Jack Porter

[Email address]

Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

[Document title]

[Document subtitle]

Contents

[Analysis and Design 2](#_Toc528750826)

[Functionality and core requirements 2](#_Toc528750827)

[Flow/Class Diagrams 2](#_Toc528750828)

[Development Techniques 2](#_Toc528750829)

[Object Oriented Design 2](#_Toc528750830)

[Task Breakdown and Rational 2](#_Toc528750831)

[User Stories 2](#_Toc528750832)

[Work Breakdown Structure 2](#_Toc528750833)

[Critical Paths, Tasks, Timescale, Dependencies Grid Tasks and Times relating to WBS 3](#_Toc528750834)

[Testing Plans 3](#_Toc528750835)

[Critical Reflection and Discussion of Group Work 3](#_Toc528750836)

[Evidence of Equal Distribution of Work 3](#_Toc528750837)

[Reflection of the Design Process 3](#_Toc528750838)

[Identification and Resolution of Problems 3](#_Toc528750839)

[Software Backup Methodology 4](#_Toc528750840)

# Analysis and Design

## Functionality and core requirements

The game that I have designed with my group that I will be making is a 3D platformer inspired by games like Super Mario 64 [APPENDIX]. The player will be tasked with navigating the world picking up 8 map pieces to locate the treasure chest which must be collected to complete the level. In the world are enemies that will patrol between points and if the player gets too close to them, they will chase the player down. The player can kill the enemies by jumping on top of them and will take damage if they collide with the enemy in any other way. If the player takes damage they will be able to collect coins to heal themselves.

## 

## Flow/Class Diagrams

Ed took the lead role on the class diagrams [APPENDIX] and flow charts [APPENDIX], however we all had input into their design and we made sure that we were in constant review of the design so that we could make changes that suited our needs.

## Development Techniques

For collision in this project I plan on using capsule collision on the player and enemies, sphere collision on the pick-ups and box collision for the environment.

## Object Oriented Design

# Task Breakdown and Rational

## User Stories

We looked at the core requirements for this development product and at our basic game design that we had created and set about writing the user stories for this project. We thought about both the stories from the player’s perspective, as well as from the client’s perspective, trying to cover every one of the core requirements in our user stories and every aspect of the gameplay.

The user stories can be found below:

* As a Player I want to Move around so that I can navigate the level
* As a Player I want to Jump so that I can kill enemies by landing on them, help navigate the level

and activate buttons

* As a Player I want to Collect coins so that I can increase my score
* As a Player I want to Collect map pieces so that the chest will spawn
* As a Player I want to Avoid touching enemies so that they will not damage me
* As a Player I want to See enemies navigate the world so that the world feels more alive
* As a Player I want to Navigate around obstacles so that I can solve problems and have fun
* As a Player I want to Collide with the world and entities in the game so that the world is shown as a set of obstacles to navigate
* As a Player I want to Stay over 0 health so that my character doesn't die
* As a Client I want to Enemies to move between nodes so that it shows patrolling in order to make enemy movement more interesting to player
* As a Client I want to Have enemies make use of a finite state machine so that it breaks up behaviour into states making code cleaner and easier to debug
* As a Client I want to have the game be 3D and written with Object Oriented C++ using Visual Studio and DirectX so that the game fulfils the requirements
* As a Client I want to have objects in the environment have textures and lighting so that the game looks presentable while showing off technical features

## Work Breakdown Structure

Lewis designed the WBS for this project [APPENDIX], using the user stories that we created as a group, he broke them down into tasks representing their smallest form that encompasses a module, after this, we all went over it as a group to make sure that we were happy with those tasks, we didn’t feel like any changes had to be made to these as Lewis had done a good job.

## Critical Paths, Tasks, Timescale, Dependencies Grid Tasks and Times relating to WBS

Using the WBS that was created, I then took all of the tasks and formatted them in to a Gantt chart [APPENDIX]. I attempted to change the daily working hours and add an hourly work field to the chart, however they caused more problems than they solved and so I opted to just use days as a measure for working time. Since we are all students and have many other units as well, I assumed that we would have less time in a day to work on this project than if we were working solely on it, as such the day is assumed to have around 4 hours of working time instead of the default 8 hours.

For the time allocated to each task I estimated how long I would take to complete a task, I also included some extra time to allow for some testing of the feature that will be implemented. Finally I took the time that I was left with and doubled the value, this allows for some wiggle room if a task is particularly tough to figure out.

After I had finished I had the rest of my team look and go over it to make sure that they were happy with the critical path shown by the Gantt chart and also with the time allocated to each task, thankfully they were pleased with my estimations and were happy with the final result.

## Testing Plans

I took on the lead role for designing and building the testing plan for this project [APPENDIX], I have based the design around 2 types of black box testing, two types of white box testing and a section for logging ad-hoc testing

The black box testing that will be used is Systems testing, which makes sure that the program is meeting the functional requirements of the design and is the first level of testing of the product as a whole. We are also doing acceptance testing, which is used to test the product against the client’s requirements to make sure that they meet the design specified at the start of the project.

The white box testing that will be used is unit testing, which allows a developer to test specific functions or areas of code against a confirmed outcome to make sure that the code is performing as it should be. We are also going to include bottom up integration testing, which requires us to build and test each system from the lowest level upwards to make sure that each system will be working completely alone and with others before moving onto the next one.

# Critical Reflection and Discussion of Group Work

## Evidence of Equal Distribution of Work

## Reflection of the Design Process

## Identification and Resolution of Problems

## Software Backup Methodology