

School of Computing, Electronics and Mathematics

PRCO304

Final Stage Computing Project

2015/2016

BSc (Hons) Computing & Games Development

Daniel Bendell

10402435

Bayesian’s Probability Controlled Artificial Intelligence in a Traditional Turn Based Strategy Game

Supervisor: Serafim Rodrigues

Second marker: Dan Livingstone

Table of Contents

[Abstract 4](#_Toc450330183)

[1. Introduction 5](#_Toc450330184)

[2. Background and objectives 5](#_Toc450330185)

[3. Approach 6](#_Toc450330186)

[3.1 Unity 6](#_Toc450330187)

[3.2 Visual Studio & Resharper 7](#_Toc450330188)

[3.3 Game Design Document 7](#_Toc450330189)

[3.4 Illustrator 8](#_Toc450330190)

[3.5 Blender 8](#_Toc450330191)

[3.6 Competition Comparison 9](#_Toc450330192)

[3.7 Platform 9](#_Toc450330193)

[3.7.1 Mobile IOS & Android 9](#_Toc450330194)

[3.7.2 Consoles Xbox & PlayStation 9](#_Toc450330195)

[3.7.3 Web & PC 9](#_Toc450330196)

[3.8 Bayesian Probability 10](#_Toc450330197)

[3.9 Architecture 10](#_Toc450330198)

[3.9.1 Characters 10](#_Toc450330199)

[4. Project Management 12](#_Toc450330200)

[4.1 Trello 12](#_Toc450330201)

[4.2 Kanban 12](#_Toc450330202)

[4.3 Source control 13](#_Toc450330203)

[4.4 Project Plan 13](#_Toc450330204)

[4.4.1 Initial Project Plan 14](#_Toc450330205)

[4.4.2 Final Project Plan 15](#_Toc450330206)

[5. Outcome 16](#_Toc450330207)

[5.1 Objective completion 16](#_Toc450330208)

[5.2 Game Design 17](#_Toc450330209)

[5.2.1 Overview 17](#_Toc450330210)

[5.2.2 Game Flow 18](#_Toc450330211)

[5.2.3 Core mechanics 18](#_Toc450330212)

[5.2.4 Bayesian Probability Adaptation 21](#_Toc450330213)

[5.3 Modelling and User Interface 25](#_Toc450330214)

[5.4 Database & API 26](#_Toc450330215)

[6. Testing 26](#_Toc450330216)

[7. Project Post-Mortem 26](#_Toc450330217)

[8. Conclusion 27](#_Toc450330218)

[References 29](#_Toc450330219)

[Appendices 30](#_Toc450330220)

Abstract

This report outlines a game development project with great focus on the underlying Bayesian’s probability artificial intelligence system.

The report starts with a series of research and investigation papers all of which focus on the initial approach towards the project’s needs. The results from each research element are summarized to provide a better understanding of their outcomes. The resulting evidence explains the reasoning behind the technology, artificial intelligence system and processes used throughout the projects lifecycle. Further analysis of the project design is shown to allow the visualization of class structure.

The strict project management tools and work flows are discussed in greater detail, giving insight into successful management techniques used throughout the projects development. The initial and final project plan comparison shows how the management allowed the game to exceed in several different areas.

The steady development processes give an overview into the design, execution and maintaining of the games core mechanics. Descriptions of the games flow, main functionality and artificial intelligence system are provided to give reasoning to each element and how they brought the project together. The complex formulas are broken down to show how the game works around the programming challenges. An evaluation of the AI’s probability allocation, decision making and learning are given.

An overview of the games incremental and overall testing is assessed, summarizing the results. This is followed by the projects post-mortem, which indicates the areas of the project that have seen successes, whilst noting the stages that could have used different approaches or strategies.

# Introduction

With the forever growing popularity of computer based games, it is becoming essential to create unique, engaging and powerful artificial intelligence (AI) systems.

The aim of this project was to create a turn based strategy game that can be played against the computer in the form of AI.

This report presents the research, design and production of the turn based strategy game, with a high focus on the artificial intelligence, the main aim being a smart AI system that is able to assess and formulate a response to a range of different situations.

# Background and objectives

The initial plan for this project was a multiplayer real time strategy game, where two players would compete against one another. The players carefully select from a set of predefined characters, position them in any strategical arrangement, then taking turns to move the chosen characters around a fixed map or environment to out manoeuvre their opposition, with an end goal of defeating the opposition in tactical combat.

This initial project idea was carefully broken down into more detail in the form of a project initiation document (PID) and is available for reference in the appendices. After further discussions with the provided supervisor it soon became clear that this was a suitable project to undergo, however it lacked slightly in terms of a challenge. This new level of challenge was expressed to give the project something more exciting than the initial plan and would be a welcome adjustment to the proposed project.

After further discussion with the supervisor it was suggested that instead of making a game that focused on a player vs player environment that it shifted towards some form of artificial intelligence (AI). Creating a game that was playable verse the computer would require more complex coding methods and research into a number of different approaches to solve this new problem. Using AI in the project would introduce a new area of development to the author.

As this was the only additional suggestion made towards the proposed project as well as the PID being approved in its original state, it was clearly a sensible project to undergo with enough reach for a prosperous game.

After assessing the project objectives within the PID and comparing them to the Specific, Measurable, Achievable, Relevant, Time-bound (SMART) principle there was room for improvement. Although the vast majority of the objectives were specific, giving clear, precise goals and outcomes, it was believed that some of them (Objective 4, 5) could have been broken down further to give a greater understanding of the original objectives.

In terms of being measurable, all bar the last objective had been formulated to give a clear indication of being completed or not. As for the last objective it was more of a personal objective. Although the objective clearly states that it will be measured by the outcome of a fully functional game. The completion of said game could be very basic and not have challenged the developers programming ability, because of this its focus was altered to make it more applicable, emphasizing the need to create a fully functional game that introduces new and challenging elements to the developer.

Creating achievable objectives was very important as it would give a sense of accomplishment throughout the research and development of the project. The moral boost that comes with completing a task is very important in the development process. The entire project was broken down into incremental steps (available Trello link) and explained in more detail further in the report. Each one of the objectives has resources available to show that they were indeed achievable, some of which are resembled within the final project and more are available in the appendices.

Relevance is an area ~~in~~ which could have been improved. Although the objectives are relevant in terms of the entire research and development process, they should have been more specific when writing objectives five and six. The objectives hold a heavy influence on the research part of the project. However point three is not as focused as it should be. This point targets the environment chosen to use when developing the project opposed to focusing on the project itself. It was a very important aspect of the entire project but does not express an overall objective for the final project.

