**Capture the Campus!**

**Initial Report**

Submitted for the BSc in   
Computer Science

October 2016

By

**Alexander C Whitehead**

Word Count:

Table of Contents

[1 Introduction 3](#_Toc463984529)

[2 Background 4](#_Toc463984530)

[2.1 Platform 4](#_Toc463984531)

[2.2 Language 5](#_Toc463984532)

[2.3 Server 5](#_Toc463984533)

[2.4 Examples 6](#_Toc463984534)

[3 Aim and Objectives 7](#_Toc463984535)

[4 Task List 8](#_Toc463984536)

[5 Time Plan 10](#_Toc463984537)

[6 Risk Analysis 12](#_Toc463984538)

[Appendix A: Capture the Campus! 14](#_Toc463984539)

[References 15](#_Toc463984540)

# Introduction

This is a report describing the initial design and research stages of the project to build an augmented reality, mobile platform based game.

The game, called Capture the Campus, is based on a combination of the classic 80’s arcade game Qix wherein players control a character around a box taking squares of area while avoiding enemies (System 16, 2014), the currently popular mobile game Pokémon GO where players’ transverse the real world in search of virtual pets (Niantic, 2016) and Ingress the predecessor to Pokémon GO which uses map data and landmarks in a capture the flag style game (Niantic, 2016).

The objective of the game will be to either in teams or individually capture parts of a predefined area by physically traveling through it while the game-running on a mobile device in the players’ possession-tracks the player’s movement.

To implement functionality a client and server should be created. The client will be created using Xamarin as it allows for the development of a cross platform solution that is easily debugged for rapid prototyping (Xamarin Inc., 2016). The server will be created using C# or Xamarin as this will allow for the server to be run from both a desktop or mobile device, this is desirable as it allows the use of a predefined external server or alternatively an ‘on the fly’ local server run from the client’s mobile device. The server will use both UDP and TCP protocols, UDP will be used to acquire a connection to the server from the client as it has low latency and general broadcast ability and TCP will be used once this connection has been attained as it is more secure and less likely to lose packets (Diffen, n.d.).

This report includes: A section discussing background research conducted and creative/contextual decisions made thusly, a section covering the aims and objectives of the project, a breakdown of the intended task list and time required to complete each individual task and a risk analysis section which will identify and then attempt to rectify any risks associated with the intended project.

# Background

## Platform

Firstly, it must be decided which platform an application will be run on. In this case it must be a mobile platform as it would be impossible to move something like a desktop computer around in order to play, also they do not usually contain GPS tracking hardware. The options for mobile platforms witch contain GPS tracking hardware are limited, they include; Android phones, iOS phones, Windows phones and GPS trackers.

Out of the choice of mobile platforms GPS trackers can be disregarded almost immediately as the software should ideally be easily distributed and in order to run the software with dedicated GPS trackers the trackers would probably have to come bundled with the software.

This leaves the choice of the three main mobile phone operating systems; Android, iOS and Windows. Windows can be eliminated because it has such a small market share (Statista, 2016).

Figure 1: This image shows the current market share of mobile operating systems (Statista, 2016).

This leaves Android and iOS.

