Contents

[Introduction 2](#_Toc445215023)

[Project Planning and Organisation 3](#_Toc445215024)

[Research 4](#_Toc445215025)

[Application Type 4](#_Toc445215026)

[Web Application 4](#_Toc445215027)

[Native Application 4](#_Toc445215028)

[Comparison of Native and Web Applications 4](#_Toc445215029)

[Android 6](#_Toc445215030)

[iOS 6](#_Toc445215031)

[Comparison of Android and iOS Applications 6](#_Toc445215032)

[Static or Dynamic Routing 7](#_Toc445215033)

[Static Routing 7](#_Toc445215034)

[Dynamic Routing 7](#_Toc445215035)

[Location Services 8](#_Toc445215036)

[Global Navigation Satellite System (GNSS) 8](#_Toc445215037)

[Cell Tower Based Location 9](#_Toc445215038)

[Wi-Fi Based Location 9](#_Toc445215039)

[Current Android Solution 10](#_Toc445215040)

[Dynamic Routing Field Test 10](#_Toc445215041)

[Improved Suggestion 10](#_Toc445215042)

[Routing 10](#_Toc445215043)

[Routing Information 10](#_Toc445215044)

[Path Finding 10](#_Toc445215045)

[Design 11](#_Toc445215046)

[Initial Design 11](#_Toc445215047)

[Final Design 11](#_Toc445215048)

[Reason for changes 11](#_Toc445215049)

[Software patterns 11](#_Toc445215050)

[References 11](#_Toc445215051)

# Introduction

The main aim of this project is to create an application that can:

* Implement a routing algorithm capable of dealing with an institute the size of the John Anderson campus.
* Visualise this route in a way that a user can easily understand.

In addition to this over the course of the project an investigation should be carried out into the feasibility of including route modifiers and location services into this application. For the purposes of this report Route Modifiers are defined as settings which depending on their values will give different routes for different users. An example of this would be to avoid stairs. Location services would be some way of automatically determining where the user is within the mapped section. This could then be used to either provide Dynamic Routing as they follow the path or to simply allow them to input their start location more easily.

# Project Planning and Organisation

This project took place over several months which meant that if time was not managed carefully important tasks may have been left incomplete. From this need a project plan was developed.

# Research

Before any development work could begin several key questions had to be answered about how the project should proceed: Would the application provide static or dynamic routing to the user? How would the Application be developed? How would the shortest path be found?

## Application Type

It was decided that the application should be able to be used on mobile devices as this offers the greatest levels of flexibility to the user. This means the user could use it throughout the day as they traverse the mapped area as opposed to having to plan their routes for the day in advance. With this requirement in mind there are two main types of application that would be possible a Web Application or a Native Application [1].

### Web Application

These applications are developed to be ran within a web browser. This means that they can be used across virtually any device including both mobile devices and more traditional computing platforms. They are designed and built as a website with special care taken to appear correctly on mobile devices. This means they can be accessed by navigating to a particular webpage on the phone.

### Native Application

Native applications are built specifically for the mobile operating system being targeted. This means if it is built for Android it cannot be used on iOS or vice versa. To use a Native application, they are first installed from an OS specific store (either the App Store for iOS or the Google Play Store for Android). They can then be launched from a home menu.

### Comparison of Native and Web Applications

To decide between Native and Web Applications they must be compared on a number of key features to the project. This reasoning can be seen in Table 1.

Table 1 Comparison of Native and Web Applications on Key Issues

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Issue to compare | Native Application | Web Application | Which is better for this project | Reason |
| Portability | Can only be used on the OS it was designed for. | Can be used on virtually any Operating System that includes support for a web browser. | Web | If this was to be released it would be important for as many students to have access to it as possible. Although multiple Native Applications could be created it would be far easier to have one Web Application. |
| Performance | Developed specifically for each platform which means greater performance. | Lower performance. | Native | Performance may be an issue as the size of the map increases. Although this is also an issue for the choice of routing algorithm it makes sense to make the later choice easier. |
| Access to phone sensors | Virtually full access to all of the phone sensors. | More restricted access to phone sensors. | Native | This could be important for location services. Although Web applications have access to geolocation information [2] they do not appear to have access to information on Wi-Fi access points they are not currently connected to [3]. Native applications by comparison have access to this information [4]. |
| Programming Language | Uses Object Oriented languages (Java or Objective C) which makes Encapsulation, Abstraction, etc far easier than they would have been otherwise. | Uses JavaScript for the main bulk of work. This can be programmed in an object oriented fashion but it is much more difficult to do so. Does not enforce strict type-checking meaning errors can be made far more easily during development. | Native | It provides a better framework for producing an error free application more quickly. |

As can be seen from Table 1 Native applications have a number of advantages that are important to the project particularly for the study into Location Services feasibility. There are two major choices to be considered when developing a Native Application: Android and iOS.

### Android

These applications are developed specifically for the Android OS and are widely available for download in the Google Play Store. Android phones are produced by a range of manufacturers including Sony, Samsung and HTC.

### iOS

iOS applications are developed specifically for Apple’s mobile devices i.e. the iPhone and iPad. They are available for download on these devices through the App Store.

### Comparison of Android and iOS Applications

A number of issues should be considered before deciding on a platform to build the application for. Several of these are examined in

Table 2 Comparison of Android and iOS Applications on Key Issues

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Issue | iOS | Android | Better for this project | Reason |
| Percentage of users worldwide | 13.9% [5] | 82.8% [5] | Android | Users are far more likely to own an Android phone and therefore be able to use the application. |
| Integration with mapping services | Wide support for many major mapping applications including Google [6] and Bing [7]. | Similar levels of support from the major mapping applications [8] [9]. | Tie | Both options offer similar levels of integration with popular mapping applications. |
| Availability | None. | Several devices of different sizes and features. | Android | Circumstances surrounding the project have dictated these are more easily available. |

For the reasons detailed above the application will be developed as an Android application.

## Static or Dynamic Routing

To determine whether the application should provide static or dynamic routing it is first important to understand what both involve.

### Static Routing

For the purposes of this report Static Routing will be defined as routing between two fixed points with the whole route available for the user to view. A popular example of this is google maps which can be seen in Figure 1.