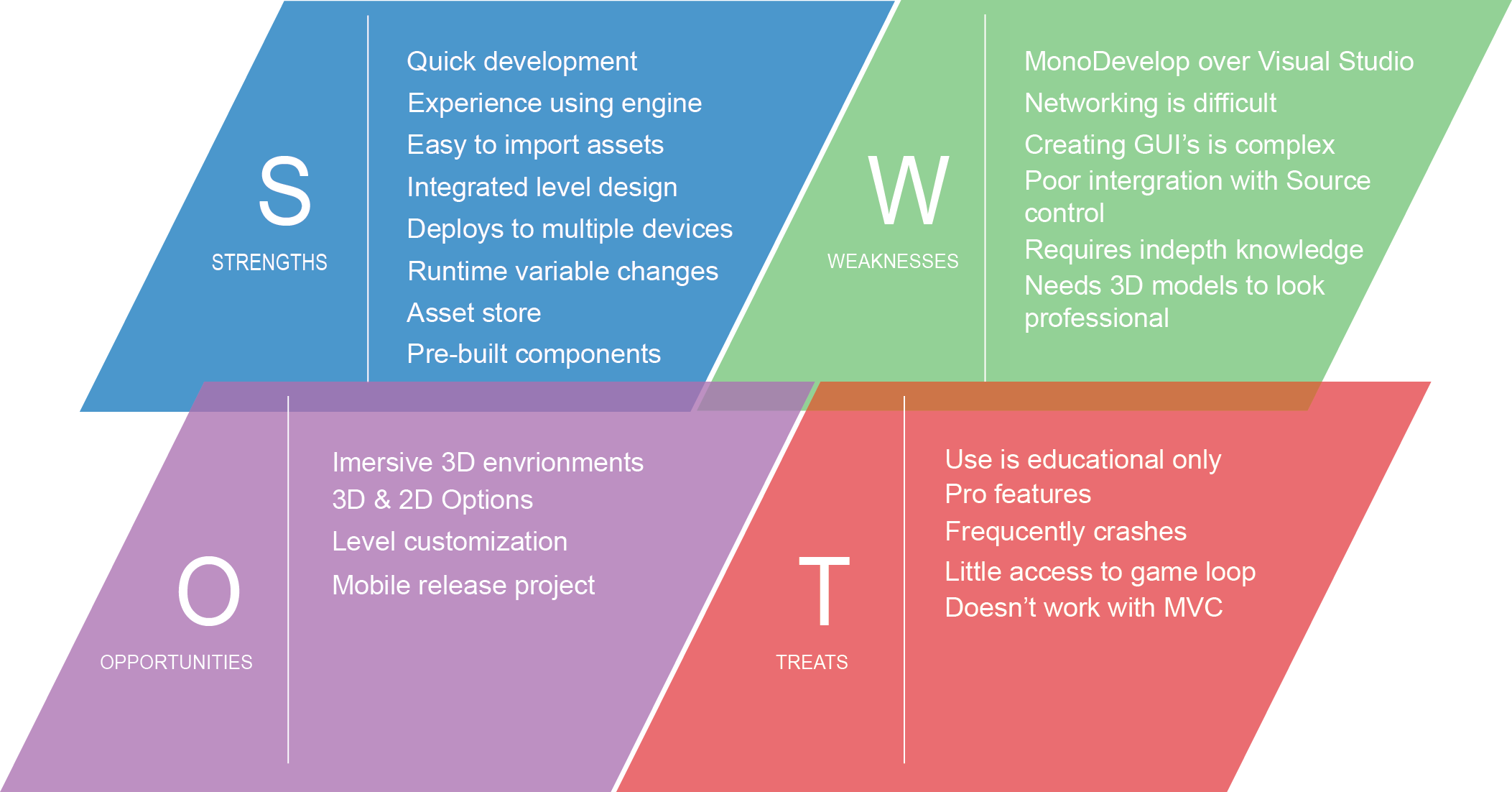
Time-bound was an interesting element to consider as the overall project was broken down into a project plan which specifies the goals and time frames in which they were given. Although the project plan didn’t cover all of the objectives it did break it down into more reasonable tasks to be achieved throughout the research and development stages. Each one of the objectives had been accounted for within the plan and progress was documented through weekly highlight reports. These reports were a healthy reminder of the initial project plan and how each weeks work had fit in with the corresponding time frame.

# Approach

As the project and its initial ideas began to unfold it was clearly a sensible plan with enough scope for expansion if necessary. As stated in the background and objectives there were little changes to the original project. These changes were made to make the game more challenging to design, develop and test. Although the changes were not set in stone it was important to evaluate them as this would ensure a better overall project.

## 3.1 Unity

One of the first hurdles that needed to be overcome before the development started was to assess the different technologies available and how each one was going to benefit the development stages. To give a better understanding of the options, a research paper was completed in which several game engines were assessed to demonstrate their strengths, weaknesses, opportunities and treats (SWOT), refer to the game engine research in the appendices for more information. Not only did the research paper bring insight into which game engine was going to help develop a top quality project, it enlightened the areas of the project that could have proved to be more difficult than first anticipated. The end result of the paper highlighted Unity as the chosen game engine. Some of the deciding factors were the experience the developer had with the environment, the rapid development made possible with the built-in level design and the access to its own asset store. These are only some of the factors that played a part in the decision but deemed the most important. Knowing that the project needed to be developed swiftly, there was a massive need for external assets, mainly 3D models.



Using Unity allowed the developer to play to their strengths as a programmer. He was able to use C#, which is the programming language he is most comfortable coding in. There was less emphasis and pressure on level design as Unity has its own built-in level editor. Unity can be debugged extremely easily, giving the option to edit variables during runtime. Unity makes it effortless to deploy to multiple platforms, this means that if the direction of the project changed and moved towards another platform, it would require little change if any at all.

## 3.2 Visual Studio & Resharper

One of the main concerns with using Unity was the integrated development environment (IDE) Mono Develop. Although this may not strike many people as a real issue, it was believed that programming on a poor IDE could lead to sloppy coding. Luckily after only a small amount of research it became clear that Unity could be integrated with Microsoft Visual Studio. The Visual Studio plugin offers full syntactic support along with simple debugging options, everything needed to get the ball rolling. Another bonus of using Visual Studio was its additional plugin Resharper. Resharper is like a second pair of eyes for your code, it will ensure you code in a uniform format, keeping the variables consistent throughout the project and offering alternatives to certain coding approaches. For example using a LINQ statement instead of iterating through a list of items in a for loop.

These are very small enhancements for such a large project but the payoff can be immense when passing on a code base to another programmer or simply ensuring the code is clean.

## 3.3 Game Design Document

After deciding on a game engine to use, a game design document (GDD) was created and made available using the reports appendices. The aim of a GDD is to be a more in-depth, descriptive document, outlining the story, characters, level, gameplay, art, sound, music, user interface, game controls (*Game design document*, 2016). All of these sections are vital when developing a successful game and each area was carefully assessed before starting the development of the project. Traditionally as the development of a game grows so does the size, structure and information stored in the GDD, however creating a skeleton for this document was important before the development started. It added direction towards all of the important areas required. The GDD is a very visual document expressing ideas with a variety of different media, whether that be links to videos or music, pictures from existing games, concept art scanned in from a scrap book, they all play a key role in the document. With this mixture of resources it was easier to visualise the intended game.

When first deciding to create a GDD, the plan was to maintain it using a blog. This would allow the developer to reach out to an audience, granting them feedback on development and providing external advice to a very remote project. The decision went against this method as a thriving GDD seemed to be a better approach and resource.

After the creation of the GDD several different areas needed to be explored before starting any of the programming.

## 3.4 Illustrator

First consideration was which resource was going to be used to create any 2D and or 3D assets. At this point it was obvious both were needed but knowing whether or not there was a need to create them or find external resources was important.

2D illustrations were needed to act as the user interface (UI), without custom made UI the game ran the risk of not looking very professional. There was less emphasis on finding a piece of software to create the 2D illustrations therefore, the research did not go into the same depth as the game engine comparison. However the more popular software options were used to test which one was going to be most useful. The software tested consisted of Illustrator, Photoshop and Inkscape. Each one was used to created basic assets, the comparison and thoughts can be found in the appendices under illustration testing. The key elements that were trialled were the variety of options available and discovering which would be the quickest, easiest and best overall. It soon became apparent that using Illustrator was going to be the quickest way of creating simple assets and was the chosen application.