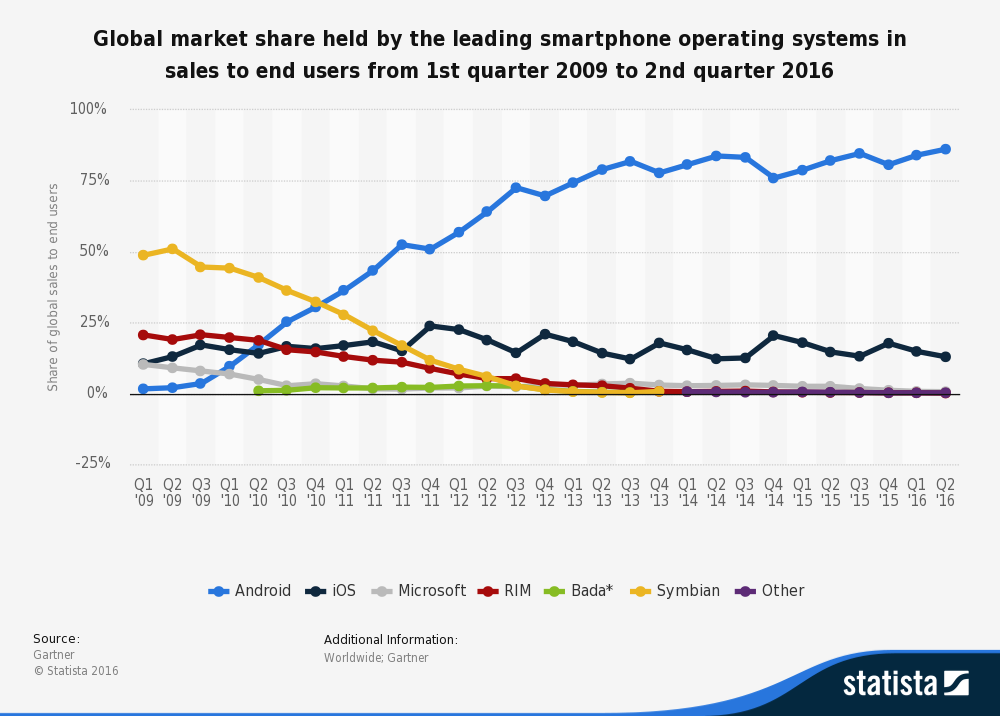


Figure 1: This image shows the current market share of mobile operating systems (Statista, 2016)

It is also relatively easy to eliminate iOS from the running as it is usually required that you be using a Mac to develop for it whereas it is possible to program on both a Mac and PC for Android, it is also much easier to publish and design for an Android device (Sinicki, 2016).

## Language

In order to begin designing a project a language must be chosen to develop it in, for Android there is the option of developing in Java using Android Studio, in C# using Visual Studio or in a multitude of none native solutions.

Both Java and Xamarin take advantage of native features of the Android operating system (Google Inc., n.d.) (Xamarin Inc., 2016) and are therefore more suitable than none native solutions but no more suitable than each other for native Android development. However, Xamarin does allow for the use of standard .NET libraries (Montemagno, 2016) and also allows for the porting of existing code from Android to either iOS or Windows (Xamarin Inc., 2016) thus increasing the speed and efficiency of work flow and also opening up the theoretical user base that the product can appeal to from 86.2% to 99.4% (Statista, 2016).

## Server

It is required that each instance of the game communicate with each other, to facilitate this a server will be required to store information about each player and pass it on when required.

There are a number of protocols that the server could be written to accept, each with its own advantages and disadvantages in a given scenario. Protocols include; UDP (User Datagram Protocol) a low latency, low reliability protocol (TechTarget, 2015) and TCP (Transmission Control Protocol) a high latency (compared to UDP), high reliability connection-oriented protocol (TechTarget, 2014).

Because of its low latency and general broadcast ability UDP is perfect for gaining an initial connection to a server from a client (Mey, n.d.), TCP is then useful once the initial connection has been verified to pass sensitive game data.

There are also a number of models of how and where a server can be implemented and how and what will send and receive information where. Models include; peer-to-peer whereby each instance on a network shares its files equally with each other with no central storage or authentication (Posey, 2000) and client/server where there are separate dedicated servers and clients (Posey, 2000).

Figure 2: This image shows a visual model of the difference between a client/server model on the left and a peer-to-peer model on the right (Philips, 2014).

Doom by id Software is an example of a game that uses a peer-to-peer model to handle its multiplayer networking. Doom was regularly mocked for its poor performance, especially with larger numbers of peers as all connections are bottlenecked by the slowest connection (Anon., n.d.). Conversely Quake-also developed by id Software-is an example of a client/server model (Anon., n.d.). Client/server handles larger numbers of users with ease as each person’s connection is only as slow as their own connection to the server, but for this to work it is required that an offsite server/s be dedicated to the game 24/7.

For this project a client/server model would be most efficient as there could possibly be an infinite number of players at any one time, also most people will be connecting to the network using mobile data so drops in connection will occur regularly.

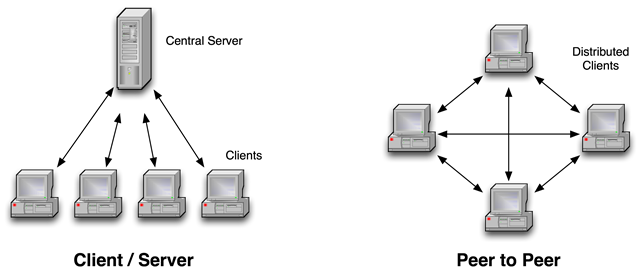


Figure 2: This image shows a visual model of the difference between a client/server model on the left and a peer-to-peer model on the right (Philips, 2014)

## Examples

Some examples of similar games to the intended project that were commercial successes would be games like Pokémon GO which use augmented reality and GPS to recreate the experience of playing the game Pokémon in the real world (Niantic, 2016). Another augmented reality game which influenced Pokémon GO would be Ingress by the same developers, this game had the influential inclusion of allowing the players to set waypoints in the game to be used by other players (Niantic, 2016).

However, some people do not believe that Pokémon GO and Ingress are perfect and that there could be lessons learnt from their mistakes (Credits, Extra, 2016).

