Literature Review

mah60

July 2019

Contents

1 Introduction

The research being completed in this project is to look into the project question; "Investigating the use of individuals mobile devices as a mechanism to enhance locational information within a geographic setting". The project will require research into multiple different areas about networking and electronics. The literature review is here to demonstrate the research that was done during the project.

The main objectives that will be looked into throughout the project is to create a device that will provide solutions towards the use cases that were given within the Project Outline. These use cases are;

1. Small Event Visit Counter
2. Medium Event Counter
3. Path Tracking Application
4. Rerouting Application

The design of the product that is being has been based around key factors that will be discussed later in this document. However, the key components that define how the structure of the design are;

- Microcontroller's - used for wifi tracking and data processing.
- Data Storage - the database that is used to store the selected data and how it will be stored.
- Data processing - what method will be used to process the data to analyse the gathered information.

The components that are selected to complete the project will determine how efficiently the different jobs are done and will effect the overall effectiveness of the results. The key variables that will need to be collected to achieve the use cases given are;

- Mac Address[1] - is a unique identifier for each device that is connected to the internet. The variable will look like '0c:cb:85:25:d1:f1'. The mac address cannot be used to trace any personal data back to the owner of the device and can only provide vendor information about the device. This data will be used as a primary key to identify each device.
- RSSI[2][3] – Resource Strength Indicator – the signal strength between the two devices, the strength values range between 0 to -100. The stronger the signal the higher the number. The RSSI will give the possibility to determine the range for readings inside the event and will be the primary resource to determine the location of a device.
- Timestamp – this variable will be used to give a time of when each individual signal was received. The variable will look like '2019-08-01 20:59:02' and will be used to calculate dwelling time and help compare individual signals and see if they are related or not.

From these variables select information can be gathered to create the use cases and produce appropriate results that can be given to the user of the product.

2 Similar Projects

The initial research for this project look towards similar ideas that may indicate different solutions strengths and weaknesses. Therefore, providing a guide to help create a path in which the project should move in and prepare us for possible issues along the way.

2.1 Smart Bins

The project we will be looking at is 'Smart Bins' that were placed around London by Renew. BBC news goes into details about the problem area around the 'Smart Bins'[4], whereas Quartz go into further details about the development of the 'Smart Bins;[5]. The main issue that was found during these reports is that the Renew company had developed these bins to count footfall data from people passing by the bins and also provide advertising at the same time. However, the company collected more data than required and started providing select advertising based on paths the devices took. The project was closed because of this breach of personal data being gathered and therefor breaking the law as they did not state they were gathering this data. They had further plans to expand to other cities and use the location tracking algorithms to gather more personal data on a device holder, such as their gender.

The 'Smart Bins' were useful for showing that it is possible to create a footfall counter and to indicate the ethical and legal issues that this project could involve. As the data collected could be abused to collect personal information on a device holder without them knowing. This is because if a device is in promiscuous mode, the device will intercept wifi packets and leave no trace that the packets being sent was read to gather data about the device sender or receiver.

2.2 Wifi Tracking Using Arduino Wifi Shield

The next report that was showed an approach towards the project was position tracking using an Arduino with a wifi shield by Nathan Conrad at Western Michigan University[6]. The product that was being developed was a network of devices that would help indicate an employee or object location within the workplace. Each employee would hold a device that would connect to a tracker, that only worked within the company building. The device being held by the user would allow for a button to be pressed to indicate an emergency inside a building and then would provide a map the location of the emergency inside the building. The main issue that was found during the development of this project was that depending on the amount of access points that are inside the building, the signals may be interrupted or become corrupted due to noise from the other access points. Therefore, meaning the reading could contain error within the RSSI, this would cause problems with location tracking as it is the main method of determining the location of a device. The wifi shield was changed from a CC 3000 to a Arduino Wifi Shield that would provide a more efficient solution as the device was less confused by noise within the building.

This report displays the possibility of location tracking through wifi tracking and a clear position of a device can be indicated if a network of device work together to collect appropriate results. However, their can be issues with accurate recordings that depend on the environment that the network is being used within. Meaning during the testing of this project different environments should be experimented with to gather information about the possible flaws of the design, such as too many access points could generate a lot of noise within the network.

2.3 TFL Wifi Tracking in the London Underground

The next report discusses possible areas of data that could be collected from wifi tracking. The report explores the testing that TFL did to experiment on whether wifi tracking could give accurate information about the whereabouts of users within the London underground and the routes they take[7]. The reasoning behind this research is because of the oyster card only being able to gather information about where a customer enters and leaves the underground. The data collected provides information about the most used routes and peak hours of when people are travelling through the underground. However, this does not provide the actual route the customer took to get between the two locations. This is where wifi tracking would help, as throughout the testing process wifi tracking was used through customers mobile devices to provide additional information to the company, such as which tube lines customers used most and least and how people got between two locations in the underground.

