Design Document ZombieShooterGame

Kjell Vos

vos.kjell@gmail.com

6

Table of contents

[Table of contents 2](#_Toc45181752)

[Introduction 4](#_Toc45181753)

[Purpose 4](#_Toc45181754)

[Scope 4](#_Toc45181755)

[Definitions, Acronyms and abbreviations 4](#_Toc45181756)

[References 4](#_Toc45181757)

[Overview 4](#_Toc45181758)

[Contact details 5](#_Toc45181759)

[Main programmer Kjell Vos: 5](#_Toc45181760)

[Statement of Goals 5](#_Toc45181761)

[Use cases 6](#_Toc45181762)

[Actors 6](#_Toc45181763)

[Gamer 6](#_Toc45181764)

[List of use cases 6](#_Toc45181765)

[Use case diagrams 6](#_Toc45181766)

[Use cases 6](#_Toc45181767)

[Current situation 7](#_Toc45181768)

[Functionality 7](#_Toc45181769)

[User interface 7](#_Toc45181770)

[Desired situation 7](#_Toc45181771)

[Functionality 7](#_Toc45181772)

[User interface 7](#_Toc45181773)

[Design overview 8](#_Toc45181774)

[Introduction 8](#_Toc45181775)

[System architecture 9](#_Toc45181776)

[System interfaces 9](#_Toc45181777)

[Constraints and assumptions 9](#_Toc45181778)

[System object model 9](#_Toc45181779)

[Introduction 9](#_Toc45181780)

[Subsystems 9](#_Toc45181781)

[Subsystem interfaces 9](#_Toc45181782)

[Object descriptions 9](#_Toc45181783)

[Objects 9](#_Toc45181784)

[Object collaboration 9](#_Toc45181785)

[Object collaboration diagram 9](#_Toc45181786)

[Data design 9](#_Toc45181787)

[Entity relationship diagram 9](#_Toc45181788)

[Dynamic model 9](#_Toc45181789)

[Sequence diagrams 9](#_Toc45181790)

[State diagrams 9](#_Toc45181791)

[Non- functional requirements 9](#_Toc45181792)

[Performance requirements 9](#_Toc45181793)

[Design constraints 9](#_Toc45181794)

[Supplementary documentation 9](#_Toc45181795)

[Tools used to create diagrams 10](#_Toc45181796)

# 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

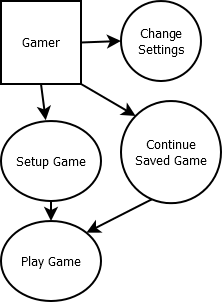
#### Gamer - Change settings

#### Gamer - Setup a game

#### Gamer - Play a game

## Use case diagrams

## Use cases



# Current situation

## Functionality

The gamer can move the main character around the map, The walls on the edge of the map keep the player inside the map. The gamer can see items on the ground which the gamer can pick up. Once picked up they are stored in the inventory. There is currently no functionality for looking at your inventory or dropping items. The gamer can go into a little house in the centre of the map where a monster is inside, If the gamer gets too close with their character to the monster the monster will start following the gamer. If the gamer get’s far away from the monster the monster will stop following having lost the gamer and wander around where the monster last saw the player.

## User interface

The user interface is pretty basic, Once the game is started you see the starting splashes(Just one now). After pressing any button the game shows the main menu where you can start the game. There is also a settings screen where the gamer can change music/sound effect levels.

# Desired situation

## Functionality

The gamer starts the game and goes through the splashes. Once in the main menu the player can continue an old game or start a new one. The gamer can also access the settings screen from the main menu where the gamer can change sound volume and other relevant settings.

If the gamer chooses to start a new game the gamer gets to pick a map, Each map is different from each other. Some maps will be bigger some smaller, The gathering spots also change for each map. After picking a map the gamer chooses a character. All characters have different abilities from each other.

Once the game is started the gamer can go and gather some resources before the first wave of enemies gets released. The gamer can use these resources to craft towers and weapons. If the gamer crafts a tower the tower gets placed into the gamer’s inventory. From the inventory the gamer can place the tower on the map. The tower automatically attacks enemies.

## User interface

I want a simple user interface, no extra bell’s and whistles. Once the gamer starts the game the application launches and shows the splash screens. The gamer can go through these by pressing any button. Once at the main menu the gamer can choose to continue a saved game or start a new one. If the gamer chooses to start a new one a new screen is shown where the gamer can pick a character and map. After choosing these the gamer can start the game.

In the game the only thing that is needed is a pause menu where you can craft and see your inventory. And a simple HUD for health/mana.

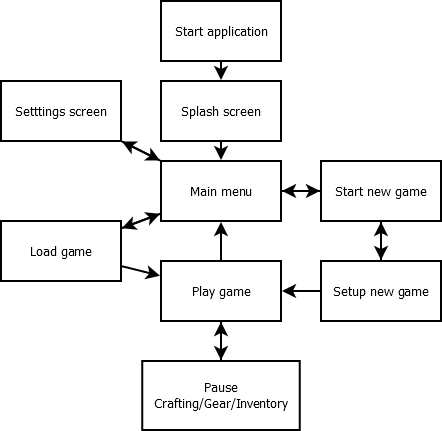
# Design overview

## Introduction

The Design Overview is a section to introduce and give a brief overview of the design. The System Architecture is a way to give the overall view of a system and to place it into context with external systems.

## System architecture

#### Views



## 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