participant is on a bus. Note that the application can only collect data once activated by the participant on board the bus and can be deactivated at any time preventing it from collecting further data.

The application collects data from the buses Wi-Fi network. The list of other devices connected to that network with their details are detected. The application scans for discoverable Bluetooth devices and temporarily records their details. Location data is collected from the participants phone. Cellular data is collected from the participants phone such as signal strength and details of the nearest cellular tower. Details about the battery of the participants device will also be collected in order to make the app more efficient.

The MAC addresses of other devices are detected and temporarily used in an algorithm for the duration of the next scan. MAC addresses are then compared to the previous scan to count how many new devices or on the bus or how many have left. The count of current devices and new devices is timestamped and sent to the server anonymously. No individual participant or passenger will be identified or have personal details transmitted to the server or stored on any device. No participant or passenger details shall be included in publications associated with this study.

Participants

Participants should be aware that their participation in this study is entirely voluntary and that they are free opt out at any time without reason or penalty. In the event a participant wants to opt out they may simply remove the application from their phone. Participants will be shown how to remove the application during their briefing.

The study will take two weeks to complete however participants are not obliged to participate for the full duration of the study. Participants will only need to contribute when on board a bus and if they want to do so.

Risks

It is possible that a participant may attract attention while conducting a head count or using the application on board the bus. Participants will be provided with access to information material regarding the application/study that may be distributed to the general public.
This information material is available on the lead researchers website.
`https://www.scss.tcd.ie/~brandons/`

- An unsigned copy of this consent form and participant information sheet.
- An information leaflet designed to give members of the public an understanding of what the application and study are aiming to achieve. This information leaflet contains lead researchers email address to answer any queries.
- A copy of the research project proposal as submitted to the Ethics Committee at the School of Computer Science & Statistics, Trinity College Dublin.

In addition to the material that will be available online, as mentioned above, each participant will be provided with:

- 10 hard copies of the information leaflet to distribute to the general public (additional copies can be made available).

- 1 hard copy of their signed consent form and participant information leaflet.
- 1 hard copy of the research project proposal.

No part of this application/study is private and no data that could relate to an individual person or participant is collected or held on any device. Only publically available data is detected by the application and there is no monitoring of individuals or their devices.

Contact information for the lead researcher is clearly marked on the information leaflet. Participant safety is the highest priority, any participant who feels they have attracted negative attention should immediately stop the study and contact the lead researcher as soon as possible.

If a participant is confronted about their actions they should explain to the person what they are doing and provide them with the information leaflet. If the person is persistent or asking questions of a technical nature that the participant does not understand the participant may refer the person to the lead researcher using the contact information provided.

If a person on the bus objects to having their presence counted the participant should stop the study on board that bus, demonstrating to the person how the application has stopped.

If a participant is confronted in an aggressive or threatening manner they should immediately stop the study and take steps to ensure their own personal safety.

The lead researcher will be contactable by phone at all times during the study. If a participant feels uncomfortable at any time for any reason during the experiment they should stop and contact the lead researcher if they have any concerns.

Appendix D

Security Aspects

This appendix is taken from an assignment given in a Security of Networks & Distributed Systems module where students are required to write a document outlining security considerations for their dissertations. The following sections have been adapted from this assignment and were written using the guidelines for writing RFC text on security considerations [33].

Due to the nature of the mobile application we have developed and the reputation that has developed around privacy and security of participatory sensing we feel it is import to include security considerations in this work.

D.1 DOS (Denial of Service)

A web service will be implemented for this project, it will be a RESTful web service using JSON in POST requests. Distributed Denial of Service (DDOS) remains a significant security risk for web services and web applications alike because they are difficult to detect and prevent. Unlike traditional DOS attacks DDOS appears like normal traffic but in higher volumes. It is straight forward for an attacker to use malware on many machines or networks to generate so many requests to the web service that the service cannot operate normally.