## 3.5 Blender

Continuing from the need for assets brought the research to 3D modelling. 3D modelling is extremely complex and takes years to master, however the need for creating models with beautiful texturing was not one of concern. It would be ideal for a full release as it increases the engagement but this was not the case. Knowing very little about 3D modelling made it important to find a tool that was going to be easy, yet efficient to use. This is where Blender proved to outshine. Free to use, with thousands of tutorials online to help with the basic requirements, it was the software used throughout the designing and modelling phase.

## 3.6 Competition Comparison

Next a competitor analysis was completed. It was beneficial to go through some existing turn based strategy games to gain a little inspiration. This research consisted of scoping out potential competition, whilst giving a better understanding of the initial project objectives. See the competitor analysis located in the appendices for more information. It was important to loosely compare the project idea against other fully developed games as they will have been completed by a full team opposed to one person. It was important to understand that the game being produced would not be finished to the same level and or enhancement, as that would require a team but knowing the future potential and building for expansion was interesting.

## 3.7 Platform

Upon deciding on all of the above, the need to focus the game towards a specific platform was still undecided. Due to selecting Unity as the desired game engine this opened up the options to target the project to several different platforms. Unity can easily build the application into Android, IOS, console (Xbox, PlayStation), web or standalone PC. Although it would be ideal to have the game available for each and every option it needed to be narrowed down.

### 3.7.1 Mobile IOS & Android

There has been a massive spike in popularity for mobile based applications and games. These are normally tailored towards people on the go, being intuitive to use, simple to maintain and engaging. Both IOS and Android are worthy candidates, however they might not produce the system power required to run the game. Knowing that Android alone has thousands of different devices all varying in size and specification was enough to sway the decision against the mobile platform.

### 3.7.2 Consoles Xbox & PlayStation

Next were the consoles. Both Xbox and PlayStation offer a lot more power than a mobile device and would comfortably be able to run any game developed through Unity. Despite their extended power the need to develop a game in such a way that is passes the criteria required by either Microsoft and or Sony proved problematic. Not only would it be mandatory to abide by their strict rules, the game would need to be finished to a professional standard and the time required to do so was not available.

### 3.7.3 Web & PC

Targeting the game towards the web would not have been a bad idea. It can be hosted on game sites, have a large audience ready and waiting and it would allow for instant exposure. However the need to play the game online is a large obstruction. Ideally the game will be accessible offline as it maximises the opportunity to be played. This brought around the final choice of a stand along PC game. Offering the options of online and offline play, being run on a powerful machine and with no need to meet the expectations of an external source. Building a standalone PC game was seen to be the best option for the proposed game.

## 3.8 Bayesian Probability

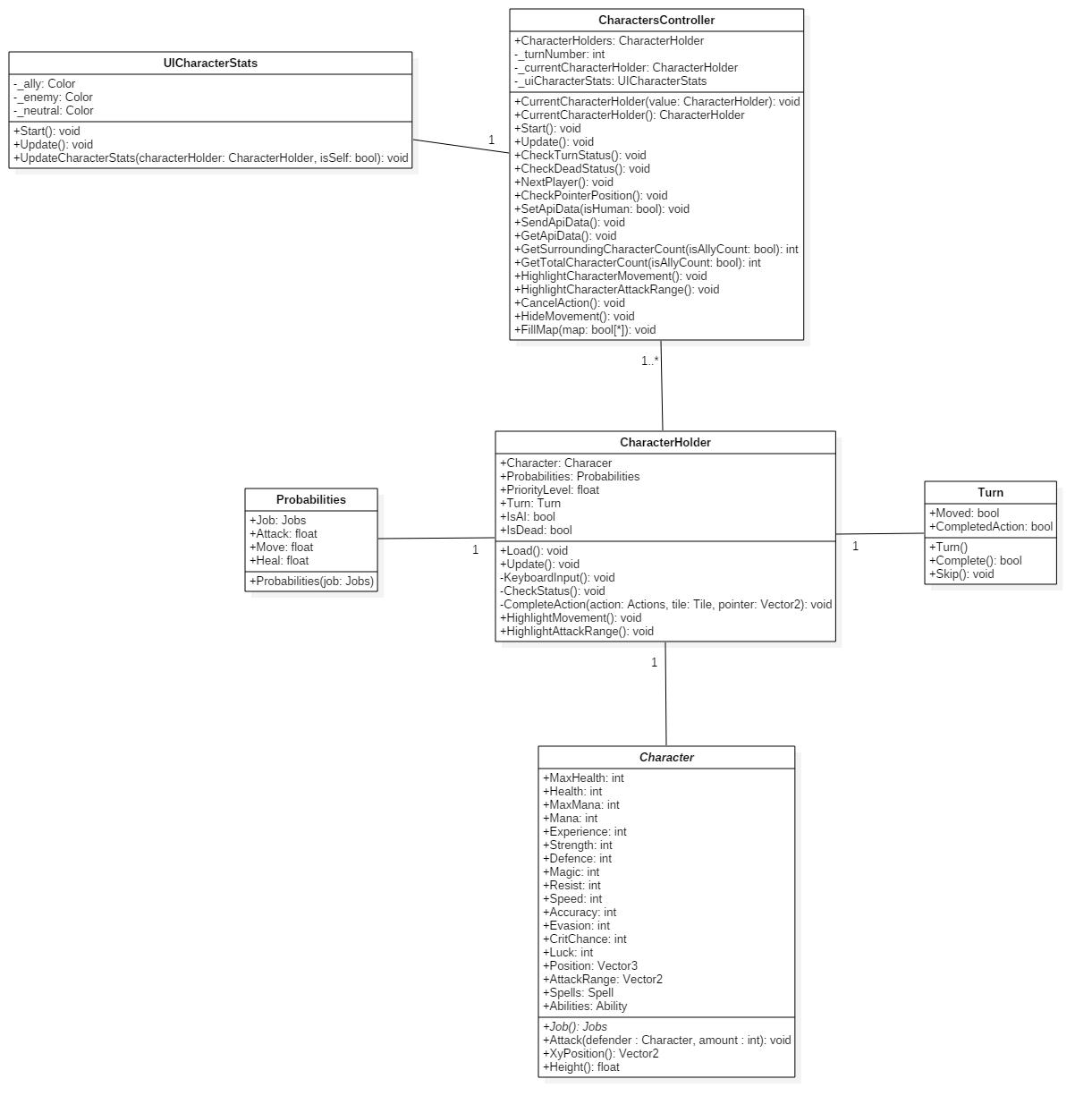
Lastly was the new focus of the project, the AI. This was going to be a very important part of the initial research as the developer had not completed any AI based programming before. It was key to understanding how different AI techniques could be used to really bring out the true potential of the game. After being provided with a little direction into which AI techniques could be appropriate, a research paper was completed outlining the potential AI techniques, refer to the AI research in the appendices. Knowing which AI system or systems were going to best benefit the games overall performance was vital. The end result showed an adaptation of the Bayesian probability technique, in which the AI will assess its given situation, calculate the probability of a series of different outcomes and execute an appropriate response. A paper better outlining the chosen technique is labelled Bayesian research in the appendices.