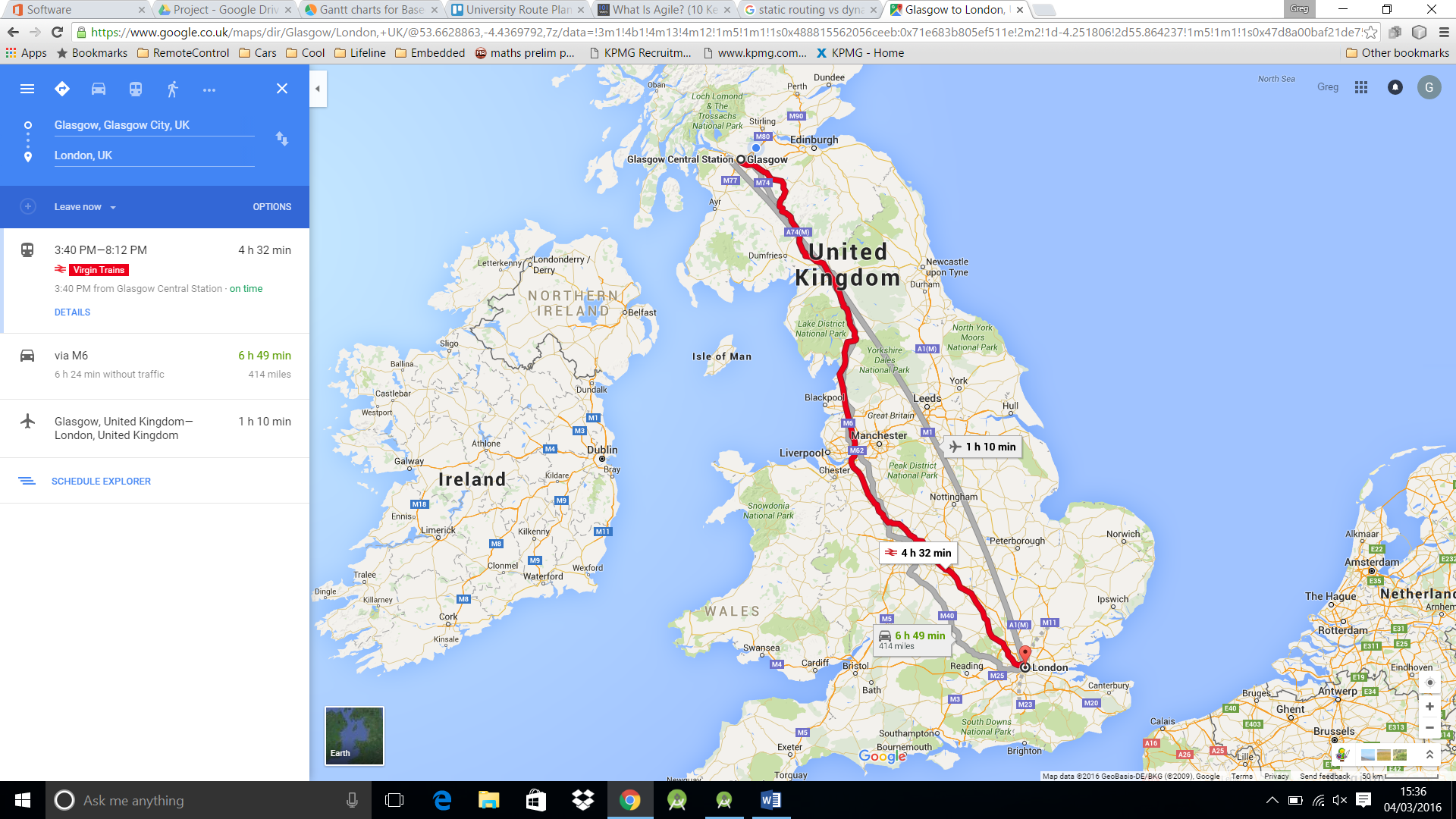


Figure 1 Static Routing Example [10]

### Dynamic Routing

By comparison Dynamic Routing will be defined as routing from the user’s changing location to a fixed endpoint. This means as the user moves the route will automatically adjust to their new location. This is the type of routing used in Satellite Navigation systems or Google Navigation which can be viewed below in Figure 2.

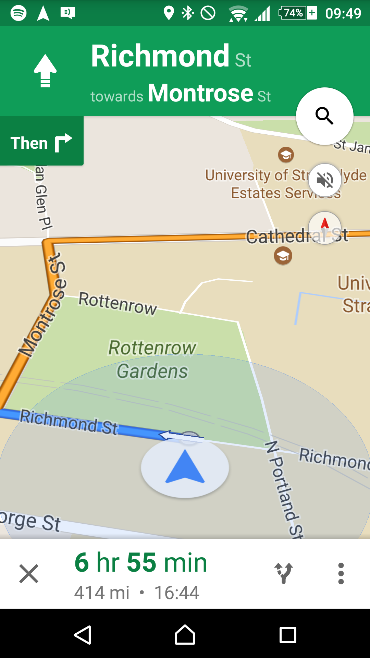


Figure 2 Dynamic Routing Example

It is also common for the display to be more restricted showing only a short section of the route at a time i.e. where the user is to go next.

## Location Services

The main difference between the two types of routing is whether or not they the user’s location is utilised to improve the user interface or not. In order to decide between them then it is important to determine whether this is possible.

There are three main technologies which could be used to determine the user location: Global Positioning Service, Cellular Positioning and Wi-Fi.

