CS3821
FULL UNIT PROJECT
BUILDING A GAME

YEAR 3 FINAL PROJECT

BY

PAVAN JANDO

October 5, 2018

SUPERVISOR
JASPER LYONS

Contents

1 Abstract

1.1 The Game

I plan to make an android game that uses GPS location tracking where the player can encounter in-game enemies to battle and level up their character. The user can pick from a selection of characters which vary in abilities and play-styles. Battling AI enemies gains experience which levels their character and increases their statistics e.g. strength, speed. Enemies also have a chance of dropping items that the user can equip, utilise or trade with other players.

Users should also be able to interact with each other by battling, trading or playing co-operatively. They can do this by grouping together for boss battles which generally would be in more popular areas such as Royal Holloway. These enemies should be stronger and more difficult to defeat but also more rewarding for defeating them.

1.2 Aims

The aim of this project is to develop and deploy a playable game whilst learning about the game development ecosystem. I will also explore the android life-cycle and how mobile games differ from PC games. There are a lot more factors to be concerned about such as battery life and performance when using less powerful hardware. I want this project to be representative of my willingness to learn technologies that are new to me and my efficiency in doing so. Knowing how to use platforms like Android Studio and programming in Unity are important skills when entering the games industry.

Furthermore, I will also learn about the different design patterns that are most commonly used in games such as Flyweight, Factory, Singleton and Game Loop which are just a few of many. Efficiency and well-designed algorithms are key to making a game run smoothly without affecting the user experience. For example, Flyweight is important for use in games where many objects need to be displayed all at once. It takes the similarities between these objects and extracts it to its own class. Therefore, less data is being processed and a load is taken of the CPU/GPU due to many objects sharing from the same set of characteristics. [4]