## 3.9 Architecture

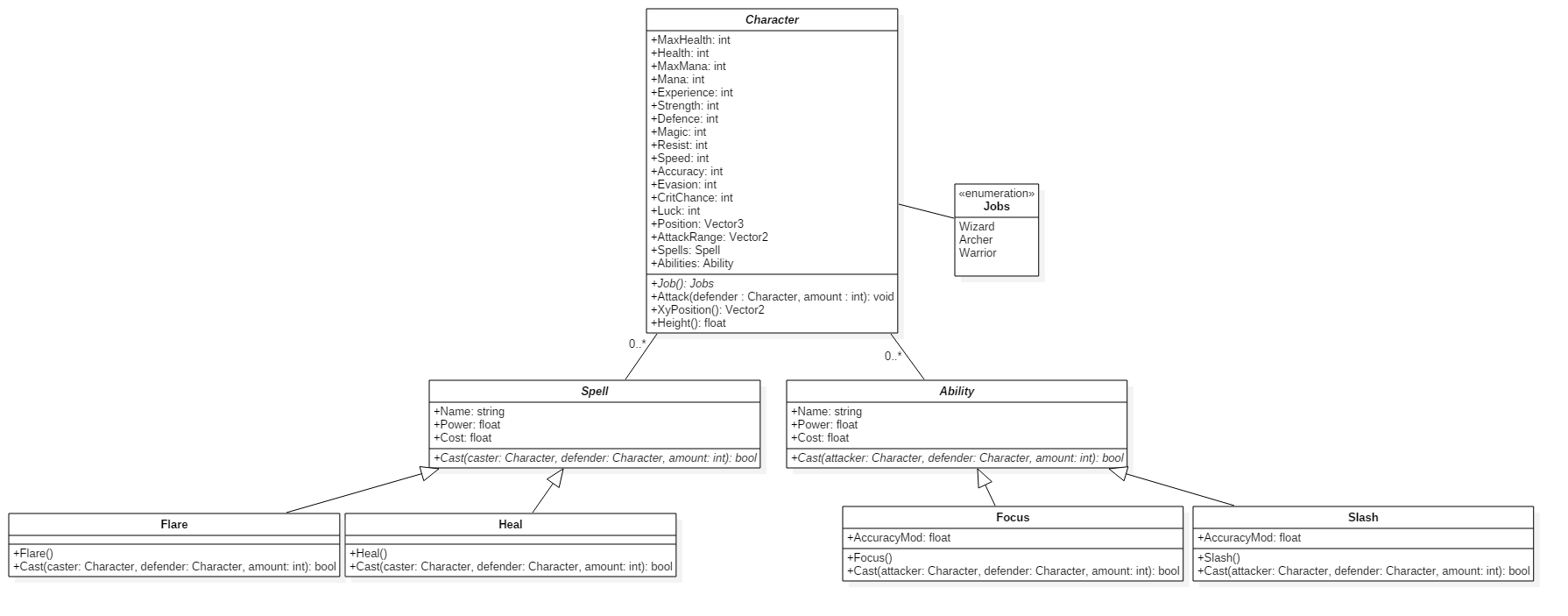
### 3.9.1 Characters

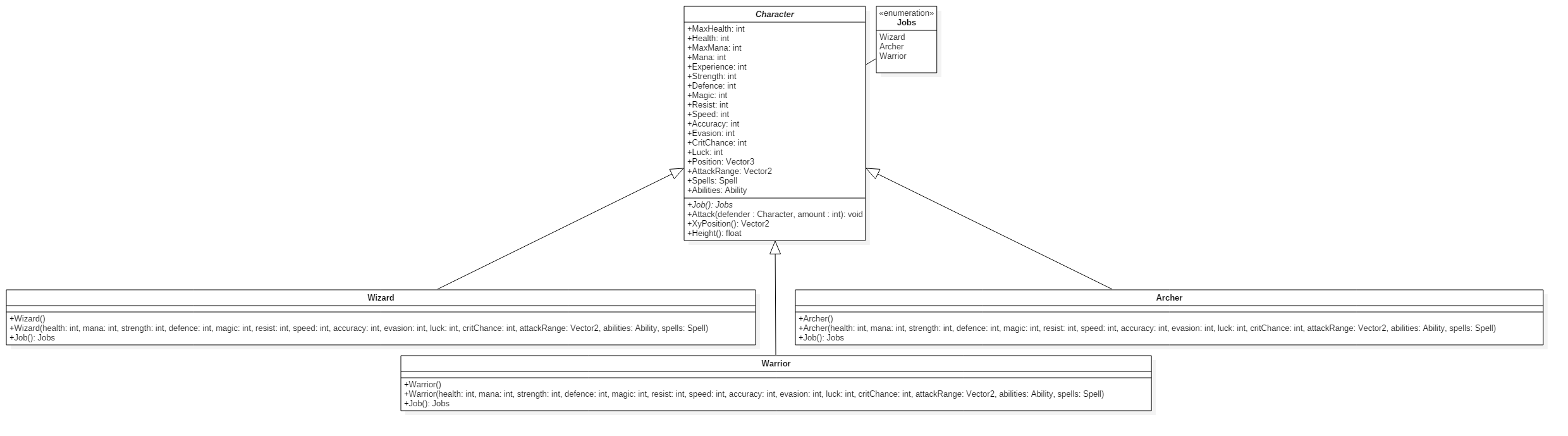
The unified modelling language (UML) diagrams used to design the project are extremely difficult to properly visualise in their complete state due to the lack of space on paper. Because of this they have been broken down into smaller UML diagrams to better display their features. All diagrams shown in the report have enlarged replicas in the appendices under section UML diagrams.

The first UML is the central hub for all the characters, based around the characters controller. The characters controller loads all of the characters and tracks them within a list. The controller switches the characters at the end of their turn, removes them when they are dead, gets and sets the API data, along with a range of other functionality. The characters controller holds a list of character holders. The character holders are used to attach to the objects within the Unity editor as every element used in the Unity editor must extend from a Mono Behaviour. Wanting to keep the pure character model free from the game it has been wrapped in a holder. This allows for extended functionality without overloading the character class. The character holder contains the probability model used by the AI system as well as a Turn class which maintains the actions completed by every character.



Next is the abstract character class. This stores all of the different stats for any given character as well as the Spell and Ability lists. These lists are also abstract and allow for a range of different classes to fill them. Below is an example of some of the intended spells and abilities for the game. The project contains many more but for the sake of the diagram it is clearer using a smaller pool of abstract classes.



This is the breakdown of the different job options a character can have. Throughout the report these jobs are referred to as ‘Classes’ but it would become confusing to use that term here. Each job specifies the different stats given to them. These are what define their strengths and weaknesses, and can easily be adapted with the multiple constructors.

# Project Management

## 4.1 Trello

During the entire research and development stages the same project management tools were used. This mainly comprised of using the online software Trello. Trello is a simplistic management tool that enables the users to organise their workflow with the use of cards. Trello was used to maintain a record of all the different research and development milestones. Each milestone was broken down into maintainable deliverables and moved along the pipeline as it progressed.

## 4.2 Kanban

Kanban is an effective way of managing ones work by visually displaying all of the current work being completed and where it is positioned in the development process. Trello partnered very well with the Kanban method as cards can be manipulated around the screen very easily and quickly depict how well everything is going. As this project was being taken on by just one developer, it might not have been necessary to maintain a Kanban workflow but it proved to be beneficial in the long run. The Kanban board used, only contained a few different columns as the development process was in need for a quick turnaround on elements. These columns consisted of a backlog or to-do list, a developing stage, testing and complete. The backlog contained all the work needed for a minimal viable product (MVP), at any time a card from the backlog could be brought into the developing column as long as the total number of cards currently in development didn’t exceed four. This is a very important aspect of the workflow, as it makes sure that not too much is being worked on at one given time, and that each card being completed is receiving the required attention. Once a card is seen to be finished it would pass on to the testing column. Here it could sit for some time but much like the developing column it had a limit. The limit for testing was a slightly larger sitting at eight different cards. This brought the focus of the Kanban board to the development but didn’t allow it to get clogged up in testing. The final column complete, consisted of every single card that passed through all the previous stages. This worked as a reference to the work completed and nothing more.