# Aim and Objectives

# Task List

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Name** | **Description** | **Duration**  **(weeks)** |
| 1 | Research | Conduct research that will aid in the writing of reports and initial designing of the project | 11 |
| 2 | Initial report | Write the initial report deliverable | 2 |
| 3 | Create server client | Create a client that is capable of interfacing with the server | 5 |
| 4 | Create TCP server | Create a TCP server that can accept TCP packets from the client | 2 |
| 5 | Add UDP to server | Add UDP to the server that can be used to identify the IP if the server | 1 |
| 6 | Add multithreading capabilities to server | Add multithreading to the server so that it is capable of accepting more than one client request at a time | 2 |
| 7 | Create main menu for game | Create a main menu for the game that will be displayed when the game is started and between every game instance. The main menu should display all options for game types and settings etc. | 1 |
| 8 | Add game screen and assets to game | Add a game screen to the game that is displayed once the play game option is selected and also add assets to the game to be used to display player characters etc. | 2 |
| 9 | Add map to game screen | Add a suitable map to the game screen | 3 |
| 10 | Add translations and scaling to map | Add translations and scaling to the map so that it is possible to move the map around and zoom in and out | 2 |
| 11 | Add player character and movements to map | Make the player character move as the player moves. This should probably work via GPS. | 2 |
| 12 | Interim report | Write the interim report deliverable | 3 |
| 13 | Final report | Write the final report deliverable | 14 |
| 14 | Add client calls to store and recall player positions from server | Make the game send its current location to the server at a reasonable interval and also make it so that the game requests the location of every other player | 3 |
| 15 | Add tracking data and bounds of playing field to game | Make it so that the game then draws all the players at the correct locations and that it is possible to create the area of play | 2 |
| 16 | Add taking mechanics from tracking data to game | Make it so that when a player completes a run from one side of the playing area to another they take the smallest area for their own team | 2 |
| 17 | Add killing mechanics from tracking data | Make it so that if a player crosses the track of another active player one of the players dies | 2 |
| 18 | Add scoring data to game | Make it so that the score of all players is tracked based on the area of land taken | 2 |
| 19 | Add team mode to game (Optional) | Make it so that players can play in teams. This is an optional extra if development is kept to schedule | 3 |
| 20 | Add multigame server (Optional) | Make it so that multiple game instances can run on one server. This is an optional extra if development is kept to schedule | 3 |

# Time Plan

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Task Name** | **University Calendar Weeks** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 |
| 1 | Research |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Initial report |  |  |  | D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Create server client |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Create TCP server |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Add UDP to server |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 | Add multithreading capabilities to server |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 | Create main menu for game |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | Add game screen and assets to game |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 | Add map to game screen |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 | Add translations and scaling to map |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11 | Add player character and movements to map |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 12 | Interim report |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 13 | Final report |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | D |
| 14 | Add client calls to store and recall player positions from server |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 15 | Add tracking data and bounds of playing field to game |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 16 | Add taking mechanics from tracking data to game |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 17 | Add killing mechanics from tracking data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 18 | Add scoring data to game |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 19 | Add team mode to game (Optional) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 20 | Add multigame server (Optional) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# Risk Analysis

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Risk** | **Current Risk** | | | **How to Avoid** | **How to Recover** | **Residual Risk** | | |
| **Severity**  **(L/M/H)** | **Likelihood**  **(L/M/H)** | **Significance (Sev. x Like.)** | **Severity**  **(L/M/H)** | **Likelihood**  **(L/M/H)** | **Significance (Sev. x Like.)** |
| Data loss | H | M | HM | Keep Backups | Reinstate from backups | L | M | LM |
| Loss of backups | H | L | HL | Multiple Backups | Use alternate | L | L | LL |
| Underestimate workload | H | M | HM | Regularly review progress against Time Plan | Invest more time into work, possible reduction of objectives | H | L | HL |
| Critical error in deliverable | H | M | HL | Perform adequate research | Debug code | H | L | HL |
| Skill Risk | M | M | MM | Perform adequate training | Invest more time into research | L | L | LL |
| Scope Creep | M | H | MH | Fully define scope | Define scope at current point | M | L | ML |
| Inefficient Program Performance | H | L | HL | Spend time testing code | Remove extraneous features | M | L | ML |
| Server Crashes | M | M | MM | Attempt to optimize server code | Implement alternative servers | L | L | LL |
| Incompatible with target device | H | L | HL | Create program with target device specifications in mind | Create alternative version for target device | L | L | LL |
| Medical emergency | H | L | HL | Care for developers health | Comment code regularly so that it is well understood | M | L | ML |
| Development software unavailable | H | H | HH | Place key dll into SVN | Download key dll | L | L | LL |
| User injured while using application | H | M | HM | Only allow authorized access to application while in development and provide warning messages while in play | Medical attention may be required if a user is injured while using the application | H | L | HL |
| Optional extras are not complete | L | M | LM | Keep to Time Plan | Ensure deliverable is acceptable | L | L | LL |

1. Capture the Campus!

Capture the flag is a well-known game (sub-) genre that requires players to capture the flag. Often there are two opposing teams each of which has a flag that must be defended from the other team.

<http://en.wikipedia.org/wiki/Capture_the_flag>

This project requires the student to design and develop an augmented reality game for a GPS-enabled mobile device (preferably Windows Phone 7). The details of the gameplay are open to negotiation, but a suggestion is that a number of flags (or capture points) are distributed in GPS locations around the campus. The players are required to visit the location for a specified time period to capture the location. The players accumulate score based on number of capture points held and time they are uncontested for. The status of the capture points should be persistent, with the game’s progress being able to be tracked over multiple (perhaps unlimited) days.

The project involves databases for storing flag locations and capture logs etc.

Project Code: DJP3

References

Anon., n.d. *Doom networking component.* [Online]   
Available at: http://doom.wikia.com/wiki/Doom\_networking\_component  
[Accessed 11 October 2016].

Anon., n.d. *Quake engine.* [Online]   
Available at: https://en.wikipedia.org/wiki/Quake\_engine#Network\_play  
[Accessed 11 October 2016].

Credits, Extra, 2016. *Improving on Pokemon GO - Making Better Augmented Reality Games - Extra Credits.* [Online]   
Available at: https://www.youtube.com/watch?v=94KwB205DDk  
[Accessed 11 October 2016].

Diffen, n.d. *TCP vs. UDP.* [Online]   
Available at: http://www.diffen.com/difference/TCP\_vs\_UDP  
[Accessed 11 October 2016].

Google Inc., n.d. *Getting Started with the NDK.* [Online]   
Available at: https://developer.android.com/ndk/guides/index.html  
[Accessed 11 October 2016].

Mey, M. D., n.d. *Network discovery using UDP Broadcast (Java).* [Online]   
Available at: http://michieldemey.be/blog/network-discovery-using-udp-broadcast/  
[Accessed 11 October 2016].

Montemagno, J., 2016. *.NET Standard Library Support for Xamarin.* [Online]   
Available at: https://blog.xamarin.com/net-standard-library-support-for-xamarin/  
[Accessed 11 October 2016].

Niantic, 2016. *Ingress.* [Online]   
Available at: https://www.ingress.com/  
[Accessed 11 October 2016].

Niantic, 2016. *Pokémon GO.* [Online]   
Available at: http://www.pokemongo.com/en-uk/  
[Accessed 11 October 2016].

Philips, B., 2014. *The P2P Witch Hunt.* [Online]   
Available at: http://blog.peer5.com/the-p2p-witch-hunt/  
[Accessed 11 October 2016].

Posey, B., 2000. *Understanding the differences between client/server and peer-to-peer networks.* [Online]   
Available at: http://www.techrepublic.com/article/understanding-the-differences-between-client-server-and-peer-to-peer-networks/  
[Accessed 11 October 2016].

Sinicki, A., 2016. *Developing for Android vs developing for iOS – in 5 rounds.* [Online]   
Available at: http://www.androidauthority.com/developing-for-android-vs-ios-697304/  
[Accessed 11 October 2016].

Statista, 2016. *Global mobile OS market share in sales to end users from 1st quarter 2009 to 1st quarter 2016.* [Online]   
Available at: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/  
[Accessed 11 October 2016].

System 16, 2014. *TAITO QIX HARDWARE.* [Online]   
Available at: http://www.system16.com/hardware.php?id=633  
[Accessed 11 October 2016].

TechTarget, 2014. *TCP (Transmission Control Protocol).* [Online]   
Available at: http://searchnetworking.techtarget.com/definition/TCP  
[Accessed 11 October 2016].

TechTarget, 2015. *UDP (User Datagram Protocol).* [Online]   
Available at: http://searchsoa.techtarget.com/definition/UDP  
[Accessed 11 October 2016].

Xamarin Inc., 2016. *Building Cross Platform Applications.* [Online]   
Available at: https://developer.xamarin.com/guides/cross-platform/application\_fundamentals/building\_cross\_platform\_applications/  
[Accessed 11 October 2016].

Xamarin Inc., 2016. *Using Native Libraries.* [Online]   
Available at: https://developer.xamarin.com/guides/android/advanced\_topics/using\_native\_libraries/  
[Accessed 11 October 2016].