|  |
| --- |
| Advanced Games Programming |
| CGP600 – AE1 |
| Game Design Report |

|  |
| --- |
| Alexander Pearson  Q13253760 |

Contents

[Introduction 2](#_Toc528918363)

[Game Design 2](#_Toc528918364)

[User Stories 2](#_Toc528918365)

[Software Design 3](#_Toc528918366)

[Testing Plans 4](#_Toc528918367)

[Work Breakdown Structure & Gantt Chart 4](#_Toc528918368)

[Reflection 4](#_Toc528918369)

[Bibliography 6](#_Toc528918370)

[Appendix 8](#_Toc528918371)

# Introduction

This report details the steps involved in the design of a new game. Working with David Mincer, he and I designed this game from the ground up, primarily focusing on the intricacies of the software design. This report outlines decisions made and why, as well as a reflection on the whole process, including what was learnt and what could have been done better.

# Game Design

The idea which David and I came up with for our game is based on the game of Dodgeball, where the rules are, at the simplest level, to hit the other team with the ball, while not getting hit yourself (Dodgeball Rules). This first-person game will feature the core mechanic that involves throwing balls at enemies with intent to deal damage or knock them off platforms, all while they attempt to do the same to you. Victory is achieved by reaching the end of the level having beaten all the enemies and not dying. The theme of the game will be that of fire and hell, taking inspiration from games such as DOOM, see figures 1 & 2 below, with the game being set over an endless pool of lava and the enemies being fire hurling demons. Overall the graphics will be simple but bright, combining lots of red and orange, with the whole scene being lit by the lava underneath the level. The player will have a mana resource limiting their attacks, something that is a classic feature of many games where often “spellcasters rely on mana” (Alex Golub, 2014), such as World of Warcraft.



Figure : An example of what the lava could look like (DOOM 2016)

Figure : An example of what the terrain will look like (DOOM 2016)



Figure : The blue bar is a mana bar from (World of Warcraft 2004)

# User Stories

Together David and I worked through our game idea and wrote out a list of user stories (ref appendix) to determine all the necessary aspects of our game that needed designing. The core of the game is of course the player which has the most components, largely focused around user input. The other important part of the game is the demons/enemies, which pose the challenge to the player. We had some thoughts to have the demons use a mana-based attack system as well but settled on a cooldown system instead since they don’t move around as much. Another decision we made was to implement jumping. We chose to include it to increase the player’s capability for dodging fireballs. Without it we thought the game would feel to static and slow when it came to dodging attacks.

Core requirements/Alternate ideas?

# Software Design

The most important aspect of the game that needed designing was the player. Most of the player design was done by David, beginning with basic movement (ref appendix). As shown in the class diagrams (ref appendix), the player class is derived from the character class, a generic class that has movement capabilities. The enemy class also inherits from character as they too move around the level like the player. The movement itself will use vectors and then matrices to determine the direction of movement and then the translation required for that movement.

Equation?

Another key part of the player movement that we have chosen to implement is jumping. David designed his implementation using a counter system to time how long the player would remain in the air, before having them drop back down. I however would use a rough implementation of gravity in which I would give the player a vertical velocity, upon jumping, and have this decrease over time due to a deceleration by gravity. This would create a much more natural looking jump that would follow a realistic arc in which the player reaches a point of having no velocity, up in the air, before falling and speeding back up again.

The player in the game will be first person, represented by the camera. To create the world around the player I will use a world view projection matrix to ensure that everything in the level is in place, with correct transforms. The values in these matrices will be passed to the vertex shader, using the constant buffer, so that the shader can apply the necessary transforms to the objects. To do this I will be multiplying the WVP matrix by the position vector of the vertices.

Equation

Lighting?

The core mechanic of the game is the throwing of fireballs, which is done by both the player and the enemies. David’s design (ref appendix) outlines, how they will work, but a key difference between the two is the requirement to fire. The player will have a mana resource, represented in the UI (ref appendix) whereas the enemy will only be able to throw on a cooldown. The mechanics of this are shown in the flow chart for the enemy (ref appendix). We chose this method since the player needing to pick up mana refills makes them explore the level more, whereas the enemies won’t need to leave their designated zones.

The mana bar will be very simple (ref appendix). The player will have an integer value denoting remaining mana, which directly correlates to how many fireballs can be thrown. For obtaining more mana, the player will need to walk over the mana pickups designed by David (ref appendix). These will replenish mana, a change that will be reflected in the UI.

Another system that I designed is a simple particle system. The particles will be used to make the lava look like it is bubbling up, helping to create the atmosphere we are trying to cultivate. They will work on a random timer and at a random location, using the activate method I outline in the design (ref appendix). The system will of course have multiple collections of particles to allow for more than one effect to occur at once.

Demon AI?

Menus?

# Testing Plans

Initially I planned to include a plan for some ad-hoc testing in my appendix. However, through my research I found that ad-hoc testing is “a method of software testing without any planning and documentation” (Software Testing Fundamentals). So instead I will test components randomly, as is the definition of ad-hoc testing. I did however create a white box test play (ref appendix). I plan to use this to ensure that the most integral parts of the code do what they are intended to once the full game has been completed. It is important to test this since any ad-hoc testing would only test the components individually, not when combined as a full unit with others. For this I chose tests based on my analysis of the internal structure of the systems (Software Testing Fundamentals). I also created a black box testing plan (ref appendix) which I will use after all other testing is completed. Using this I can have external testers test the game and ensure all the features work as intended on the surface, all ready knowing that it works fine in the code, since black box testers need no knowledge of the internal structure (Software Testing Fundamentals).

# Work Breakdown Structure & Gantt Chart

The WBS (ref appendix) was made by David and shows all the tasks necessary to completing the game. Most of the tasks are derived from the user stories (ref appendix) or from the breakdown of the user stories in the software design. David also put together the Gantt chart (ref appendix) which shows how log each task will take and when they will be completed. Using the Gantt chart, I can see the critical path that must be taken when developing the game to ensure that the most important tasks are completed on time, to not set the project behind.

# Reflection

Working on this design with David showed a lot of strengths of our ability to design together. We created some detailed designs which I believe will be very useful when it comes to developing the game. We also have the help of a completed WBS and Gantt chart to help us with our own implementations of the game. Working on the testing plans also taught me more about the differences between he different types of testing. I think in the future I will be able to be more thorough with my testing as I apply different techniques than I knew before. I was also reminded how important it is to backup our work to not lose anything important. I have created a GitHub repository (CGP600 – AGP - Assignment) to store my project in to ensure safe backups as I develop the project. It also will allow me to roll back any changes if I make a mistake or break something.

Some of the weaknesses of my software design is the simplicity in certain areas. Looking back, I think that my AI for the demon could be more sophisticated. Since designing it, I have learnt more about different AI implementations and will likely look to use one in the future when I develop the game. Our design plans for the enemies to only path between two points and so for that task my design is suitable, however, implementing something such as a state machine would allow for more functionality, creating a more engaging enemy.

One problem that we identified in out work was the lack of detail in the Gantt chart. After looking through the tasks laid out in the WBS (ref appendix), David set to work revamping the Gantt chart to include much more detail. This led a more in-depth Gantt chart (ref appendix) which shows all the necessary tasks that would go into developing this game.

One of my weaknesses which hindered our progress throughout this assignment is my time management. I could have made more effort to structure when I would do tasks and plan how long they would take. Since I had a partner, it made it more important that I complete tasks when necessary and in some regards that was not the case here. In the future I will use what I have learnt from the WBS and Gantt chart in this assignment to help me better work on other assignments, without having timing problems.

Overall, I would say our biggest weakness in this assignment was our lack of collaboration. Our plan to complete the design was to allocate an even number of tasks to each of us and then design them on our own. We did this, and while we did collaborate on some tasks, I now wish we had done so on more. A lot of the software design for this gam is intertwined and if I had a chance to do this project again, I would try to work on more design as a team rather than hoping to put it all together individually. This would allow for not only a more coherent design, but for us each to improve upon aspects that maybe we think are not up to standard.

# Bibliography

## Images

Figure 1 - “Lava Example” - DOOM (2016) –

<https://guides.gamepressure.com/doom/guide.asp?ID=35388>

Figure 2 - “Arid Terrain Example” – DOOM (2016) -

<https://guides.gamepressure.com/doom/guide.asp?ID=35390>

Figure 3 - Gello – Adapt Addon – “Mana Bar Image” – World of Warcraft (2004) -

<https://wow.curseforge.com/projects/project-2341/images>

## References

Title: Dodgeball Rules

Website Title: Web.mst.edu

URL: <http://web.mst.edu/~ima/Rules/dodgeballrules.html>

Title: The History of Mana

Author: Alex Golub

Publication Date: June 17, 2014  
Website Title: The Appendix

URL: <http://theappendix.net/issues/2014/4/the-history-of-mana-how-an-austronesian-concept-became-a-video-game-mechanic>

Title: Ad hoc Testing

Website Title: Software Testing Fundamentals

URL: <http://softwaretestingfundamentals.com/ad-hoc-testing/>

Title: White Box Testing

Website Title: Software Testing Fundamentals

URL: <http://softwaretestingfundamentals.com/white-box-testing/>

Title: Black Box Testing

Website Title: Software Testing Fundamentals

URL: <http://softwaretestingfundamentals.com/black-box-testing/>

Title: CGP600 – AGP – Assignment

Author: Alex Pearson

Website Title: GitHub

URL: <https://github.com/Shades02/CGP600-AGP-Assignment>

# Appendix