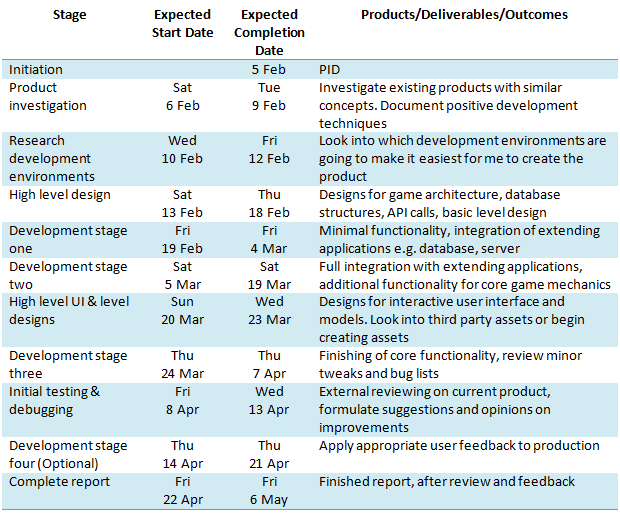
## 4.3 Source control

To manage the code base, assets and documentation Git was used. Whenever a card or several cards moved across the Kanban board the work would be committed across to the Git repository. This served as a backup to all the work and enabled the project to be cloned onto any computer making it effortless to work anywhere and everywhere. Git also allowed for a review of progression, looking through all the commits and what they included showed how much work was done at each stage. After reviewing the process used for Git it could have benefited from some improvement. A system to tag each Trello card and commit would have been extremely useful as a back reference. Also following the Git flow method would have been useful but not deemed as a priority, as the project was maintained by an individual without the need to keep track of release, patches and or testing branches. If the project was to be moved into a team environment this would be put into place to ensure the Git history, branches and releases are all maintained correctly.

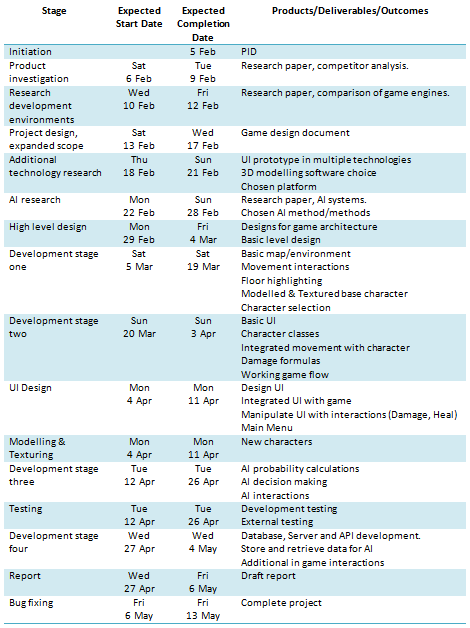
## 4.4 Project Plan

Taken directly from the PID the initial project plan clearly states the different levels of research and development, providing each one with a time frame and outcome. The project plan was very vague in terms of deliverables, proved difficult to follow, but clearly showed the different stages which needed to be completed.

Even though the project was kept to a strict management plan, with several techniques in place to ensure progression, there were expected changes when comparing the initial plan to the final outcomes and timeframes.

4.4.1 Initial Project Plan

### 4.4.2 Final Project Plan



After comparing the two tables it is obvious that the initial plan was far too vague but this can only be expected seeing as it was completed before the project started. The final plan shows a higher level of detail specifying the direct outcomes for each stage. Some of the stages were able to be run at the same time due to the project management being extremely strict and allowing for constant progression. A vast amount of the time given to the project was taken up in research, although this might have stunted the development, it played such an important role within the overall project that the time given was completely justified. Without the extended research time, elements within the project could have suffered. The times and dates given for the final plan are a rough estimate but a clear representation of the time allocated to each stage.

# Outcome

## 5.1 Objective completion

Direct snippets have been taken from the PID’s objectives, and analysed to briefly show their completion.



During the project approach phase, one of the biggest benefits was analysing the potential competition. Not only did it give a larger insight into how successful turn based strategy (TBS) games have been developed and marketed, it acted as inspiration.

This research showed that the level of detail needed to immerse the user into a strong storyline was extremely difficult to achieve. TBS games are traditionally very slow paced yet keep the user involved, this slightly deters users from a storyline as they want to return to the interactive gameplay opposed to listening through a series of dialog. Due to this reason it is thought that this is why many TBS games remove themselves from in-depth stories.

This resulted in not adding a storyline to the final project. Instead the focus was more towards immersing the players into the core gameplay. The game needed to have a simple to follow environment, smooth flowing game mechanics and intuitive user interface to avoid obstructing the player with too much information. This allows the player to concentrate more on the game and out manoeuvring their opposition.

Assessing the weaker areas of the project was crucial to ensure that the end result was going to be the best it could be. The developer noted their weaker points and the areas in which they felt could jeopardise the final result. However this was carefully evaluated prior to development and smart solutions arose.

When the comparison of competition was completed the worry of needing top grade 3D models was extinguished. A lot of indie TBS games focus more on the intuitive user interface (UI) rather that the models within the game. A result of this was the need for appealing UI but this was more achievable that modelling and texturing quality assets.

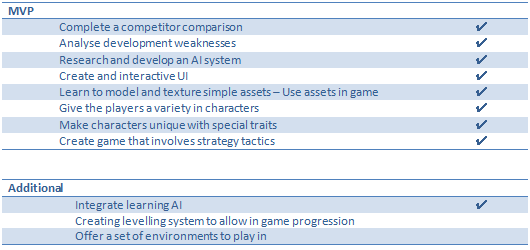


Mentioned within the approach several game engines, illustration tools, platforms, AI techniques and 3D modelling tools were assessed to understand their strengths and weaknesses. Several documents are available in the appendices that further explain all the outcomes.



When the PID was constructed the initial scope listed several objectives all of which would be vital to make the game a success. This intended scope was then expanded upon within the GDD. These milestones or development objectives were used as a guide to create the final project and acted as a reminder to what was required for a minimal viable product (MVP)

The development objectives from the GDD are as follows.



The checkmarks resemble the completion of an objective.



This point is justified with the final project and the user testing. The user testing can be seen in more detail later on in the report. Although this objective was very broad it was still the most important, specifying the overall goal for the project.



The final objective was aimed to be more personal towards the developer. It might not seem very measureable and or specific but it was achieved. The developer had never programmed any form of AI system before this project. He was introduced to a series of issues like creating 3D assets, all of which were overcome. He completed a project to a high standard, following strict project management all of which has benefited his development and programming skills.

## 5.2 Game Design

### 5.2.1 Overview

The title “Final Adventure” was the result of this project, in which one player will compete against the created AI system. The goal of the game is to strategically move your chosen characters around a ten by ten map forcing your opponent into difficult to fight scenarios. The player will carefully select five characters of either different classes or the same and place them into starting positions. Each class type boasts advantages and disadvantages within the battlefield, so selecting a worthy combination is the first step to victory. As stated previously the player manoeuvres their characters around the map and is permitted to execute one action per turn. This may be a simple attack or a more complex ability or spell. Once a character’s health points (HP) have been reduced to zero they are removed from the game giving the advantage over to the opposition. The game will end once the player or the AI has run out of characters.