Individual peers or clients using the mobile application could also be denied service if an attacker persistently kills their sessions/connections with the web service or other clients. In peer to peer environments this is known as targeted DOS. In this case the attacker would need to be connected to the same network as the victim but this could be likely if the victim is using the Wi-Fi provided by the bus.

Due to the volume of different types of DOS attack one of the best defences is the ability to scale ones infrastructure in order to simply absorb attack. This can be achieved by using services such as Amazon web services or Microsoft Azure which allow users to snapshot existing server images and use those snapshots to spin up new instances to cope with the extra demand. However there can be a significant cost associated with this depending on the scale of the attack.

Unfortunately there is no real protection against denial of service attacks but one can mitigate the risk of an attack succeeding by implementing a highly scalable architectures as well as using additional resources such as content distribution networks (CDN) and edge caching [1].

D.2 Communications Security Issues

Location and bus occupancy data will be posted to the web service as well as shared between devices in a peer to peer fashion. This creates confidentiality, privacy and authentication issues where a user would rather their location be kept private.

Any HTTP post requests sent across the local network that the user is connected to could be captured, tampered with, or simply deleted. The HTTP protocol is vulnerable to both passive and active attacks. An attacker could capture the users data or simply modify it to maliciously pollute data in the database. These security issues affect data integrity as well as privacy. For example a replay attack may cause data to be logged twice by the server, this could have consequences for other users if they get incorrect occupancy data.

Eavesdropping on a HTTP packets in a network is trivial for an attacker. To provide privacy, data integrity and authentication the SSL/TLS protocol over HTTP should be used. To protect peer communication a protocol such as (Secure Real-Time Transport Protocol) should be used.

Where data is exchanged between peers it should also be encrypted to prevent eavesdropping. The encryption comes at a cost for this type of application as it may involve using asymmetric keys which can be time consuming and require both computing power and randomness to create a sufficient key.

Further problems arise if that key can be sniffed by an attacker on the same network (Because the user will likely be connected to public Wi-Fi on the bus). If an attacker can get the key, all messages may as well be transmitted in plain text.

For this particular application we feel that the use of secure protocols such as SSL is sufficient to protect a users data. No private data is transmitted with the application however the use of SSL is good practice with web servers as it prevents attackers from getting hold of clients server tokens.

D.3 Inappropriate Usage

A common security concern with wireless sensor networks is that the sensors can be tampered with causing the data they report to become polluted or unusable. An attacker could intentionally modify sensors in their device to pollute data in the system.

One way to mitigate possible data pollution attacks is through the use of API keys where the offending device can be identified using outlier detection or confidence levels and then have their key removed so they can no longer contribute.

Some applications have implemented a three strike system where if bad data is detected from the same client three times they are no longer allowed to provide data. It's important to identify bad data quickly and deal with it because if users cannot trust the data produced by the system they will not use it and this type of application relies on users contribution.

D.4 Identity Protection

A user of our mobile application would not want both their Identity and location disclosed when reporting on bus occupancy. An attacker could hack the data base and reveal users private information. Hacking the database could be done in many different ways such as social engineering, brute force, dictionary attack, malware to steal passwords, and so on.

To protect the users Identity results sent between peers and to the web service should be anonymous by providing each user with unique alphanumeric key rather than using something that easily identifies the user such as an email or user name. Anonymous data would be of little use to any attacker however an attack may still take place where an attacker intends to simply damage or pollute the data base.

One consideration is to only identify users with their API key and no other identifying data however if an attacker could identify a person's key then they could proceed to identify them in other places on the database for example if bob is the only person on a certain bus at a certain time the attacker could know that the key belonged to bob's.

While the API key does a sufficient job at protecting the users identity further steps should be taken to prevent unauthorised access to the database.

