McGill University

# Tower Defense - Software Requirements Specifications

*Team 6: Amlekar R, Aydede E, Gupta Y, Wright A*

*ECSE 321: Introduction to Software Engineering*
*Dr. Sinnig*

*2015/02/15*

## Revision History

| Date | Changes to Document | Version | Changes made by |
|---|---|---|---|
| 2015/02/08 | Creation of Document | 1.0 | Team 6 |
| | | | |

## Table of Contents



Note: This document uses intra-document references. To navigate from the Table of Contents, click the page number on the right. If there is an underlined Keyword, clicking or ctrl-clicking it will move the document to the Glossary.

# Illustrations

# Project Description

## Introduction

The Tower Defense Game (TDG) is a video game playable on personal computers. The goal of the TDG is to stop the Critters from reaching the end of the path on the map by creating Towers that shoot them. When a Critter is killed, the Player receives money that can be used to build or upgrade Towers. Players must attempt to survive the most number of Critter Waves.

## Context

The Tower Defense Game (TDG) is designed with the versatility to create and play on custom created Maps. Players may create a Map using the Map Editor or can choose an existing Map to play on.

## Business Goals

The successful outcome of this project will be a game that allows for the Player to create a Map and then place Towers on the map to defend against incoming waves of Critters.

## Scope

The Tower Defense Game (TDG) can be complex but given the project timeframe and for the sake of simplicity and elegance, certain features are omitted. They include:

Multiplayer feature:

One might expect a survival mode where players can compete against each other in real time, to outlast the Critters longer than their opponent. Although this would be a great feature, it would require a longer timeframe to implement.

3D graphics:

Since we are developing a Tower Defense Game, one might expect it to not only have a state of the art game engine, but also have immersive graphics. We will only be implementing the game engine, as time spent working on the engine will produce higher value than better graphics. The graphics will be implemented from a library.

## Domain Model

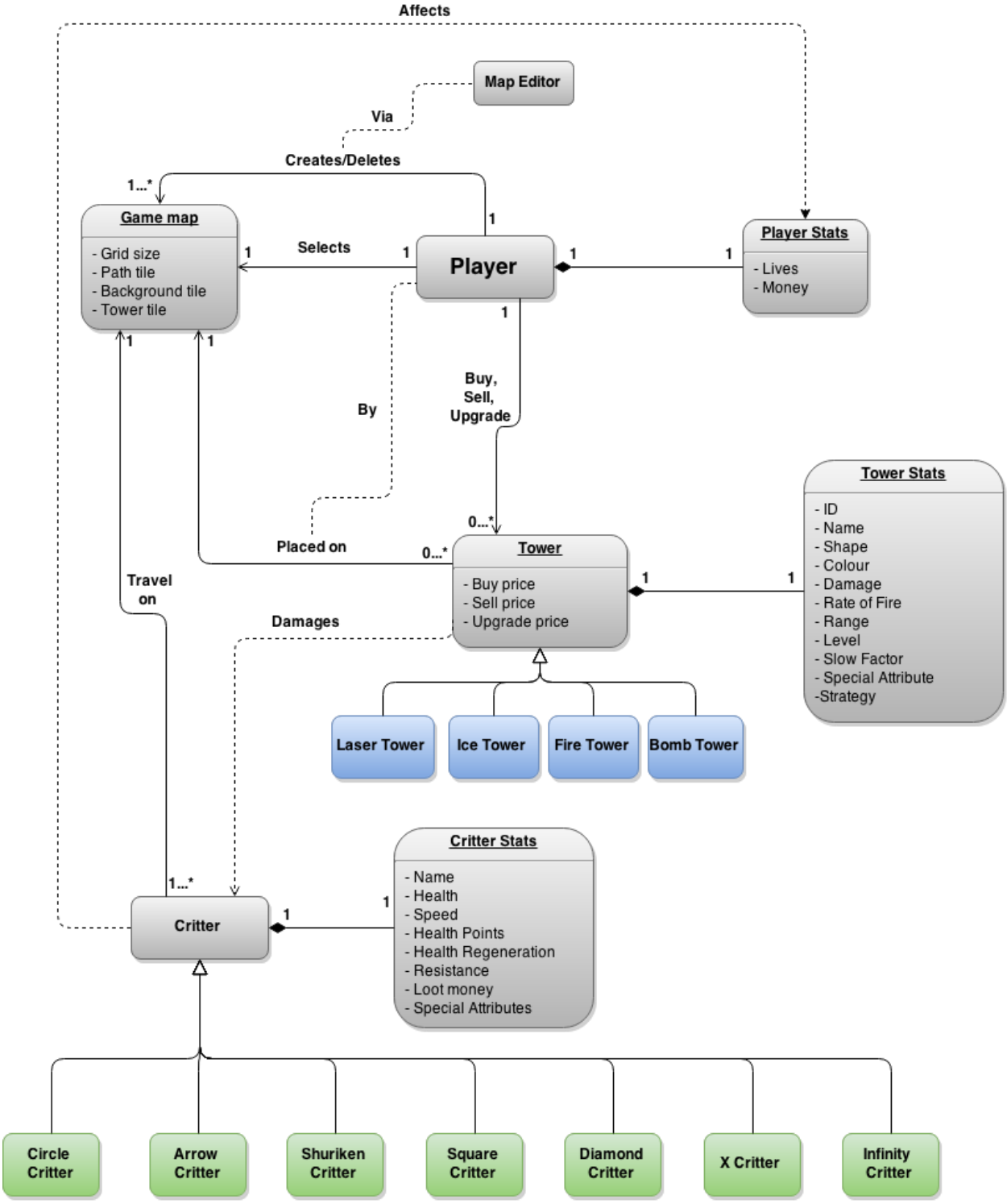


Figure 1 – Domain Model

**Actors**

The Player is the primary actor. The player plays the Tower Defense Game and is in charge of managing towers.

## Functional Requirements

### Overview

There are two following use case diagrams. The first is a general overview of the Tower Defense Game, labeled “Play Game”. This use case diagram encompasses the entirety of the game. The second use case diagram is the bubble labeled “Manage Tower”, in the first use case diagram.

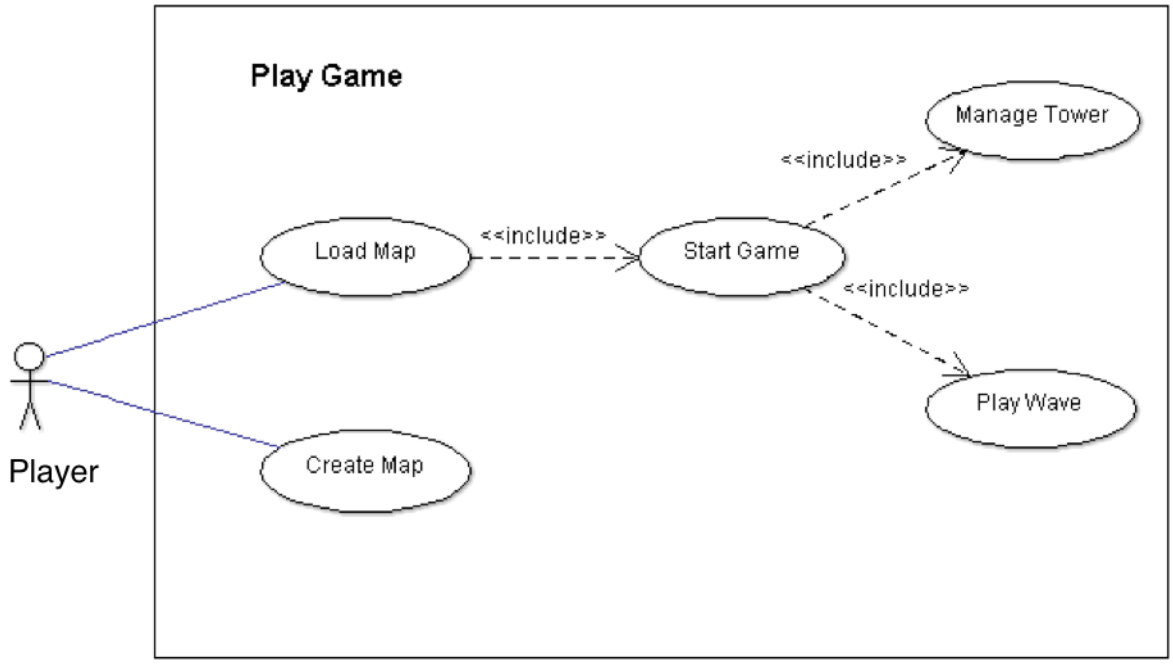


Figure 2 – Use Case Diagram: Play Game

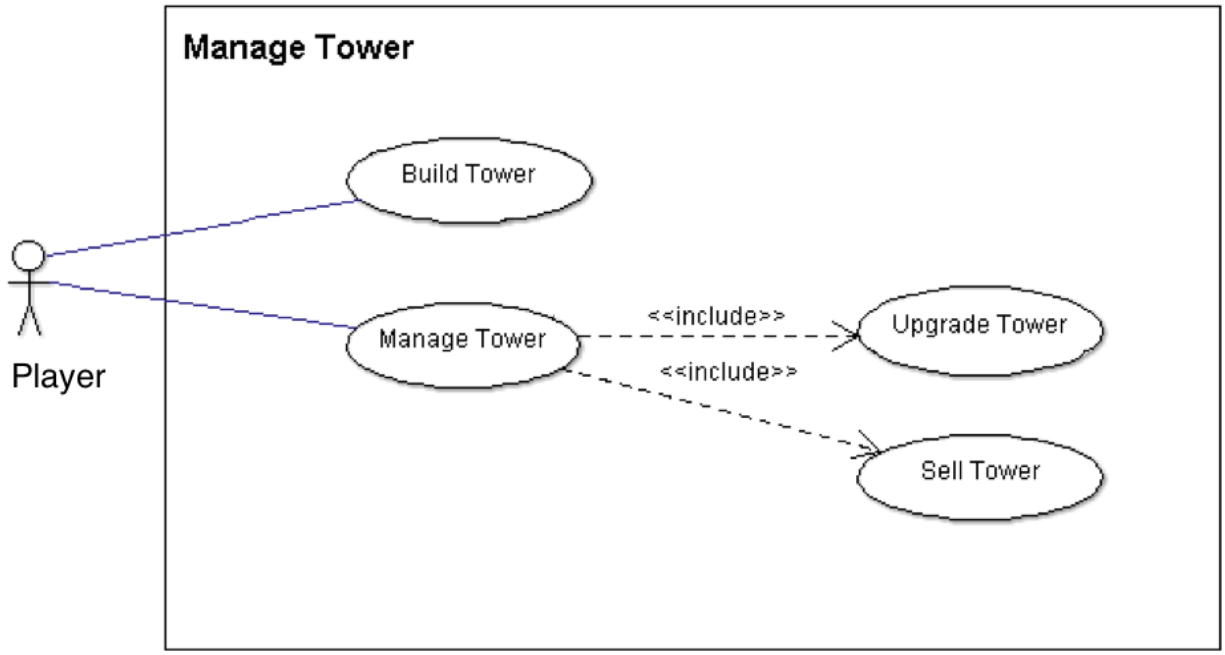


Figure 3 – Use Case Diagram: Manage Tower

## Use Cases

### Use Case: Play Game

*Successful Outcomes: The Player plays and completes the whole Tower Defense Game.*

| | |
|---|---|
| **Use Case Package** | Tower Defense |
| **ID** | UC-TD-PG |
| **Use Case Goal** | The primary actor plays and completes the Tower Defense Game |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player has opened the Game. |
| **Domain Entities** | Game, Game Map, Wave, Tower |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to play the game | |
| **2** | System prompts the Player to select an existing Game Map | |
| **3** | Player selects game map | |
| **4** | System prompts the Player to prepare for a Wave | |
| **5** | Player manages towers to prepare for the Wave | Use Cases UC-TM-1, UC-TM-2 and UC-TM-3 |
| **6** | Player selects to Play a Wave | Use Case UC-TD-PW |
| **7** | Return to Main Success Scenario Step 4 as long as Player has Lives, else proceed to Main Success Scenario Step 8 | According to GL-Player |
| **8** | System indicates to Player that the game is over and returns to main menu | |
| **9** | Use case ends successfully. | |

*3a. Player selects to make a new Game Map:*

| Step | Action | Notes |
|---|---|---|
| **3a.1** | Player Creates a new Map | Use Case UC-TD-Map |
| **3a.2** | Return to Main Success Scenario Step 4 | |
| **3a.3** | Use case ends successfully. | |

**Use Case: Create a new Game Map**

*Successful Outcomes: The Player makes a new Game Map.*

| | |
|---|---|
| **Use Case Package** | Tower Defense |
| **ID** | UC-TD-Map |
| **Use Case Goal** | The Player makes and saves a new Game Map. |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player is on the Main Screen and selects to make a new Game Map |
| **Domain Entities** | Game Map, Map Editor, Path |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to make a new Game Map. | |
| **2** | System prompts the Player to choose a map background | |
| **3** | Player selects a map background. | |
| **4** | System prompts Player to select a Grid Size. | |
| **5** | Player specifies a Grid Size. | According to GL-Map |
| **6** | System prompts player to create a Critter Path | |
| **7** | Player makes and submits a potential Critter Path | |
| **8** | System validates the Critter Path and saves it to the Game Map. | According to GL-Map |
| **9** | Player indicates intent to save the Map. | |
| **10** | System saves the Map | |
| **11** | Use case ends successfully. | |

*Alternative Flows:*

*5a Player selects an Invalid Grid Size*

| Step | Action | Notes |
|---|---|---|
| **5a.1** | System displays a message that the selected Grid Size is invalid. | |
| **5a.2** | Player acknowledges the message. | |
| **5a.3** | Return to Main Success Scenario Step 5. | |

*8a Player Builds a Non Valid Critter Path*

| Step | Action | Notes |
|---|---|---|
| **8a.1** | System displays that the Critter Path is invalid and discards | |

| | | |
|---|---|---|
| | the Critter Path | |
| **8a.2** | Player acknowledges the fact | |
| **8a.3** | Return to Main Success Scenario Step 7 | |

**a Player discards the new Game Map*

| Step | Action | Notes |
|---|---|---|
| ***a.1** | Player indicates intent to discard the current Map. | |
| ***a.2** | System deletes the Map | |
| ***a.3** | Use Case ends unsuccessfully. | |

**Use Case: Build Tower**

*Successful Outcomes: The Player builds a new tower to prepare for the next Wave.*

| | |
|---|---|
| **Use Case Package** | Tower Management |
| **ID** | UC-TM-1 |
| **Use Case Goal** | The primary actor builds a new tower |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player is preparing for a Wave and selects to build a new Tower |
| **Domain Entities** | Tower, Map |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to build a new Tower | |
| **2** | System prompts Player to select a position to place new Tower on the Map | According to GL-Map |
| **3** | Player selects a position on the Map for the new Tower | |
| **4** | System displays available Towers. | According to GL-Tower |
| **5** | Player selects the Tower to be built | |
| **6** | System validates Player attributes | According to GL-Player |
| **7** | System builds a new Tower and modifies Player attributes | According to GL-Tower |
| **8** | Use case ends successfully | |

*2a. The selected position on the Game Map is invalid:*

| Step | Action | Notes |
|---|---|---|
| **2a.1** | System displays to the Player that the selected position on the Map is invalid. | |
| **2a.2** | Player acknowledges this fact | |
| **2a.3** | Return to Main Success Scenario Step 2 | |

*6a. Player does not have enough Resources to build Tower:*

| Step | Action | Notes |
|---|---|---|
| **6a.1** | System displays to the Player that he/she does not have enough Resources to build the Tower | According to GL-Tower |
| **6a.2** | Player acknowledges this fact | |
| **6a.3** | Use case ends unsuccessfully | |

**Use Case: Upgrade Tower**

*Successful Outcomes: The Player upgrades a Tower to prepare the next Wave.*

| | |
|---|---|
| **Use Case Package** | Tower Management |
| **ID** | UC-TM-2 |
| **Use Case Goal** | Primary actor upgrades an existing tower. |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player is preparing for a Wave and selects to upgrade a Tower. |
| **Domain Entities** | Tower, Map, Upgrade Price |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to upgrade a Tower. | |
| **2** | System prompts Player to select a Tower to upgrade | |
| **3** | Player selects one Tower to upgrade | |
| **4** | System displays available Tower upgrade options and the corresponding costs for the selected Tower. | According to GL-Tower |
| **5** | