Design Document ZombieShooterGame

Kjell Vos

vos.kjell@gmail.com

6

Table of contents888

[Table of contents 2](#_Toc44541879)

[Introduction 4](#_Toc44541880)

[z](#_Toc44541881)

[Scope 4](#_Toc44541882)

[Definitions, Acronyms and abbreviations 4](#_Toc44541883)

[References 4](#_Toc44541884)

[Overview 4](#_Toc44541885)

[Contact details 5](#_Toc44541886)

[Main programmer Kjell Vos: 5](#_Toc44541887)

[Statement of Goals 5](#_Toc44541888)

[Use cases 6](#_Toc44541889)

[Actors 6](#_Toc44541890)

[Gamer 6](#_Toc44541891)

[List of use cases 6](#_Toc44541892)

[Use case diagrams 6](#_Toc44541893)

[Use cases 6](#_Toc44541894)

[Current situation 7](#_Toc44541895)

[Functionality 7](#_Toc44541896)

[User interface 7](#_Toc44541897)

[Desired situation 7](#_Toc44541898)

[Functionality 7](#_Toc44541899)

[User interface 7](#_Toc44541900)

[Design overview 7](#_Toc44541901)

[Introduction 7](#_Toc44541902)

[System architecture 7](#_Toc44541903)

[System interfaces 7](#_Toc44541904)

[Constraints and assumptions 7](#_Toc44541905)

[System object model 7](#_Toc44541906)

[Introduction 7](#_Toc44541907)

[Subsystems 7](#_Toc44541908)

[Subsystem interfaces 7](#_Toc44541909)

[Object descriptions 7](#_Toc44541910)

[Objects 7](#_Toc44541911)

[Object collaboration 7](#_Toc44541912)

[Object collaboration diagram 7](#_Toc44541913)

[Data design 7](#_Toc44541914)

[Entity relationship diagram 7](#_Toc44541915)

[Dynamic model 8](#_Toc44541916)

[Sequence diagrams 8](#_Toc44541917)

[State diagrams 8](#_Toc44541918)

[Non- functional requirements 8](#_Toc44541919)

[Performance requirements 8](#_Toc44541920)

[Design constraints 8](#_Toc44541921)

[Supplementary documentation 8](#_Toc44541922)

[Tools used to create diagrams 8](#_Toc44541923)

# Introduction

This Software Design Document is made to provide documentation to aid in software development by providing details for how the software should be built. Within the Software Design Document are narrative and graphical pieces of information. This includes use case models, sequence diagrams, collaboration models, object behaviour models and other supporting requirement information.

## Purpose

This documents purpose is to provide a description of the design fully enough to allow for software development to proceed with an understanding of what is to be built and how it is expected to be built.

## Scope

The scope of this document is for a base level system. Which will work as a proof of concept for the use of building a system that provides base level of functionality to show feasibility of the system.

## Definitions, Acronyms and abbreviations

## References

## Overview

# Contact details

### Main programmer Kjell Vos:

[Vos.kjell@gmail.com](mailto:Vos.kjell@gmail.com)

[Website: kjell-vos.com](http://kjell-vos.com)

# Statement of Goals

The goals for this project are that it becomes a fully fledged single player game. You can choose from a couple of maps then after you have chosen a map you can choose your player character from four characters, each with different magical abilities and are attuned to different weapons.

Once the game starts after picking the map and character you are dropped in a world where you notice a room with a beating heart that you have to protect. How the player does this is by building a long channel of towers which the monsters will travel through towards the heart, The first round there would be a few enemies and then every next round a couple more. If the player kills all the enemies a timer starts after which the next round of enemies comes. Players can use the time in between rounds to gather resources used for weapons and towers.

If the player blocks entrance to the heart with towers the monsters will start attacking the towers till a tower is destroyed and a route is found towards the heart.

* Working physics world
* Working lighting engine
* Working pathfinding for monsters
* Player can gather resources (Wood, iron etc)
  + Player can spend these resources on weapons, Armor and towers
  + There is a special create screen in the pause menu
* Player has inventory
  + Inventory supports stacking items
  + The inventory can be found in the pause menu
* Towers can be placed in the world
* Towers attack the monsters
  + Towers have different abilities(Freeze, burn, armor piercing etc)
* Monsters spawn near the edge of the map every once and a while on a timer
* Monsters have different abilities
* The sprites for the game are all open source and free to use or created by the team
* The music and sound effects in the game are all open source and free to use or created by the team
* The game has a tight time bound gameplay style where you feel like you need to always be busy
* The game has a creepy ‘in the dark’ aesthetic

# Use cases

## Actors

### Gamer

The gamer is the only use case we will focus on. The gamer starts the application to play a fun game, Once booted and past the splash screens he will land on the main menu. From here the player has the possibility to open the settings screen, setup the game, and quit the game.

In the settings screen the gamer can change the music and sound effect levels to their hearths desire, The settings screen can only quit back to the main menu.

In the setup a game screen the player can choose which map to use and which character he wants to play, after choosing both the player can start the game. The gamer can also exit out of the setup a game screen back to the main menu.

## List of use cases

#### Gamer - Change settings

#### Gamer - Setup a game

#### Gamer - Play a game

## Use case diagrams

## Use cases

# Current situation

## Functionality

## User interface

# Desired situation

## Functionality

## User interface

# Design overview

## Introduction

## System architecture

## System interfaces

## Constraints and assumptions

# System object model

## Introduction

## Subsystems

## Subsystem interfaces

# Object descriptions

## Objects

# Object collaboration

## Object collaboration diagram

# Data design

## Entity relationship diagram

# Dynamic model

## Sequence diagrams

## State diagrams

# Non- functional requirements

## Performance requirements

## Design constraints

# Supplementary documentation

## Tools used to create diagrams