D.5 Man in the Middle Attack

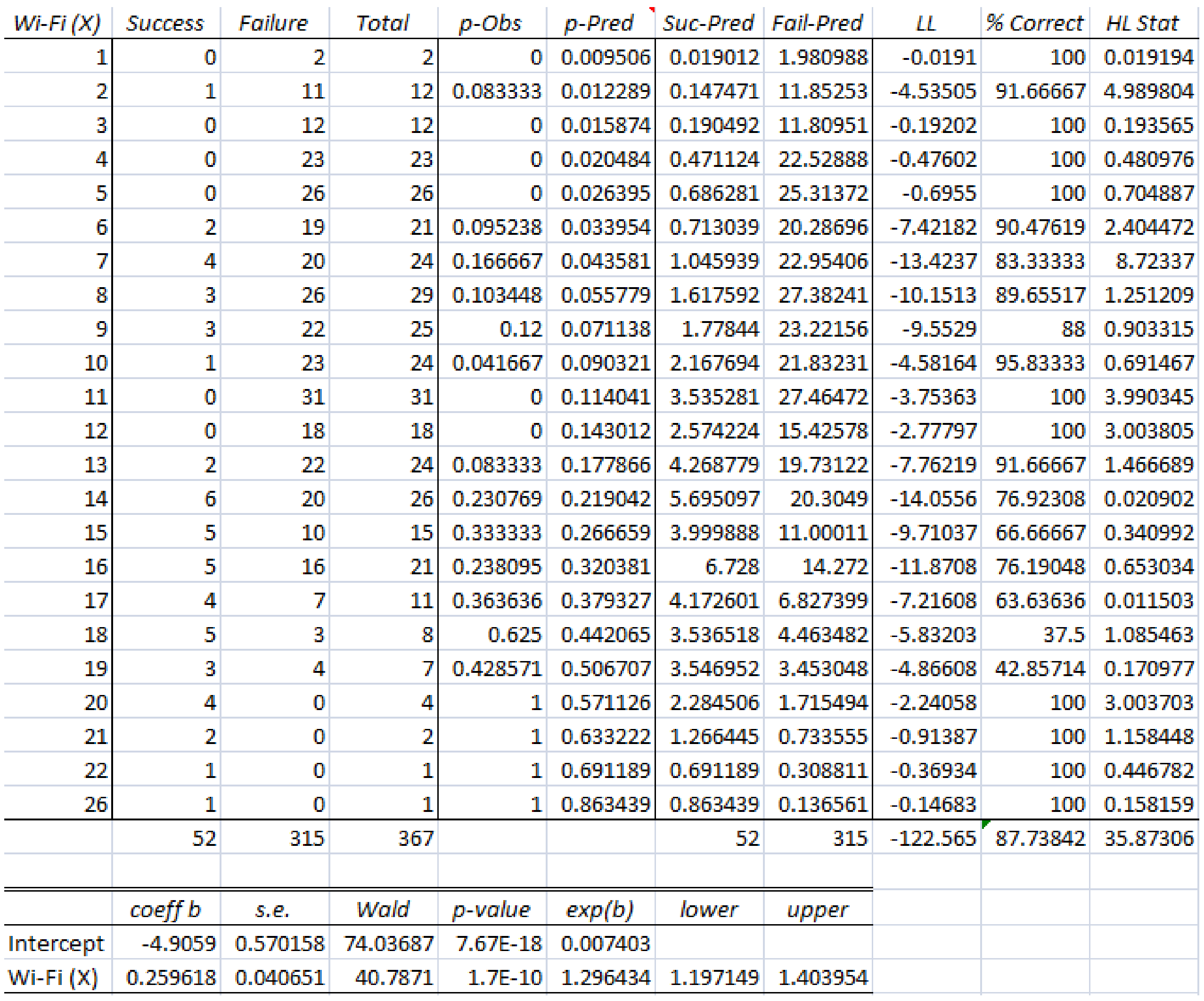
Man in the Middle (MITM) attacks could be carried out between the server and client or client to client communication. As mentioned in section D.2, SSL/TLS should be used with this application which will mitigate the risk of successful MITM attacks on the application.

However, even with protocols such as TLS there are still ways in which attackers can exploit vulnerabilities in those protocols meaning careful consideration should be given as to the nature of data exchanged in the application. A risk analysis should be carried out to assess the risk and impact of leaking of any exchanged data.

