Requirements and Analysis Document for Proximity

Date: 4/5 2015

Author: Group 11

This version overrides all previous versions

1. Introduction

1.1. Purpose of application

The purpose of this project is to create an entertaining game in the popular genre of Tower Defence. To differentiate from other tower defence games this project will have a bigger focus on direct user interactions. This will be accomplished by including a spell system. For information about the Tower Defence genre, see appendix 1.

1.2. General characteristics of application

The application will be a cross-platform desktop offline game runnable on Mac/Windows/Linux platforms. The game will be scalable and adapt to different screen sizes.

The game will be a real time strategy (RTS) game based on the tower defence concept. It will be built of the basic components of a tower defence game: it will feature a map with enemies approaching the players base and the player will place towers and spells to kill enemies and protect the base.

The enemies will be generated in waves. These waves will continue to be generated until the player base has no more health or until the player has defeated a certain amount of waves. Devolving enemies will yield experience and resources (money).

The player will be able to choose a faction before starting a certain map. The player will always have four spells, but depending on what faction is chosen the player will have different kinds of spells at their disposal. The faction also affects the design of some artwork.

It is possible to level up a faction. This is done automatically when experience is gained.

1.3. Scope of application

The application will have at least one playable map with finished artwork, enemies, working towers and an end game scenario. Between sessions the game will store the player data. This includes the different faction levels the player has.

1.4. Objectives and success criteria of the project

1. It should be possible to chose a map, buy towers with resources and cast spells. Enemies should be approaching on the map and if they reach the base it should lose health. Enemies should yield experience and resources when devolved.

2. The game should save the player data when it is closed.

3. The game should be relatively balanced, several strategies should be possible and viable to use.

1.5. Definitions, acronyms and abbreviations

- **TD**, Tower defence
- **Experience**, Points gained from killing enemies. When a certain amount of experience is collected the next level is reached.
- **Level**, The players factions has a faction level. It is a display of how much experience the player has with that faction.
- **Wave**, The enemies come in clusters, the current wave is the cluster the player is currently facing.
- **Faction,** A theme with a spell-set that the player can chose from; different factions offer different spells. The player can level up their factions.
- **Tower:** A block that the user can place on the screen. The tower costs resources. Most towers fire projectiles at enemies or attack them in some other way.
- **Projectile:** Most towers can shoots projectiles that hit the enemies, these projectiles affect the creep(s) it hits, for example by freezing or devolving them.
- **Creep:** Enemies that follow a certain path in the game and tries to kill the player. These can be of different types and strengths.
- **Devolve:** When an enemy dies, it “reincarnates” as a smaller enemy type, if it’s the smallest type of enemy the enemy gets destroyed
- **Resource:** “Money” that the player can use to purchase towers, different towers require different resources.
    - Line, A type of resource
    - Point, A type of resource
    - Polygon, A more valuable type of resource
- **Upgrade:** The player can upgrade towers to make them better
- **Spell:** The player gets four spells from the faction and this spell can be used to affect the game in the players favor

2. Requirements

2.1. Functional requirements

The player should be able to:

1. Change the game speed:
    a. Pause
    b. Play (normal speed)
    c. Fast forward
2. Start a new game
    a. Choose a map
    b. Choose a faction
3. Play the game
    a. Choose and place towers
    b. Go to the main menu
    c. Choose and place spells
4. Change settings
    a. Change the volume
5. Exit the game. The faction levels will be stored.

The application should:

1. Generate creep waves

2.2. Non-functional requirements

2.2.1. Usability

The game should not require a manual or a lot of previous experience to use except basic computer knowledge.

User tests will be performed to improve the usability. The results of the tests will be included in the final documentation.

2.2.2. Reliability

Not applicable.

2.2.3. Performance

The interaction with the application should always seem instantaneous to the user.

2.2.4. Supportability

The application should be possible to run on Mac, Windows and Linux platforms. It should be easy to adapt the application to also run on a website (HTML5/Javascript) or an android or iOS device.

There should be automated tests for all use cases. GUI based use cases will be tested and manually recorded. The results of these tests will be included in the final documentation.

2.2.5. Implementation

To make the application possible to run on all specified platforms the library LibGDX will be used.The bundled JVM (Java virtual machine) takes care of cross-platform compatibility.

2.2.6. Packaging and installation

The application will start from an executable file, no installation will be required.