### Global Navigation Satellite System (GNSS)

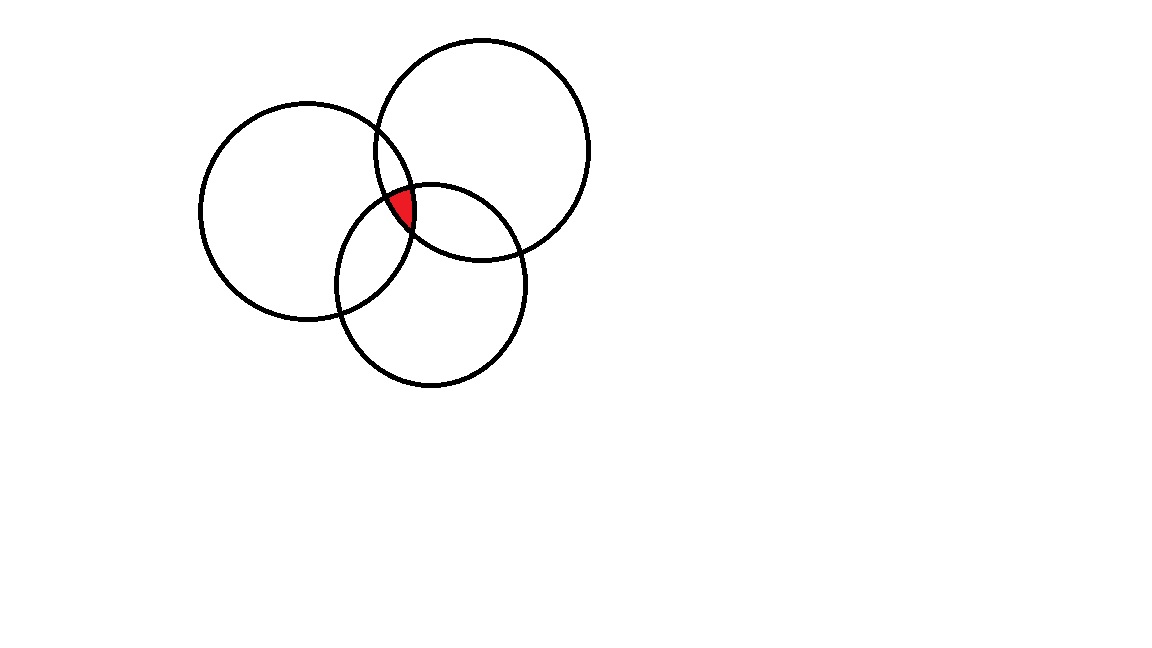
There are several of these systems currently in operation. The most famous of these being the U.S. funded Global Positioning Service (GPS) however there are others such as the Russian Federation’s GLONASS and soon the European Galileo [11]. These were devised as systems to enable users worldwide to calculate their current location. The GPS system for example comprises of twenty-four satellites continuously producing signals that tell a user where a satellite is and highly accurate timing information about when this signal was produced. All twenty-four satellites communicate on the same two frequencies using Code Division Multiple Access in order to avoid collisions. By listening to the signals generated by at least four of the satellites to discover their locations and calculating the time it has taken to reach the user and hence how far away they are from each the users own location can be calculated. [12]

The accuracy of GPS is roughly 3.5m on the horizontal plane without using additional information. There are other systems which can be used in conjunction with GPS to provide up to millimetre accuracy however 3.5m should be accurate enough for this project [13]. The accuracy of Galileo is expected to improve to the centimetre range [14] once it is completed in 2020 [15].

There are two main limiting factors however: the accuracy of the GPS receiver in the user’s phone and that much of the time the user will be indoors. The accuracy of the receiver in a typical mobile phone is roughly 2-3m under good multipath conditions or up to 10m under bad multipath conditions [16]. Multipath is when the signal may reach the receiver via many paths for example direct from the satellite compared to being reflected by a building in between the satellite and receiver. This means that the signal will degrade when near buildings due to the weak multipath suppression available on mobile phone antennae. The app will always be expected to be running in an urban landscape therefore meaning the GNSS location may have quite a low accuracy. Receiving GNSS signals indoors can become a challenge as the signal when received outdoors is already very weak when compared with the background noise and the signal can be 10-100000 times weaker when indoors [17]. This can make it difficult to receive the necessary signals from four satellites at once making it near impossible to reliably and quickly determine location indoors via this method.

### Cell Tower Based Location

Location can be calculated using cell towers in a similar way to that of GNSS. Essentially the phone sends a ping to the nearby cell towers and times how long it takes to receive a response. If the locations of the cell towers are known then an estimate of the phone’s location can be calculated using the respective distances from each cell tower [18]. An example of how this works can be seen below in Figure 2 Cell Tower Triangulation.