Appendix E

Binary Classification Calculation

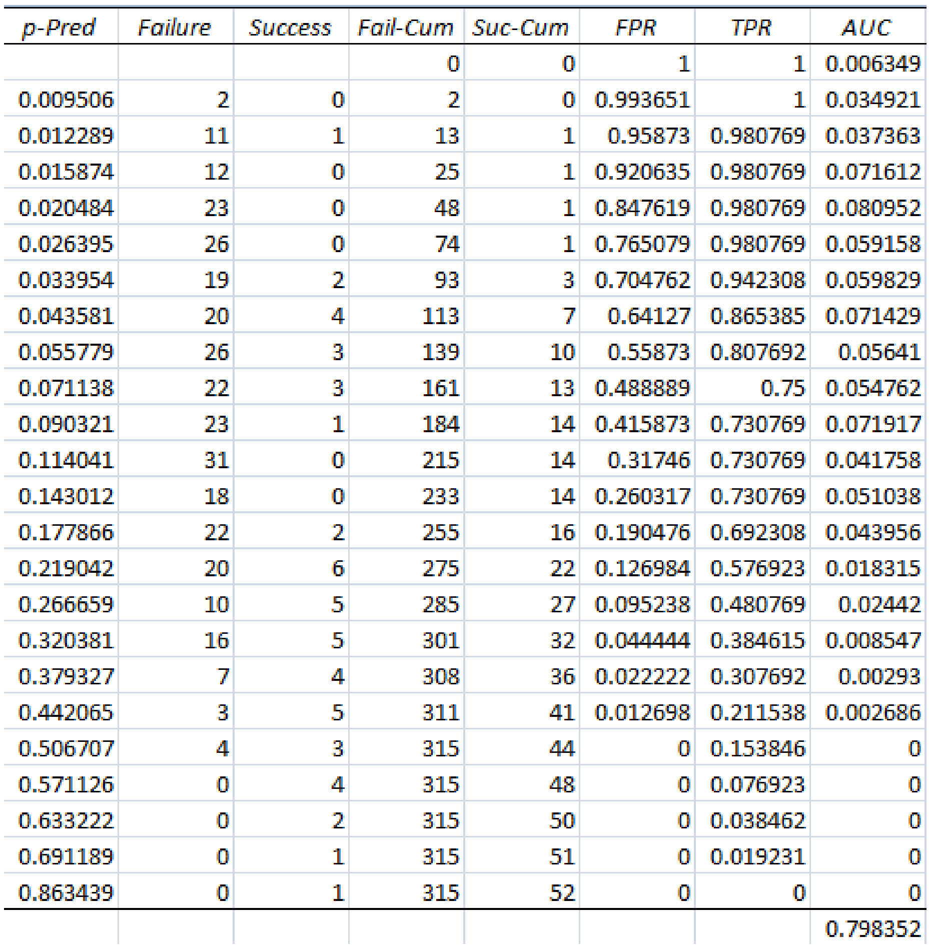
Logistic regression calculations were carried out using the ”Real Statistics Resource Pack” for Microsoft Excel [8]. Fiqure E.1 shows the calculations used to produce the classification table in Section 5.1.4. Figure E.2 shows receiver operating characteristic data which is used to plot the chart in section 5.1.5. The true positive and false positive rates are displayed in this data. The following figures represent data calculated using the entire data set.