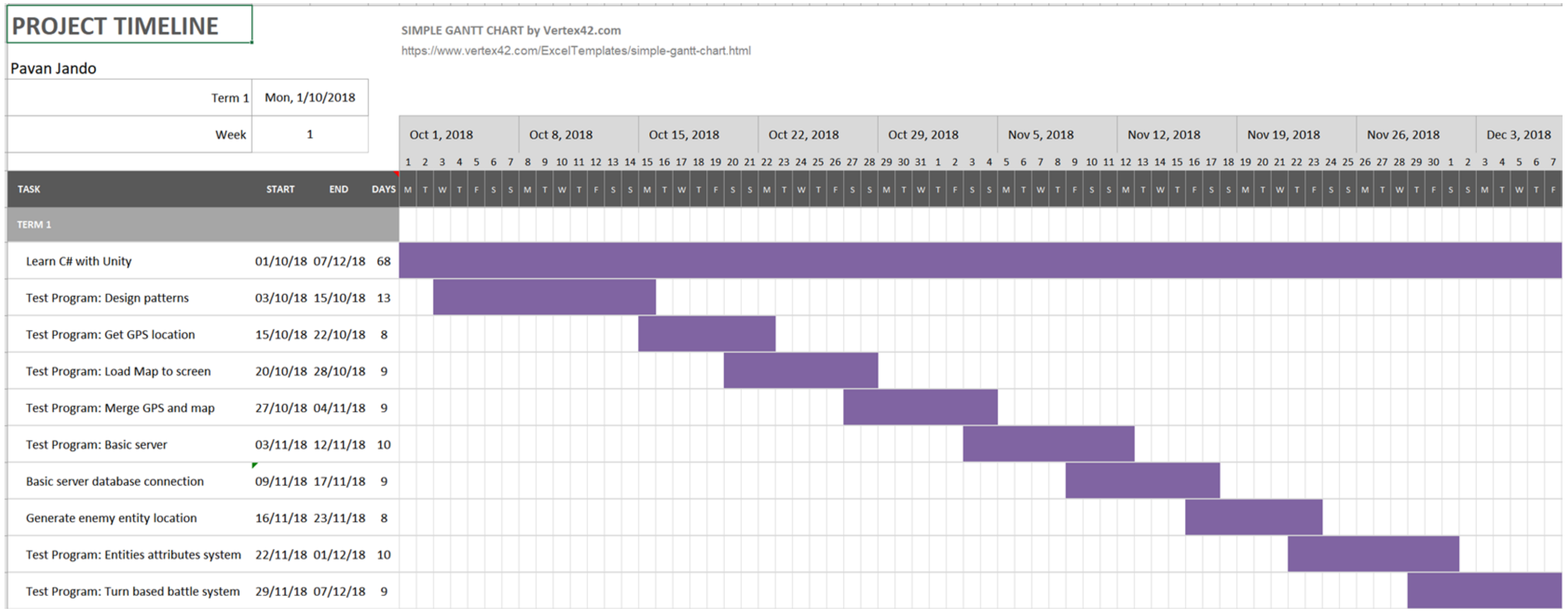
1.3 Proof of Concepts

The end goal of my game is to have many players connecting to a dedicated server which then retrieves and stores data about theirs and other characters in a database. This will allow players to interact with each other via trading and battling. I will need to create a basic client-server model to show that I can transmit data between the client and server then also store this information in a database. A difficult part of this will be organising and scheduling requests the server gets from multiple clients and ensuring that data does not get lost in transmission. For this I will need to create a thread for each client connecting to the server to allow for continuous connection and disconnection of clients [9].

Some popular developing platforms for android games include Unity, Corona SDK, Unreal Engine 4 and Amazon's Lumberyard to name a few [2]. For my needs, the users character model will be 3D, animated to walk or run depending on the user's speed whilst moving. Corona SDK is limited to 2D development and does not help me fulfil my requirements. After doing some further research I concluded that Unity would be the ideal engine to use as it is very popular among android developers meaning there is more resources to learn from [8]. Lumberyard is relatively new and has not yet been widely adopted as the likes of Unity. Unreal Engine is a good alternative but again I felt that many sources encouraged Unity especially for beginner game developers such as myself.

Another decision that needed to be made was which mapping software would be ideal. I narrowed my search to OpenStreetMap (OSM) and Google Maps (GM) as I felt these were the most popular mapping tools used by developers for their accuracy and large amounts of data. The main difference between the two is that OSM is open versus GM being closed source. This mainly means there is less control over data and any customisations with using GM over OSM [1]. I discovered that OSM API is not as clear and easy to read as the GM API [5]. However, I was able to find some examples to learn from which were clearer [6]. Taking these factors into consideration I have decided to use OSM for its superior control of data and customisations.