1

3

2

Figure 3 Cell Tower Triangulation

In the figure above the arrows represent the calculated distance from the cell tower. With only one cell tower response this means the user could be anywhere in the surrounding circle. This is as the furthest the user could possibly be from the tower would be is a straight line from it (the radius of the circle) however there is no guarantee that is the path being taken. Instead it could be reflecting off of buildings or other materials meaning the user could be anywhere within the surrounding circle. With responses from more cell towers the location becomes more accurate. As can be seen in Figure 2 Cell Tower Triangulation with these responses the user could only be in the overlapping area from each circle.

Cell Tower Triangulation could potentially be a more reliable method of geolocation as it even tends to work indoors, presuming there is a mobile phone signal, and even works better in more urban areas where this application would be expected to be used, due to the higher number of cell towers [18]. It does however have some drawbacks in that it requires an accurate knowledge of cell tower locations which are not typically available publically making this more difficult [19]. On average for example in the United States of America (USA) the average accuracy of a three tower triangulation is 0.75 square miles [20]. This is far too inaccurate for the purposes of this application.

### Wi-Fi Based Location

In much the same way cell towers can be used to calculate the user’s location so can Wi-Fi base stations. By using easily available data such as signal strength a rough estimate of distance from the base station can be calculated. If several base stations can be identified, then the user’s location can be calculated using triangulation.

### Current Android Solution

Android as an OS has several built in features for detecting the user’s current position and provides app developers with the opportunity to use this data. It currently uses a combination of all 3 of the location services previously mentioned [4]. The accuracy while outside is particularly high as can be seen in

TODO

This is likely due to the availability of a strong GPS signal in combination with cell tower and Wi-Fi location services. When inside however the provided location services are far weaker as can be seen in Figure 4.

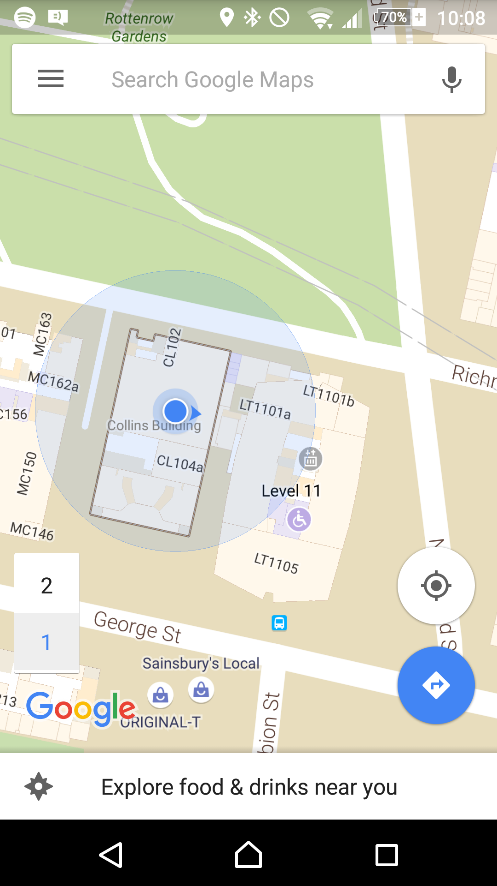


Figure 4 Indoor Location Example

In Figure 4 again the reported location is shown with a small blue circle this can be compared to the actual location represented by the red circle to show that when indoors there can be a large margin of error. The larger blue circle represents the area in which google admits the location may fall within. Although technically the actual location does fall within this area the data is still almost unusable from the perspective of this app. The area of uncertainty shown above covers an entire building as well as significant amounts of two neighbouring buildings meaning this is unlikely to be useful when trying to navigate within one building. This large degree of uncertainty and inaccuracy is most likely due to a lack of a reliable GPS signal combined with a low knowledge of the exact Wi-Fi and Cell Tower locations. This should be improving however as google is trying to improve its knowledge of Wi-Fi locations using their massive user base. This is being done by taking the times when a user has a good knowledge of their own location using GPS to send google information on the surrounding Wi-Fi signals and strengths [21]. This could eventually theoretically provide strong indoor locations worldwide however is currently falling short of this. Even if it does improve using GPS as the starting point means height will always likely be a problem meaning differentiating between floors of a building could be difficult.