Wi-Fi (X)	Success	Failure	Total	p-Obs	p-Pred	Suc-Pred	Fail-Pred	LL	% Correct	HL Stat
1	0	2	2	0	0.009506	0.019012	1.980988	-0.0191	100	0.019194
2	1	11	12	0.083333	0.012289	0.147471	11.85253	-4.53505	91.66667	4.989804
3	0	12	12	0	0.015874	0.190492	11.80951	-0.19202	100	0.193565
4	0	23	23	0	0.020484	0.471124	22.52888	-0.47602	100	0.480976
5	0	26	26	0	0.026395	0.686281	25.31372	-0.6955	100	0.704887
6	2	19	21	0.095238	0.033954	0.713039	20.28696	-7.42182	90.47619	2.404472
7	4	20	24	0.166667	0.043581	1.045939	22.95406	-13.4237	83.33333	8.72337
8	3	26	29	0.103448	0.055779	1.617592	27.38241	-10.1513	89.65517	1.251209
9	3	22	25	0.12	0.071138	1.77844	23.22156	-9.5529	88	0.903315
10	1	23	24	0.041667	0.090321	2.167694	21.83231	-4.58164	95.83333	0.691467
11	0	31	31	0	0.114041	3.535281	27.46472	-3.75363	100	3.990345
12	0	18	18	0	0.143012	2.574224	15.42578	-2.77797	100	3.003805
13	2	22	24	0.083333	0.177866	4.268779	19.73122	-7.76219	91.66667	1.466689
14	6	20	26	0.230769	0.219042	5.695097	20.3049	-14.0556	76.92308	0.020902
15	5	10	15	0.333333	0.266659	3.999888	11.00011	-9.71037	66.66667	0.340992
16	5	16	21	0.238095	0.320381	6.728	14.272	-11.8708	76.19048	0.653034
17	4	7	11	0.363636	0.379327	4.172601	6.827399	-7.21608	63.63636	0.011503
18	5	3	8	0.625	0.442065	3.536518	4.463482	-5.83203	37.5	1.085463
19	3	4	7	0.428571	0.506707	3.546952	3.453048	-4.86608	42.85714	0.170977
20	4	0	4	1	0.571126	2.284506	1.715494	-2.24058	100	3.003703
21	2	0	2	1	0.633222	1.266445	0.733555	-0.91387	100	1.158448
22	1	0	1	1	0.691189	0.691189	0.308811	-0.36934	100	0.446782
26	1	0	1	1	0.863439	0.863439	0.136561	-0.14683	100	0.158159
	52	315	367			52	315	-122.565	87.73842	35.87306

	coeff b	s.e.	Wald	p-value	exp(b)	lower	upper
Intercept	-4.9059	0.570158	74.03687	7.67E-18	0.007403		
Wi-Fi (X)	0.259618	0.040651	40.7871	1.7E-10	1.296434	1.197149	1.403954

Figure E.1: Logistic Regression Calculation Data



p-Pred	Failure	Success	Fail-Cum	Suc-Cum	FPR	TPR	AUC
			0	0	1	1	0.006349
0.009506	2	0	2	0	0.993651	1	0.034921
0.012289	11	1	13	1	0.95873	0.980769	0.037363
0.015874	12	0	25	1	0.920635	0.980769	0.071612
0.020484	23	0	48	1	0.847619	0.980769	0.080952
0.026395	26	0	74	1	0.765079	0.980769	0.059158
0.033954	19	2	93	3	0.704762	0.942308	0.059829
0.043581	20	4	113	7	0.64127	0.865385	0.071429
0.055779	26	3	139	10	0.55873	0.807692	0.05641
0.071138	22	3	161	13	0.488889	0.75	0.054762
0.090321	23	1	184	14	0.415873	0.730769	0.071917
0.114041	31	0	215	14	0.31746	0.730769	0.041758
0.143012	18	0	233	14	0.260317	0.730769	0.051038
0.177866	22	2	255	16	0.190476	0.692308	0.043956
0.219042	20	6	275	22	0.126984	0.576923	0.018315
0.266659	10	5	285	27	0.095238	0.480769	0.02442
0.320381	16	5	301	32	0.044444	0.384615	0.008547
0.379327	7	4	308	36	0.022222	0.307692	0.00293
0.442065	3	5	311	41	0.012698	0.211538	0.002686
0.506707	4	3	315	44	0	0.153846	0
0.571126	0	4	315	48	0	0.076923	0
0.633222	0	2	315	50	0	0.038462	0
0.691189	0	1	315	51	0	0.019231	0
0.863439	0	1	315	52	0	0	0
							0.798352

Figure E.2: Receiver Operating Characteristic Calculation Data

Bibliography

[1] Mehmud Abliz. Internet Denial of Service Attacks and Defense Mechanisms. (March):42, 2011.