The data collected by TFL was used to produce multiple results that expressed important information, like the most routes between two stations. This allowed TFL to search for all the different routes between two stations, such as King Cross to Water Loo, and discover the most used ones. Therefor this could allow for a journey planner to be developed and supply users of alternative routes to avoid traffic. However, this report also talks about a problem called a feedback loop, due to if you tell everyone to go one way to avoid traffic then that route will become crowded and so on. They are looking into solutions for this problem, however, this does bring the problem forward when developed the Routing Application. The report expresses the need for wifi tracking to be used in the London Underground and transportation systems. This is due to the information being collected could help to support travel between two locations, to indicate when development can be done on the train line, e.g. construction or repairs, or rerouting customers if delays occur.

3 Product Development Research

The core part of the project is to build a product that can track mobile devices and record the data from those devices. There are three core structures that will provide a resource to conduct research into the usefulness of wifi tracking. The core parts of the product are the microcontrollers that can be used as a tracker, data storage and processing. Then research will need to be completed to create the architecture for the database and the network. These parts will be used to data collection and perform data processing techniques during the development of the product.

3.1 Microcontrollers Research

The microcontrollers are a key part to the project as this is where the devices will manage the collection, storage and processing of the data. There were two microcontrollers were chosen for the tasks in the project, as they were easy to obtain and will be sufficient enough to do the tasks selected. The microcontrollers selected are;

- ESP8266 - ESP-12E with Arduino IDE[8] - is a low cost wifi microcontroller that will be programmed using the Arduino IDE[11]. The application for the microcontroller is to be used as a tracker to gather information about mobile devices within the vicinity. If possible the device will also be used to store and process the data to get the appropriate output.
- Raspberri Pi Model B Revision 2[9] - this is a computer that can run Linux. This device will be able to run python, which is a language that is very powerful for processing and analysing data[10]. This device will be used for processing the data collected by the individual trackers.

These microcontroller will be used to build the product and create a network of devices that will be able to track mobile devices to retrieve the required information. Then once the data is collected it will be stored and processed on these microcontrollers.

3.2 Database Design

The database is where all data will be stored and processed. Therefor it is essential to think about how the data be collected and a sufficient way on how to store the information. The first task is to look at the key variables that will need to be stored. Taking into account that there will be multiple trackers and one database, these are the variables that will be collected for each recording.

- Mac Address[1] - the mac address of the device being recorded. A one-way hash will be required to apply against the mac address.
- RSSI[2] - signal strength for the connection between the tracker and the mobile device.
- Timestamp - time and date of when the reading was taken.
- Tracker Mac Address - tracker mac address as each device recording the data will need to be uniquely identified.

There are multiple database management systems (DBMS) that could be used to store this data, such as a document orientated DBMS or relational DBMS. The relational DBMS would be the most ideal choice for this type of data collection as the recordings will always be fixed variables and information. Meaning there should be no missing values or additional variables in the data collected. Therefor a document orientated DBMS is not required.

This reduces the amount of DBMS down to one category of research and means that there are select DBMS that could be used for the task. These DBMS will be required to be used in either Arduino or Python. The DBMS that are looked at are;

- MySQL [12] - The relational DBMS that contains a predetermined schema for the incoming data. Meaning the data handed to the database must be fixed.
- SQLite[14] - This is a simplified version of MySQL allowing for easier commands and processing. However, the main problem that resides with this language is that it struggles to handle multiple calls at once. Meaning if a lot of data is coming in at once the database may become locked.
- PostgreSQL[13] - This language is familiar to MySQL but is more supported of extensibility and technical standards. Meaning the DBMS can be used on one device to a warehouse full of servers.

The DBMS that would be the best to support the project would be MySQL as it can handle multiple inputs at the same time. Whereas, SQLite cannot as it is a simplified version. PostgreSQL is suitable for the project but could be seen as too heavy duty to build on a single Raspberry Pi. Therefore if the project was upgraded for strorage on multiple servers it would be more reliable to use PostgreSQL in that scenario. There is also code that supports the development of MySQL on Raspberry Pi devices, therefor making the DBMS more suitable for the task at hand.

3.3 Network Structure

The final key part is to create a system that would be able to link the different microcontrollers together. This will need to look into two different scenarios, these are whether the internet is turned off or on. The data will need to be sent to the Raspberry Pi to upload the data to the database for further data analyses.

When the database is turned off uploading the data will simply require the SD card to be taken from the ESP8266 and placed in the Raspberry Pi to upload the data. This will require for another SD card slot to be installed or the SD card to be inserted into a USB converter. Another method is to wait till the ESP8266 device to be connected to the wifi and send the data through the method that is developed to send data over a secure connection between the two devices.

Transmitting the data between the devices over wifi will be more complex as the connection will need to be secure and made properly through transactions. Therefor sufficient research into the subject will need to be done before testing the product, this method will need to be developed for the use of three to five ESP8266 to be able to connect to the Raspberry Pi and send the data collected over. This task will need to be done at a reassemble speed to ensure little data is lost while the transaction is being made. This is because while the wifi is turned on to make the transactions, promiscuous mode will be turned off and in the time we could miss valuable information if the code takes too long to switch back to promiscuous mode. The methods that could be used to do this are;

-

——- need to complete research ————

4 Code Development

This section discusses the research that was done to develop the code for the two devices, the tracker and the data processor. These parts will be split into two different sections that will demonstrate the research that was conducted for the project prior and during the development of the project.

4.1 Tracker Code

The requirements for the tracker is to simply intercept packages between the holders device and the wifi router. This will give a confirmation that a device holder is near by and the RSSI will indicate how far the holder is from the ESP8266. The key requirements for the ESP8266 is to do the following tasks;

- Collect data
- Storing the data
- Sending the data to the data processor

The following sections will discuss how these key tasks were looked into and complete to create a tracker that will collect the data required about mobile devices in the local area.

4.1.1 Collecting Data

The first task for the required data is to retrieve the Mac Address of the device holder and the RSSI from using promiscuous mode. The code for this can be found in a blog by Łukasz Podkalicki for 'ESP8266 - Wifi Sniffer'[15] and was adapted to only provide the data that was needed. This callback function will be called each time a package is spotted, after this done the other variables that is needed will be added to the string to provide all the information needed.

Once this was done the next task is to encrypt the Mac Address, the initial research was looking into different encryption algorithms and library's that can be used on the ESP8266. The core algorithm that was looked into was the AES algorithm[16] that is used by the USA to encrypt their data. This would of secured the data and the libraries that were looked into was 'Crypto'[17] and 'AES', however, through extensive research and testing the algorithm to set up an encryption was being too complex and taking too much time. Therefor an alternative method was used to obscure the Mac Address this was applying a one-way hash against the Mac Address, the chosen hashing algorithm used was 'MD5' using the 'MD5' library[18]. This was deemed a simplified way to obscure the Mac Address, the main task for this security is to scare off any people who do not intend to malicious attack the data. The core principal behind the hashing is how password protection is done, the only way to compare a Mac Address is to apply the MD5 algorithm against the mac address and match the outputted string to data inside the database. Therefor obscuring the data to anyone looking at the database for processing purposes or collecting the results from the network.

The next part of data collection is to gain a time stamp of when the data was intercepted. This will allow for dwelling time to be recorded. The initial look was to use the 'NTPclient' library[19] to gain a real Date Time stamp for the database. However, the main problem with this idea is that this time stamp can only be received while the wifi is active. Therefor, the next task for to gain a time stamp is to set the real time clock (RTC) on the ESP8266 but the microcontroller does not hold an RTC on the board without adding an additional device. With all the research being collected, the final decision was to provide two sets of data. The date time stamp provided from the 'NTPclient' while the wifi is turned on and to use the 'millis()' command[20] to get the number of milli-seconds the device had been running. The milli-seconds provided from the command was converted to seconds and subtracted the start up time to make the recordings more accurate.

The final output for the collection of the data gave a string of data that followed this format; 'Mac Address, Tracker Mac Address, start date time stamp, run time (seconds), RSSI;'. This string will be saved and sent to the Data processor to be split up and stored into a database.

4.1.2 Storing Data

This therefor brings us to the next objective, which is storing the data collected. When first looking into this the initial idea was to put the data into a SQLite database on the ESP8266, however, this was determined to be a incorrect methods as on the ESP8266 device a SQLite library can only hold 500 records[21]. Which wouldn't be suitable for the task at hand. So a much easier solution was developed to save the data to a text file and using delimiters to separate the data. The initial research showed the library 'SPIFFSIniFile'[22] to be a suitable method to save the data, however, a problem appeared that the file would be wiped every time the ESP8266 was reset or turned off as the data resided in memory. A solution to this problem would be to get a SD card chip installed onto the ESP8266 to store the data collected and allow for more storage, a simple method found Fernando Koyanagi in his article about 'SD Card Module With ESP8266'[23] shows a simple method of using the library 'SD' library to write a file to the SD card and store the data on that SD card.

4.1.3 Sending Data

The final task for the tracker to complete is to send the data collected to the data processor. To be stored onto the database and be processed. ——- need to complete research ————

4.2 Data Processor Code

5 Version

Version	Description	Date
0.1.0	Created the initial design for the document and created the introduction chapter.	01/07/2019
0.2.1	Literature reviews chapters that have been joined up are; RSSI, Mac Address, tomographic reconstruction. Relatable Projects; smart bins, position tracking using arduino wifi shields.	02/07/2019
0.2.2	proofread created chapters	06/07/2019
0.2.3	Completed reading on Position Tracking Using Arduino Wifi Shield, added access points	13/07/2019
0.3.0	Redesign of document to make it more narrative and in clude a bibliography. Completed up to database design.	03/08/2019
0.3.1	Continued the database design review and worked on the Network Structure.	09/08/2019
0.3.2	Started writing and researching into code development. As well, as adding the references and citing's	10/08/2019
0.3.3	completed proofreading and citation for the research done so far.	13/08/2019

References

[1] Privacy Company; Mac Address; 10/08/2019; *https://www.privacycompany.eu/en/what-does-the-gdpr-say-about-wifi-tracking/*

[2] NetSpot Pro; RSSI; 10/08/2019;
https://www.netspotapp.com/what-is-rssi-level.html

[3] ScienceDirect; RSSI research; 10/08/2019; *https://www.sciencedirect.com/topics/computer-science/received-signal-strength*

[4] BBC News; Smart Bins News Report; 10/08/2019;
https://www.bbc.co.uk/news/technology-23665490

[5] Quartz; Smart Bins Technology; 10/08/2019;
https://qz.com/112873/this-recycling-bin-is-following-you/

[6] Western Michigan University; Nathan Conrad; Position Tracking using wifi; 10/08/2019;
https://pdfs.semanticscholar.org/c369/f4ea8cf81b14e445ac45fd4f2fa72d3f8c3f.pdf

[7] Gizmodo; London Underground Wifi Tracking; 10/08/2019;
https://www.gizmodo.co.uk/2017/09/london-underground-wifi-tracking-heres-everything-we-learned-from-tfls-official-report/

[8] Amazon; ESP8266 - ESP-12E with Arduino IDE ; 13/08/2019;
https://www.amazon.co.uk/HiLetgo-Internet-Development-Wireless-Micropython/dp/B0791FJB62?th=1

[9] Pololu; Raspberry Pi Model B, Revision B; 13/08/2019;
https://www.pololu.com/product/2750

[10] Python Software Foundation; Python; 13/08/2019;
https://www.python.org/

[11] Arduino; Arduino IDE ; 13/08/2019;
https://www.arduino.cc/en/main/software

[12] Guru99; MySQL ; 13/08/2019;
`https://www.guru99.com/sql-vs-mysql.html`

[13] PostgreSQL ; 13/08/2019;
https://www.postgresql.org/

[14] SQLite; 13/08/2019
https://www.sqlite.org/index.html

[15] Łukasz Podkalicki; ESP8266 - Wifi sniffer; 13/08/2019;
https://blog.podkalicki.com/esp8266-wifi-sniffer/

[16] Wikipedia; AES; 13/08/2019;
https://en.wikipedia.org/wiki/Advanced$_E$*ncryption*$_S$*tandard*

[17] Arduino Crypto Library; Crypto; 13/08/2019;
https://rweather.github.io/arduinolibs/crypto.html
https://github.com/rweather/arduinolibs/tree/master/libraries/Crypto

[18] Arduino Core Library's; MD5 Hashing; 13/08/2019;
https://github.com/esp8266/Arduino/blob/master/cores/esp8266/md5.h

[19] arduino-libraries; NTPclient; 13/08/2019;
https://github.com/arduino-libraries/NTPClient

[20] Arduino; Millis function; 13/08/2019;
https://www.arduino.cc/reference/en/language/functions/time/millis/

[21] ESP8266 Community Form; ESP8266 SQLite; 13/08/2019;
https://www.esp8266.com/viewtopic.php?f=8t=18603

[22] TechTutorialsX; SPIFFS:Writing a file; 13/08/2019;
https://techtutorialsx.com/2018/08/05/esp32-arduino-spiffs-writing-a-file/

[23] Indestructible Circuits; Fernando Koyanagi; SD Card Module with ESP8266; 13/08/2019;
https://www.instructables.com/id/SD-Card-Module-With-ESP8266/