### Improved Suggestion

As previously stated it is believed that the main problem currently facing indoor location services is a knowledge of the locations of different Wi-Fi sources. Although steps are being made to try to improve this globally within a smaller area that is to be mapped such as a University campus this problem becomes far easier to deal with. Within this area the

locations could be found manually preferably by using documentation written during the installation of a Wi-Fi network however if not a Wi-Fi map could potentially be built up when the app is being built.

## Routing

In order to build an application that will provide a route for the user between their chosen start and destination a graph representing the area to be mapped must be built. Once the graph has been built a method for finding the shortest path between two nodes in that graph must be found to indicate the route that the user should follow.

### Representing a Graph

There are three main ways of representing a graph within a computer system Edge List, Adjacency Matrix and Adjacency List [22].

#### Edge List

Consists of a list of all of the edges in the system. This is a fairly easy to understand representation of a graph however can be difficult to traverse during path finding algorithms. By indexing the information by edges rather than nodes it can make moving from one node to another more difficult. Each time it is needed to check what edges are connected to the current node it is necessary to move through all of the edges checking whether or not that node is attached to that particular edge.

#### Adjacency Matrix

Consists of a Matrix with the nodes along both axes and the weight of the edge connecting them at the point where they meet. An example of this can be seen in Figure 5 and Figure 6.

Figure 5 Example Graph

4

C

6

5

D

1

A

B

2

2

E

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| A | Null | 2 | 4 | Null | 1 |
| B | 2 | Null | Null | Null | 2 |
| C | 4 | Null | Null | 6 | Null |
| D | Null | Null | 6 | Null | 5 |
| E | 1 | 2 | Null | 5 | Null |

Figure 6 Example Adjacency Matrix

Adjacency matrices are typically used in very densely populated graphs i.e. those with a lot of edges per nodes. This is as the number of edges does not affect the storage space required for this implementation however if the graph is sparser with each node having far fewer edges than there are other nodes this method becomes increasingly inefficient with almost the entire matrix being filled with null values to represent no edge.

#### Adjacency List

This is a method in which each node has a list of the nodes it is connected to. By storing the data in this way it makes routing algorithms far simpler as it is very easy to quickly determine which nodes are connected to the one that is currently being examined. Naturally this does not support edge weights however this functionality can be added via a slight modification. By storing a list of edges for each node this provides all of the benefits of an Adjacency List however also allows for weights to be added to the connections allowing for a representation of a weighted graph.

### Path Finding

Once the graph representation has been constructed an algorithm for finding the shortest path must be implemented. There are many algorithms that have been developed for finding the shortest path between two nodes in a graph and each have their own areas of expertise depending on the type of graph.

#### Breadth-First Search

This algorithm starts at one of the two nodes it is to find the route between. From there it then moves out to all of the surrounding nodes and assigns them a value of one. It then takes each of the nodes with a value of one and assigns all of the surrounding nodes with a value of two. This continues until it reaches the desired node. It is guaranteed to find the path between the two in the fewest number of hops however does not take into account edge weights.

By not taking into account differing edge weights it would make constructing a graph representing the campus to be mapped far more difficult. This is as each of the nodes would have to take roughly the same amount of time to walk between to produce an acceptable route. This becomes particularly difficult when adding lifts to the routes as waiting on lifts could potentially take several minutes. Waiting times like this mean either all of the nodes would have to be particularly far apart, which is likely not possible in a corridor with many rooms coming off of it, or there would have to be conceptually many nodes the user would move between while waiting on a lift which is essentially just adding weights in a very inefficient way.