A lot like Chess the game focuses on a simple terrain, yet the understanding of each different character and their varying strengths in battle is what brings victory to a player.

### 5.2.2 Game Flow

Upon loading the game the player is presented with the main menu. There is not much in terms of options however they are able to continue towards the game or they are able to familiarise themselves with the controls and aims of the game before starting.

Once the game has begun the player will be navigated towards the character select scene. This is where the player is able to explore the different types of character classes ranging from Wizard to Warrior. This scene displays the given stats for each class e.g. strength, defence, health and more, as well as their available Abilities and or Spells. Understanding how these stats effect the gameplay are important, however it is also something that the player will need to experience in the game to fully understand. The player is also able to read an overview of spells and abilities available to give them more of an insight into the class choices.

Once the player has selected their five characters and placed them within the designated positions, the game will begin. Starting with the player they will be given the option to move each one of their characters in turn as well as the option to complete an action. Once a characters has moved and completed an action their turn is over, however if they choose not to move or not to complete an action they are able to wait. This will skip their turn and move on to the next character. The back and forth moves between the player and AI will continue until a victor has emerged. The game will then return to the main menu allowing for another go.

### 5.2.3 Core mechanics

#### 5.2.3.1 Selecting a team of characters

This is the first interaction for the player, increasing the need for it to be totally immersive. This scene includes a lot of animating between actions e.g. spinning the carousel of character types and moving the characters into position. These subtle actions are enough to draw in the players attention getting them hooked as soon as possible. This core mechanic allows the user to explore the characters for the first time, granting them additional knowledge of the overall look and feel. Using the left and right arrow keys the user rotates the carousel displaying either the next or previous character. Each character in the carousel can be rotated on its Z axis with the D key rotating them right and the A key rotating them left. This simple feature allows the user to better familiarise themselves with the character models to avoid confusion once the game has started. Finally the player is able to use the up and down arrow keys to move through the menu. The menu does not offer much but it does allow for the current character to be selected and placed into position and displays additional information on the current character. This additional information better explains their strengths, weaknesses plus their abilities and spells.

#### 5.2.3.2 Moving the pointer

Once the game begins the player is able to start their first move. As long as the player is in control they are able to explore the map and investigate the random opposition. To do this the player can use the in-game pointer. Resembled as a purple tinted square, this indicates where the current game focus is. Moving the pointer around the map will show both ally and enemy information e.g. Health, Mana, Class. The pointer is controlled using the W, A, S and D keys, moving it up, left, down and right. When the pointer is manipulated around the map the current active character will rotate to face its direction. This was a simple, yet effective minor mechanic that helps bring the game alive. The pointer is used to aid the selection of movement and character selection when completing an action.

#### 5.2.3.3 Using the UI

The UI has been created so it is extremely simplistic yet intuitive to use. The majority of the UI throughout the game uses the same scrolling menu on the side of the screen. The menus are created dynamically from a list of options. This has been done so it can easily be altered for each character as they all have different options. The up and down arrows are used to cycle through the menu options while the return key (carriage return) is used to select an option within the menu. The menus only ever show a maximum of four items at one given time, so by cycling down the items the menu scrolls down to reveal the entirety of the list. As the player is only able to complete one movement and one action per turn, each item in the menu can be disabled to visually show that the action has been completed. The currently selected item is also highlighted in a light blue and is animated to stretch out beyond the alternate options.

These menus use a method of main and sub menus to traverse through the cascading options e.g. selecting the ‘Spells’ menu item will hide the main menu, and reveal the sub menu containing the available spells. When in a sub menu or while finishing an action the player can press the right shift key to return to the previous menu.

#### 5.2.3.4 Moving characters

To move the character during their turn the player must first select ‘Move’ from the menu. Once selected the floor will highlight with the available tiles to move to, these tiles are highlighted in a red tint. Each character has a speed stat which determines how many tiles they are able to travel each turn. The player may decide to move the character to any one of the surrounding tiles assuming that it is within their speed limit and it is not taken by another character. The user is able to decide which tile they wish to move to using the game pointer mentioned previously.

When an appropriate tile is selected using the enter key (carriage return), the game will use the A star path finding algorithm to find the best route to the chosen destination. This algorithm returns a list of coordinates that navigate from start to finish. The game then moves the character from tile to tile whilst rotating them to face the correct direction.

#### 5.2.3.5 Completing an action

Much like moving the character, to complete an action the player must use the menu and sub menus to narrow down the options to just one. The actions have been broken down into more defined sections explaining their purpose in a little more detail. However as an overview, the actions are used to bring the game state in favour of the characters team. Generally this is in terms of an offensive action, tailored to bringing down the HP of the opposition.

**Attack**

Every class has access to the ‘Attack’ action as it is the fundamental element of damaging the opposition. An attack has no prerequisites bar that the character has more than one health point. Even though every class can attack an opposition they all have varying attack ranges. The attack range has been used to bring another level of diversity towards the different classes. Traditionally the attack range is one, meaning the characters have to be next to each other. However an example of a more complex attacking action is the Archer. As the Archer’s role includes using a bow and arrow, they are not able to attack characters within a close proximity, but are able to attack further than any other class. Knowing that every class type can attack is not enough, as attacking is not always the best option, as it is based on the characters strength statistic, meaning some classes are better at attacking than others.

**Spells & Abilities**

Spells and Abilities are more tailored towards the given class and not necessarily always offensive. Most characters have access to the ‘Heal Spell’ which simply heals a targeted character but the heal amount is based on the classes magic statistic. This brings another level of strategy to picking the perfect team composition. It means that a strong team doesn’t rely on a set of offensive character choices, but requires a combination of offensive and defensive. There are a range of other damaging spells available for use and have elegant partial effects to really bring them to life.

‘Abilities’ are similar to Spells however they are aimed towards manipulating statistics and damaging the foe. E.g. boosting accuracy to ensure the attack hits its target, or charging up power to unload a devastating attack but with reduced chance of success than a normal attack.

Both Spells and Abilities consume Mana when cast, meaning they are to be used situationally opposed to every turn. This again brings more strategic thinking towards each and every turn, getting the user to think about every action and how it is going to affect them in future turns.

**Wait**

Waiting, is the last option the player has for each characters turn. This will end their turn and move on to the next character. Waiting is used when a character wishes to attack an enemy yet stay in the same position. Or run away from the danger and not complete an action.

#### 5.2.3.6 Statistical formulas (Damage, Abilities, Hit/Miss, Critical hit)

Every action available in the game except waiting needed a tailor made formula to calculate the changes in values. This could be calculating the damage of an attack or the amount of health received from healing. A lot of these formulas are similar in structure but have been manipulated to produce the required outcomes. Here is the formula used to work out the physical damage produced for a normal attack.

D = ((SV / 250) \* (SV / DV) \* B + 2) \* Mod

D = Damage

SV = Attackers Strength Value

DV = Defenders Defence Value

B = Base Damage Value

Mod = Additional Modifiers

This formula was adapted from the damage formula used in the popular game Pokemon (Bulbapedia, 2016). They are very similar in structure with minor changes tailored towards the project’s needs. The Spell and Ability formulas have been altered a lot more however, but this gives a good representation of how they were constructed.