2 Timeline



PROJECT TIMELINE

Pavan Jando

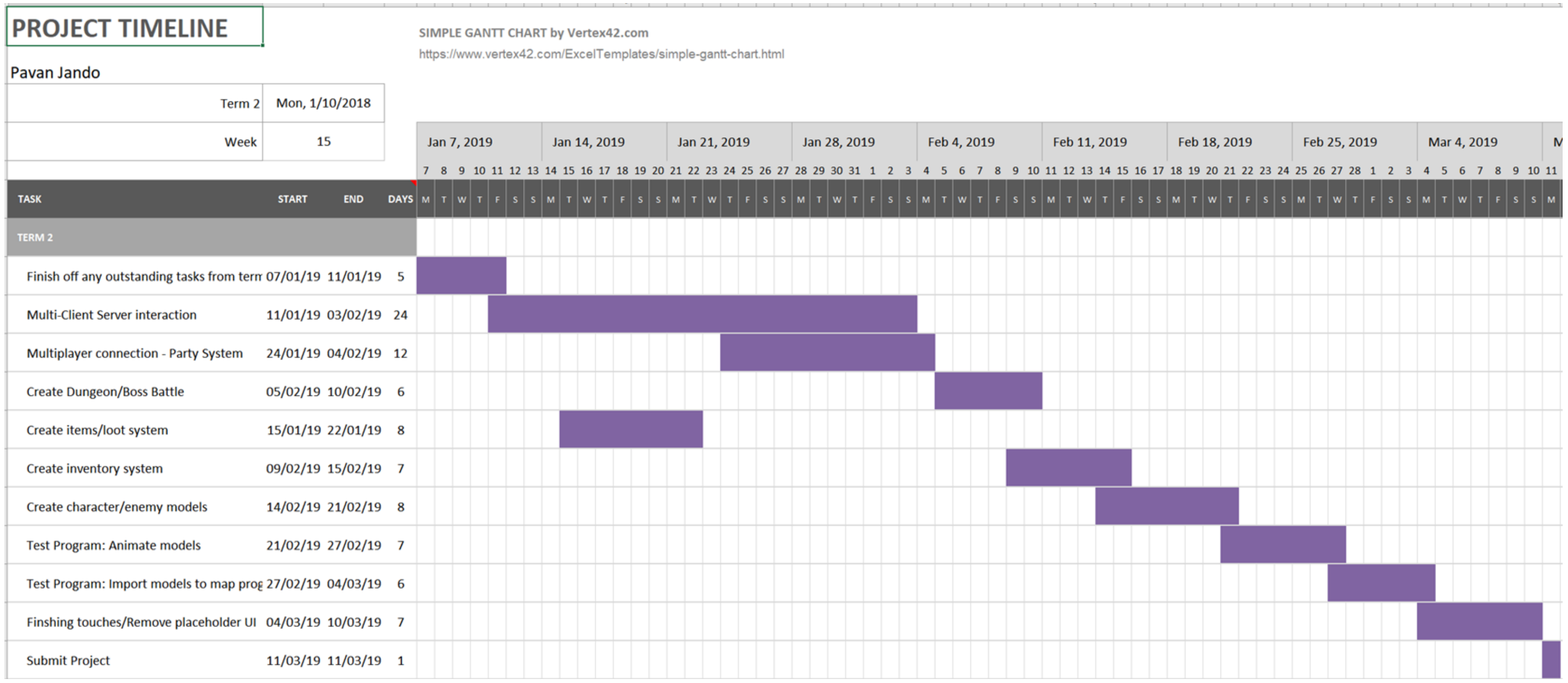
SIMPLE GANTT CHART by Vertex42.com

https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Term 1	Mon, 1/10/2018
Week	1

TASK	START	END	DAYS
TERM 1			
Learn C# with Unity	01/10/18	07/12/18	68
Test Program: Design patterns	03/10/18	15/10/18	13
Test Program: Get GPS location	15/10/18	22/10/18	8
Test Program: Load Map to screen	20/10/18	28/10/18	9
Test Program: Merge GPS and map	27/10/18	04/11/18	9
Test Program: Basic server	03/11/18	12/11/18	10
Basic server database connection	09/11/18	17/11/18	9
Generate enemy entity location	16/11/18	23/11/18	8
Test Program: Entities attributes system	22/11/18	01/12/18	10
Test Program: Turn based battle system	29/11/18	07/12/18	9

Figure 1: Term 1 Gantt Chart



PROJECT TIMELINE

SIMPLE GANTT CHART by Vertex42.com

https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Pavan Jando

Term 2	Mon, 1/10/2018
Week	15

TASK	START	END	DAYS
TERM 2			
Finish off any outstanding tasks from terr	07/01/19	11/01/19	5
Multi-Client Server interaction	11/01/19	03/02/19	24
Multiplayer connection - Party System	24/01/19	04/02/19	12
Create Dungeon/Boss Battle	05/02/19	10/02/19	6
Create items/loot system	15/01/19	22/01/19	8
Create inventory system	09/02/19	15/02/19	7
Create character/enemy models	14/02/19	21/02/19	8
Test Program: Animate models	21/02/19	27/02/19	7
Test Program: Import models to map prog	27/02/19	04/03/19	6
Finshing touches/Remove placeholder UI	04/03/19	10/03/19	7
Submit Project	11/03/19	11/03/19	1

Figure 2: Term 2 Gantt Chart

2.1 Milestones

As shown in my Gantt chart, each "Test Program" is a milestone for me as it shows that I am able to learn a new concept or use a new technology. With these milestones completed it is proof that I am able to do a lot of the work that is required to make this game. For example, being proficient in C# using the Unity environment is a key skill to that needs to be of a high enough standard to complete the project. This is why I will be using it in the "Test Programs" that have been labelled.

An essential milestone for me is to be able to use the fundamental game design patterns in a language I have not used before. This will be proof of my ability to confidently program using C#. Such a pattern is the Game Loop which exists in nearly all games. One iteration of the loop represents one frame being drawn according to any user input [7]. It updates the game state and renders it to the screen. Therefore it is crucial each iteration is fast to provide a smooth user experience.

I have divided the work load across the two terms such that single-player will be focused on in the first term and multi-player in the second term. Multi-player is a lot of work to implement and I want to be sure that there is at least a single-player version that can be delivered by the end of the second term.

3 Risk Assessment

The first risk, as in most projects, is falling behind schedule meaning key features do not get developed fully or are dysfunctional. This can happen for a plethora of reasons such as wasting time, over-ambitiousness or difficulty of the task. It is important I try to remain on task ensuring that I focus on the core features of the game and add finishing touches afterwards. I understand that my idea is ambitious and that I need to break down the game into smaller components that need to be completed before moving on to the next. For example, ensuring that the game works as a single-player game before moving towards multi-player. If I fail to keep up with my schedule, I will need to revise my plan and decide which features take priority and need completing first.

As previously discussed, I will be using OSM over GM to get my map data from. The risk is that the API looks difficult to use as it is not as clear as the GM API. After doing more research I found a few more resources that I can learn from to help me progress forward [6]. However, if I decide in future that it is necessary to change mapping software I will create an adapter class to decouple my code base and the mapping software. This way the only thing that needs changing is the adapter class and it should not affect the entire game too much.

My game relies a lot on networking and its ability to communicate with a server to get game updates and interact with nearby players. Therefore, there will be a lot of data transmission from more than one client, requiring high efficiency in data management and no data loss. To prevent building a poor architecture, I will need to utilise design patterns such as the multi-client pattern and use a database to store important character information [3].

There is a risk that I have not planned enough time to implement certain features of my game in the timeline and will result in being incomplete or broken. To avoid this, I have ordered my tasks so that single-player has taken priority and the user can play the game and battle enemies as they journey around. Once this aspect of the game is completed I can move onto making the multi-player and boss levels as I feel that this could take a lot more time than I must implement. In the case that I run out of time to implement a feature I will have to remove it from the final product as it should be in a functional and usable state.

Lastly, a risk is that I could lose my work due to hardware failure which would affect me greatly as the project continues. To prevent this, I will use SVN to store my work, which also allows me to work on a different computer if my home PC fails.

Bibliography

[1] Aleks Buczkowski. Why would you use openstreetmap if there is google maps? `http://geoawesomeness.com/why-would-you-use-openstreetmap-if-there-is-google-maps/`. October 23, 2015.

[2] New Generation Applications Pvt Ltd. Top 5 tools for mobile game development. `https://www.newgenapps.com/blog/top-sdk-tools-for-mobile-game-development`. October 25, 2017.

[3] Aaron McIver. Best approach to designing multi-client applications. `https://softwareengineering.stackexchange.com/questions/34577/best-approach-to-designing-multi-client-applications`. January 7, 2011.

[4] Robert Nystrom. *Game Programming Patterns*. November 2, 2014.

[5] Open-Source. Api v0.6. `https://wiki.openstreetmap.org/wiki/API_v0.6`. August 1, 2018.

[6] Open-Source. How to use the osmdroid library. `https://github.com/osmdroid/osmdroid/wiki/How-to-use-the-osmdroid-library`. May 18, 2018.

[7] Anni Rautakopra. *GAME DESIGN PATTERNS Utilizing Design Patterns in Game Programming*. June, 2018.

[8] Unity. Unity user manual. `https://docs.unity3d.com/Manual/index.html`. October 2, 2018.

[9] Yunming Zhang. Multithreaded server (multiple clients to a single server) programming in java with examples. `https://yunmingzhang.wordpress.com/2013/12/18/multithreaded-server-multiple-clients-to-a-single-server-programming-in-java/`. December 18, 2013.