#### Dijkstra’s Algorithm

The main problem with git

## Software Engineering

Basic Principles

Model View Controller

Design Patterns

# Design

## Initial Design

## Final Design

## Reason for changes

## Software patterns

# References

|  |  |
| --- | --- |
| [1] | P. Viswanathan, “about tech,” about, 15 December 2014. [Online]. Available: http://mobiledevices.about.com/od/additionalresources/a/Native-Apps-Vs-Web-Apps-Which-Is-The-Better-Choice.htm. [Accessed 08 March 2016]. |
| [2] | w3schools, “w3schools,” [Online]. Available: http://www.w3schools.com/html/html5\_geolocation.asp. [Accessed 8 March 2016]. |
| [3] | A. De Rosa, “envatotuts+,” 8 August 2014. [Online]. Available: http://code.tutsplus.com/tutorials/html5-network-information-api--cms-21598. [Accessed 8 March 2016]. |
| [4] | Google, “Location Strategies,” [Online]. Available: http://developer.android.com/guide/topics/location/strategies.html. [Accessed 8 March 2016]. |
| [5] | International Data Corporation, “Smartphone OS Market Share, 2015 Q2,” [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp. [Accessed 8 March 2016]. |
| [6] | Google, “Google Maps SDK for iOS,” [Online]. Available: https://developers.google.com/maps/documentation/ios-sdk/. [Accessed 8 March 2016]. |
| [7] | The Bing Maps Team, “New Bing Maps iOS SDK,” Microsoft, 5 May 2011. [Online]. Available: https://blogs.bing.com/maps/2011/05/05/new-bing-maps-ios-sdk/. [Accessed 8 March 2016]. |
| [8] | Google, “Google Maps Android API,” [Online]. Available: https://developers.google.com/maps/documentation/android-api/. [Accessed 8 March 2016]. |
| [9] | R. Brundritt, “Bing Maps Android SDK available on Codeplex,” 31 March 2011. [Online]. Available: https://blogs.bing.com/maps/2011/03/31/bing-maps-android-sdk-available-on-codeplex/. [Accessed 8 March 2016]. |
| [10] | 4 March 2016. [Online]. Available: https://www.google.co.uk/maps/dir/Glasgow/London,+UK/@53.6628863,-4.4369792,7z/data=!3m1!4b1!4m13!4m12!1m5!1m1!1s0x488815562056ceeb:0x71e683b805ef511e!2m2!1d-4.251806!2d55.864237!1m5!1m1!1s0x47d8a00baf21de75:0x52963a5addd52a99!2m2!1d-0.1277583!2d51.507350. |
| [11] | M. Rouse, “TechTarget,” September 2005. [Online]. Available: http://searchnetworking.techtarget.com/definition/GNSS. [Accessed 7 March 2016]. |
| [12] | E. K. a. C. Hegarty, “GPS Overview,” in *Understanding GPS: Principles and Applications*, Artech House, 2005, pp. 3-4. |
| [13] | GPS.gov, “GPS Accuracy,” U.S. Air Force, 18 September 2014. [Online]. Available: http://www.gps.gov/systems/gps/performance/accuracy/. [Accessed 7 March 2016]. |
| [14] | T. Bowler and M. Wall, “BBC News,” BBC, 26 August 2014. [Online]. Available: http://www.bbc.co.uk/news/business-28779524. [Accessed 7 March 2016]. |
| [15] | European Space Agency, “European Space Agency,” 18 December 2015. [Online]. Available: http://www.esa.int/Our\_Activities/Navigation/The\_future\_-\_Galileo/What\_is\_Galileo. [Accessed 7 March 2016]. |
| [16] | K. M. Pesyna, Jr, R. W. Heath, Jr and T. E. Humphreys, “GPS World,” 2 February 2015. [Online]. Available: http://gpsworld.com/accuracy-in-the-palm-of-your-hand/. [Accessed 7 March 2016]. |
| [17] | iPosi, “iposi.com,” 2014. [Online]. Available: http://www.iposi.com/tech-talk/the-technology. [Accessed 7 March 2016]. |
| [18] | D. Vaughan, “Geographic Location,” in *Windows Phone 7.5 Unleashed*, Sams Publishing, 2012, pp. 529-552. |
| [19] | J. Yang, A. Varshavsky, H. Liu, Y. Chen and M. Gruteser, “Accuracy Characterization of Cell Tower Localization,” UbiComp '10, New York, 2010. |
| [20] | Federal Communications Commission, “Accurate Location Detection 911 Help SMS App,” 2015. |
| [21] | S. J. Vaughan-Nichols, “ZDNet,” 16 November 2011. [Online]. Available: http://www.zdnet.com/article/how-google-and-everyone-else-gets-wi-fi-location-data/. [Accessed 9 March 2016]. |