More will be added on this section.

2.2.7. Legal

The LibGDX library is distributed with the Apache2 license. The documented is not allowed to be obfuscated.

The term “tower defence” is trademarked. This term can’t be used in the game name, however the term can still be used as a reference to the game genre.

2.3. Application models

2.3.1. Use case model

See APPENDIX for UML diagram of use cases.

2.3.2 Use cases priority

1. Buying Tower
2. Spell use - area effect
3. Spell use - global effect
4. Spell use - tower effect
5. Upgrading tower
6. Exit game
7. Press Play

2.3.3 Domain model

See Appendix for the analysis model.

2.3.4 User interface

The design of the user interface will be based on a theme of geometrical figures. The game will be scalable and adapt to different screen resolutions.

Se appendix for the preliminary GUI.

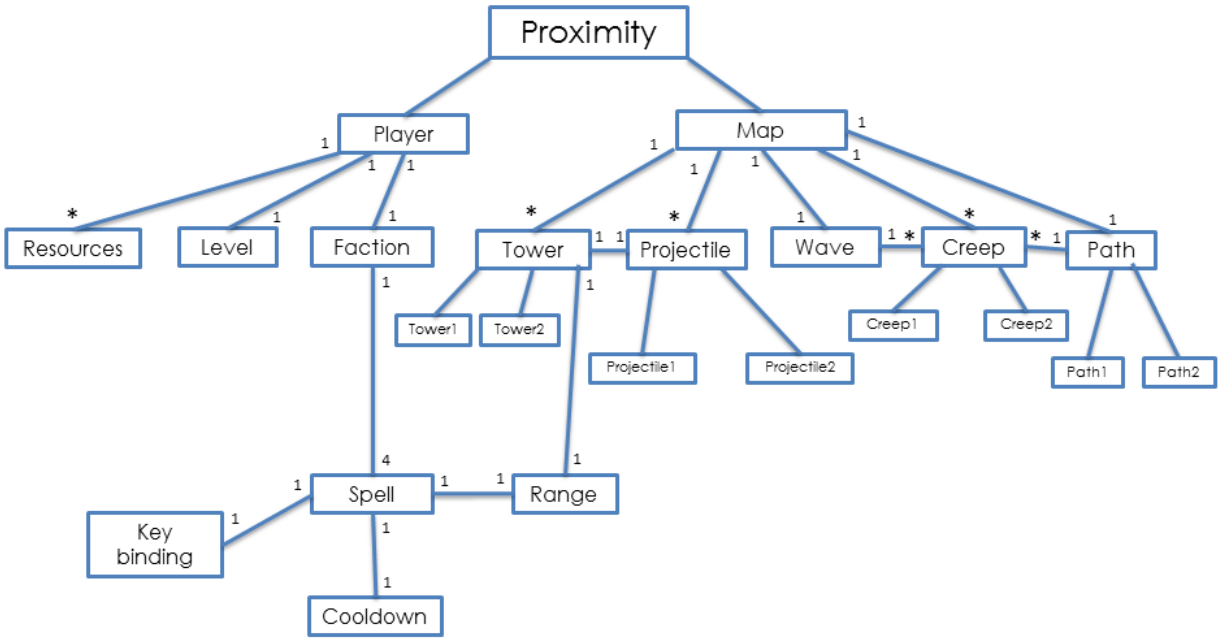
2.4 References

Tower defence: http://sv.wikipedia.org/wiki/Tower_defense

APPENDIX

Analysis model

This is the preliminary analysis model for Proximity



GUI

Preliminary GUI

Use cases

A graphic image of the use cases will be added here

Use case texts

This will be added later.

The tower defence genre

A tower defence game is a strategy game. The player decides where to position different kinds of towers on a map, and enemies will walk on this map to approach the players “home base”.

The strategy element consists of the player choosing which towers to buy, and where to position them. These towers often cost different amount of resources and have different effects, most often the towers shoot out projectiles on nearby enemies which cause damage or destroy the enemies.

The player loses health if the enemies reach the players base, as a result of the player not building a sufficient defence. A normal tower-defence round can last between 5-20 minutes, at that time the player has either died as a result of the approaching enemies, or has survived long enough that he has passed a limit which is considered a win-condition.

The player can then start a new round, where he will once again have to build up a sufficient defence.