This formula allows the damage to scale on several different aspects. The damage is scaled based on the class’s strength statistic. Traditionally the Warrior type class has boosted strength stats as they are the head of a fight, opposed to a Wizard who should keep their distance. Secondly is the base Damage value. In a normal attack action, the characters strength statistic is used, however this is changed for certain abilities like the Focus or Slash ability. Lastly is the Modifier Value, which mainly consists of the critical Hit chance. This is where the characters critical Hit value is used against a random number between zero and one hundred to determine if the hit should deliver twice the intended damage. The Modifier Value has been created to introduce weakness and advantage attacks, however this has not been included in the final product as there are not enough character classes available to create a healthy balance.

When a physical attack is being carried out the game also takes into consideration the chance of successfully hitting. The formula used to calculate each hit is not very complex when compared to working out damage values.

HC = HR \* (AC / EV)

HC = Hit Chance

HR = Hit Rate

AC = Attackers Accuracy Value

EV = Defenders Evasion Value

Hit rate is a Class specific value, based on the class attack range. This means that Archers have a slightly smaller hit rate as they are able to attack from several tiles away, whereas a Warrior has a higher hit rate because their attack range is very small. This formula will result in a value relatively close or marginally above one. Any result over one is an automatic hit, where any hit chance under one will be compared to a random number between zero and one hundred. If the random number is higher than the hit chance multiplied by one hundred the attack will miss, otherwise it will hit.

All Spells are directly cast upon the chosen recipient therefore there is no need to calculate a hit chance for them.

### 5.2.4 Bayesian Probability Adaptation

#### 5.2.4.1 Summary

Bayesian probability is used to produce statistical probabilities for an unknown event based on information available either after or prior to the action. An adaptation of this method has been used to create the probability values for each human based character in terms of the chance to attack, move and heal. These are the three main options a player has when completing a turn so are the bread and butter for all the probability calculations.

#### 5.2.4.2 Calculating probabilities

At the start of every AI controlled characters turn, the game will run the current situation through the probability calculator. Every AI character will assess every human controlled character comparing stats like surrounding ally and opposition count, current health percentages and other variables. All of these and more have been used to carefully produce a set of values deemed viable as accurate predictions.

Ensuring that the probabilities for the three options attack, move and heal are calculated to benefit the AI’s decision making is crucial. Without this the AI might seem to make rash or in other terms stupid decisions going against the goal of an intelligent AI system.

There is no right way of correctly calculating the probability values but the formulas used have gone through several iterations to help make them more precise and meaningful.

**Attack Probability**

AP = ((ANT / ONT) \* ((TPH / OTH) \* (SV / DV))) \* Mod

AP = Attack Probability

ANT = Allies Near Target

OPT = Opposition Near Target

TPH = Target Percent Health

OTH = Opposition Percent Health

SV = Strength Value

DV = Defence Value

Mod = Additional Modifiers

This formula was created whilst playing other TBS games, by carefully analysing the different components in a player’s decision. Some of the reasoning behind a players choice may not always be cognitively assessed, instead it may be just a habit from playing the game or similar TBS titles. This habit or instinct is comprised of so many different variables it is extremely difficult to focus them down into one formula. The above attack probability formula uses a variety of elements seen to be the most impactful in a player’s decision.

To breakdown the implemented attack probability falls into four main sections. Firstly is the character’s surrounding the target. Normally in warfare there is a great power in numbers and generally seen as a reason to attack or retreat. The number of allies is compared to the number of opposition characters but only if they are in movement and attacking the target. Any character out of range from the target is disregarded as they will not be able to directly impact the target in the given game situation.

Secondly is the remaining health of the target character verse the active AI character as a percent. These are compared to adjust the aggressiveness of the characters decision. Typically if a character is low on health they will become less aggressive and more defensive, this is also amplified when the targeting character has a greater health percentage.

Thirdly are the targets strength and the AI’s defence stats. These resemble the chances of the attack being an impactful action, swaying the decision to attack against a strong foe but enhancing it if they are weak.

Lastly is the modifier at the end of the formula, much like the attack damage calculation this acts as a final buffer towards the value. In the attack probability it is very simple, either increasing or decreasing the overall probability based on the targets Class type. Assuming classes such as a Warrior are going to be used more aggressively than a Wizard.

**Heal Probability**

HP = (DT (DT\*JM, …, …)) \* (OTH / ATH)

HP = Heal Probability

DT = Damage Taken

JM = Job Modifier

OTH = Opposition Total Health

ATH = Ally Total Health

The Heal probability is certainly the most interesting formula to be created for the game. It has a reoccurring calculation for each and every ally surrounding the current target that exponentially increases the chance of the heal action given the targets health lost as a percent and their class type.

To break this down a little further each human controlled character assesses their surrounding allies. The assessment is manipulated to increase the heal chance if one of the following situations is correct. An ally surrounding the target has very low health, the majority of the surrounding allies have lost a portion of their maximum health or the target assessing their allies are of class type that would deliver the greatest heal values e.g. a Wizard.

Even after this calculation there is the chance that the heal probability could be altered. The sum of all the surrounding AI characters health is compared to the sum of the surrounding human controlled characters and used to amplify the probability. This ensures that the heal action being completed couldn’t be used more effectively as an attack due to the health differences or vice versa.

**Movement** **Probability**

Probably the least important of the three options, but still an important element to take into account is the Movement probability. Calculating the movement probability of a character is by far the most difficult to achieve as it can be done either offensively, defensively, tactically or sporadically.

Due to the sheer amount of different possible uses of a movement it was not feasible to break it down into a formula. Instead it is based on the character’s current health points. The lower the health points the greater the chance of that character moving. This was done because the most important time a character might move is when they are retreating from battle and this is normally resembled by their health points.

#### 5.2.4.3 Assessing probabilities

At the end of the probability calculating each and every human controlled character will have a value ranging from zero to one for each of one of the three assessed options. Each one of the options is used to finalize the actions completed by the AI. The vast majority of the time, the actions completed by the AI are rather predictable but this is a great sign. This shows the AI system is resulting in actions a human player would cognitively decide to do, oppose to something seen as random.

#### 5.2.4.3 Decision making

Ensuring the AI make intuitive decisions was one of the biggest back and fourths in terms of development. Even though the final outcome is actually rather simple it was important to test a variety of different methods to truly understand the underlining elements needed. The decision is broken down into two stages.

The first stage is finding a target on the opposite team. To do this the game goes through all the human controlled characters that are in attack range and applies weighted modifiers to the probabilities created in the previous step. The three weighted modifiers hold different values but starting with the most impactful is the amount of health missing from the target character, next is the chance that the character is going to heal another member of their team, lastly is their chance of attacking a fellow AI character. These weightings have been associated with the given attributes because they are seen to be the most impactful when deciding which character to target.

To give a better understanding of the approach the following has been produced.

* Targets missing health produces a value between zero and three
* Chance of the target healing another, produces a value between zero and two
* Chance of the target attacking another, unity produces a value between zero and one

The sum of the three values is assigned to each assessed target as a priority level.

The target with the highest priority level will be selected as a target. However there might not be any targets in attack range to evaluate in the first place. If this is true all of human controlled targets out of attack range are then assessed with the same criteria. It is important to assess the targets in attack range first as they are the only characters that can directly impact the AI unit.

This first step will always produce a target, even if their priority level is extremely low. The second step is for the AI unit to evaluate it’s in-range allies for healing. Nowhere near as complex as the targeting system, the healing target finder compares all the ally units within movement and heal range, then orders them descending by the percentage of health remaining. If the target at the top of the list has a health percent lower than the provided bound the AI will overwrite any human controlled target with its ally and prepare for healing. The bound used for comparison is altered based on the AI’s Class type e.g. increasing if the current AI is of a Wizard class type. This way the AI units are going to act more specifically to their given role.