[2] Apple Inc. Apple - Multipeer Connectivity Framework. `https://developer.apple.com/library/prerelease/ios/documentation/MultipeerConnectivity/Reference/MultipeerConnectivityFramework/index.html`.

[3] National Transport Authority. Bus Statistics for Ireland. Technical Report June.

[4] J Burke, D Estrin, M Hansen, N Ramanathan, S Reddy, and M B Srivastava. Participatory sensing. *In: Workshop on World-Sensor-Web (WSW06): Mobile Device Centric Sensor Networks and Applications*, pages 117–134, 2006.

[5] Giuseppe Cardone, Andrea Cirri, Antonio Corradi, Luca Foschini, Raffaele Ianniello, and Rebecca Montanari. Crowdsensing in Urban Areas for City-Scale Mass Gathering Management : Geofencing and Activity Recognition. 14(12):4185–4195, 2014.

[6] Giuseppe Cardone, Andrea Cirri, Antonio Corradi, Luca Foschini, and Dario Maio. MSF : An Efficient Mobile Phone Sensing Framework. 2013, 2013.

[7] Giuseppe Cardone, Luca Foschini, Paolo Bellavista, and Antonio Corradi. Fostering ParticipAction in Smart Cities : A Geo-Social Crowdsensing Platform. (June):112–119, 2013.

[8] Charles Zaiontz. Real Statistics Using Excel. `http://www.real-statistics.com/logistic-regression/`, 2015.

[9] Emiliano De Cristofaro and Claudio Soriente. Participatory privacy: Enabling privacy in participatory sensing. *IEEE Network*, 27(1):32–36, 2013.

[10] Deloitte. Cost and Efficiency Review of Dublin Bus and Bus Éireann Contents. Technical Report January, 2009.

[11] Dublin Bus. Dublin Bus Fleet - Dublin Bus. `http://www.dublinbus.ie/en/About-Us/Dublin-Bus-Fleet/`, 2015.

[12] Global System for Mobile Communications. Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); AT command set for User Equipment (UE) (3GPP TS 27.007 version 8.5.0 Release 8). 0:81, 2008.

[13] Google. Wi-Fi Peer-to-Peer — Android Developers.

[14] Google. Services — Android Developers. `http://developer.android.com/guide/components/services.html`, 2015.

[15] Yuan Hejin. Bus Passenger Flow Estimation Method Based on Feature Point's Trajectory Clustering. *Power*, pages 426–430.

[16] Norihiro Ishikawa, Takeshi Kato, Hiromitsu Sumino, Nippon Ericsson K K, and Johan Hjelm. Jupiter : Peer-to-Peer Networking Platform over Heterogeneous Networks. *Cybernetics*, 4(5):55–61.

[17] Waheb a. Jabbar, M. Ismail, and R. Nordin. Peer-to-peer communication on android-based mobile devices: Middleware and protocols. *2013 5th International Conference on Modeling, Simulation and Applied Optimization, ICMSAO 2013*, 2013.

[18] Aubort Jean-Baptiste. Network Discovery - Android Apps on Google Play. `https://play.google.com/store/apps/details?id=info.lamatricexiste.network\&hl=en`, 2013.

[19] Aubort Jean-Baptiste. Network Discovery. `https://github.com/rorist/android-network-discovery`, 2015.

[20] Salil S. Kanhere. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7753 LNCS:19–26, 2013.

[21] Vassilis Kostakos, Tiago Camacho, and Claudio Mantero. Wireless detection of end-to-end passenger trips on public transport buses. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pages 1795–1800, 2010.

[22] Ioannis Krontiris and Tassos Dimitriou. A platform for privacy protection of data requesters and data providers in mobile sensing. *Computer Communications*, 65:43–54, 2015.

[23] Nicholas D Lane, Shane B Eisenman, Mirco Musolesi, Emiliano Miluzzo, and Andrew T Campbell. Urban sensing systems. *Proceedings of the 9th workshop on Mobile computing systems and applications - HotMobile '08*, page 11, 2008.

[24] Nicholas D. Lane, Emiliano Miluzzo, Hong Lu, Daniel Peebles, Tanzeem Choudhury, and Andrew T. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140–150, 2010.

[25] Elsa Macias, Alvaro Suarez, and Jaime Lloret. Mobile sensing systems. *Sensors (Basel, Switzerland)*, 13(12):17292–321, 2013.

[26] Microsoft. Correl function. `https://support.office.com/en-us/article/CORREL-function-995dcef7-0c0a-4bed-a3fb-239d7b68ca92`, 2015.

[27] Snehal Mumbaikar and Puja Padiya. Web Services Based On SOAP and REST Principles. 3(5):3–6, 2013.

[28] Nternet Of. Mobile Crowdsensing : Current State and Future Challenges. (November):32–39, 2011.

[29] E. ONeill, V. Kostakos, T. Kindberg, a. Fatah gen. Schiek, a. Penn, D. Stanton Fraser, and T. Jones. Instrumenting the city: developing methods for observing and understanding the digital cityscape. pages 315–332, 2006.

[30] Thongtat Oransirikul, Ryo Nishide, Ian Piumarta, and Hideyuki Takada. Measuring Bus Passenger Load by Monitoring Wi-Fi Transmissions from Mobile Devices. *Procedia Technology*, 18(September):120–125, 2014.

[31] Jeongyeup Paek, Kyu-Han Kim, Jatinder P. Singh, and Ramesh Govindan. Energy-efficient positioning for smartphones using Cell-ID sequence matching. *Proceedings of the 9th international conference on Mobile systems, applications, and services - MobiSys '11*, page 293, 2011.

[32] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks*, 6(2):1–27, 2010.

[33] Eric Rescorla and Brian Korver. Guidelines for Writing RFC Text on Security Considerations. pages 1–44, 2003.

[34] Antony Rowstron and Peter Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. *Middleware 2001*, 2218(November 2001):329–350, 2001.

[35] Chithra Selvaraj and Sheila Anand. A survey on Security Issues of Reputation Management Systems for Peer-to-Peer Networks. *Computer Science Review*, 6(4):145–160, 2012.

[36] Skype. Skype — Free calls to friends and family. `http://www.skype.com/en/`, 2015.

[37] Ion Stoica, Ion Stoica, Robert Morris, David Karger, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM '01*, pages 149–160, 2001.

[38] H. Sumino, N. Ishikawa, and T. Kato. Design and implementation of P2P protocol for mobile phones. *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, pages 0–5, 2006.

[39] Yasha Wang, Jiangtao Wang, and Xiaoyu Zhang. QTime: A Queuing-Time Notification System Based on Participatory Sensing Data. *2013 IEEE 37th Annual Computer Software and Applications Conference*, pages 770–777, 2013.

[40] Jens Weppner and Paul Lukowicz. Bluetooth Based Collaborative Crowd Density Estimation with Mobile Phones. *IEEE International Conference on Pervasive Computing and Communications*, (March):193 – 200, 2013.

[41] Haibin Yu, Zhiwei He, and Jilin Liu. A vision-based method to estimate passenger flow in bus. *2007 International Symposium on Intelligent Signal Processing and Communications Systems, ISPACS 2007 - Proceedings*, pages 654–657, 2008.

[42] Pengfei Zhou, Yuanqing Zheng, and Mo Li. How Long toWait?: Predicting Bus Arrival Time with Mobile Phone based Participatory Sensing. *Proceedings of the 10th international conference on Mobile systems, applications, and services - MobiSys '12*, 13(6):379–392, 2012.

[43] Fang Zhu, Junhua Gu, Ruixia Yang, and Zhifeng Zhao. Research on counting method of bus passenger flow based on kinematics of human body and SVM. *Proceedings - 2008 2nd International Symposium on Intelligent Information Technology Application, IITA 2008*, 3:14–18, 2008.