If the AI does not need to heal one of its fellow team members, it will resume attacking the opposition target. At this point the game will pick a random offensive Spell or Ability from the list of available options linked to that class type. Assuming the current AI character has enough Mana to cast or complete the action they will do so, otherwise they will fall back to the standard attack option.

#### 5.2.4.4 Storing data

Part of the implemented AI system includes a level of learning, this enables the AI to adapt over time making its decisions less programmatic and more assessment based. In order for the AI to begin its learning, there needs to be a constant source of data being fed to it. This data is comprised of several different variables but the end result is taken from the human players and what they decided to do at the end of each given turn.

To store the required data a web server, application program interface (API) and database were created. The focus is not towards the web production of the product but more details can be found in the appendices in section Database & API documentation.

The API has two simple calls available. The one in regards to storing data requires each one of the following fields to be filled.

* Number of surrounding allies
* Number of surrounding AI
* Total number of ally characters
* Total number of AI characters
* Class type
* Target health percent
* Target Mana percent
* Movement (True/False)
* Action completed

The variables are taken at the beginning of the characters turn, and sent off to the database at the end of their turn.

#### 5.2.4.5 Learning

One of the more important areas of the AI is the learning system in place. Although it isn’t the most intricate in design or uses, it proves to be a vital aspect of the overall behaviour. At the beginning of the AI’s turn it will assess the opposition and retrieve all the corresponding data to their current in-game scenario. This will return all of the data entries matching their current situation and is used to aid the decision making process. The returned entrees are first grouped by the action value, and then ordered descending by the count of each group type. This will place the most common completed action at the top of the list. The game takes into account that the information gained from the database might not always be useful. If only a few entrees are returned it is ignored completely. This stops the AI from making decisions based on too little background knowledge. When enough data is returned and organised it plays a small part in making an overall decision, if this wasn’t the case the game could be played and manipulated in such a way to jeopardise the AI.

## 5.3 Modelling and User Interface

The user interface (UI) plays a strong part in the final project. Navigating the players through their battle options and displaying character statuses. It was important to make the UI very simple yet effective at displaying its meaning. Using a simplistic UI allowed for the focus to remain on the game environment, letting the player focus on the tactical play style over the interactive UI. All of the UI and its features have been explained in more detail within the GDD.

The modelling for the game was initially seen as a weakness towards the project. It was one of the major hurdles that needed to be overcome in order for the game to truly excel. After creating the simplistic UI the theme moved towards the models and their texturing. It soon became apparent that the models for the game didn’t need to be intricate in design or to a professional standard. Due to this all the models were designed by the developer giving them the chance to broaden their game development skill set.

The game does fall short in terms of professional UI and models but this was never the focus of the project. It certainly will be something that sees further adjustments as the project continues to grow in the future. For additional insight into the UI, modelling and other design features refer to the GGD.

## 5.4 Database & API

The database and API used for the game were introduced late into the development stages and therefore did not receive the same attention as the rest of the project. Using these features was an additional extra in terms of the MVP and only completed as the rest of the features were done. The structure of both database and API are explained in greater detail in the GDD. The need for these extra elements were to allow the AI to learn from existing data, storing the data from human controlled characters and using that information to fuel the AI’s actions.

One of the more important parts of this section was the use of RestSharp (*Simple REST and HTTP client for .NET*, no date). This is a very simple to use library, incorporated to access the REST calls available on the API. As time was of the essence and this feature needed to be completed as soon as possible the use of a library was very welcome. It meant that the development was completed a lot faster and there was no need to worry about going through all the minor testing phases if created by the developer.

# Testing

As stated previously in the project management section of the report, each and every element of the development process was tested before being labelled as complete. This meant that everything being placed into the game was double checked and reduced the chances of any problems as development continued.

During the end stages of development the testing focus moved towards the game as a whole opposed to individual aspects. This was a vital part of the entire project as it brought up several issues that could have been missed. As part of the overall testing phase a user test was completed. The user test consisted of someone using the game as intended, getting their feedback on its strengths, weaknesses, and any issues in the game flow or additional pointers to help as the development came to a close. The user testing report can be found in the appendices along with the series of test cases carried out on the game, see section labelled testing. These test cases show each and every test carried out on the application to ensure that the proposed MVP objectives were met. The overall result of the testing phase was positive with the user pointing out some of the key features as the most intriguing.

# Project Post-Mortem

As an overview of the entire project the results have been extremely promising. This being said there are several areas within the research, developing and testing phases that could have had some fine tuning or alternate approaches.

All of the initial objectives have been targeted and achieved, however the overall scale of the project could have been improved. The original plan for the game was much larger, offering a variety of options in terms of Maps, additional Spells, Abilities and Characters. This is purely down to the amount of time consumed by researching, understanding and development of the AI. Luckily the focus of the project swayed towards the AI from the very early stages, so the personal expectations of the entire game should have been lowered but the amount learnt and gained from taking the time to properly develop an AI system has been invaluable. It offered a much larger challenge in terms of development, and expresses skills that would not have been used.

The approach towards the project was strict in terms of how it was executed. Based on past experiences, knowing how to properly maintain ones development time is of the highest priority. This is why such a high emphasis was placed upon the project management. Being able to manage a project efficiently leads to a more robust and detailed project.

One of the biggest regrets in regards to the development stage was the structure of the project. Unity is known for being a difficult game engine to structure code and this has been one of the downfalls of this project. Additional time should have been allocated to better understand how to structure a project using the given technology. This would have led to more extractable, adaptable and maintainable code. If the project was to be started a fresh this would without a doubt be the area of the project that saw greater attention.

It has been mentioned that each element created was thoroughly tested before passing through the Kanban board. The time taken to do so was perhaps detrimental towards the project. This time was beneficial and professional in practice, but given the proved time scale for the entire project it could have been used to enhance other aspects of the overall project.

Another area which could have improved the project is using third party libraries. Only a minute amount of work was produced using third party libraries, all of which worked perfectly and sped up development time. Utilizing this more could have led to a larger project but the learning behind building a project from scratch is more important that the sheer size of it.

# Conclusion

Creating an intelligent AI system that fully immersed the user in interactive game play was without a doubt a challenge. The project successfully allows a user to compete against an AI controlled opposition, testing their wits, intelligence and tactics in warfare.

The completed game and AI offers a wide range of possible uses within a series of different game types instead of just the intended TBS game type. This adaptability really proves that the chosen techniques are one hundred percent viable for a production level game.

The project allowed for expansion on existing skills and the introduction of many more, the final product is seen as a humongous personal success. Not only did the project meet the initial objectives, it has allowed for the professional production of a stimulating game that will see continued development in the future and a public release.

# References

* *Bulbapedia.* (2016) *Damage - Bulbapedia, the community-driven Pokémon encyclopedia*.[Online] Available at: http://bulbapedia.bulbagarden.net/wiki/Damage (Accessed: 6 March 2016).
* *Game design document* (2016) in *Wikipedia*. [Online] Available at: https://en.wikipedia.org/wiki/Game\_design\_document (Accessed: 6 May 2016).
* *Simple REST and HTTP client for .NET* (no date). RestSharp. [Online] Available at: http://restsharp.org/ (Accessed: 26 April 2016)

# Appendices

~~User Guide~~

~~PID~~

~~Game engine research~~

GDD

Illustration testing

AI research

Bayesian research

Competitor analysis

UML diagrams

Database & API documentation

Testing

User testing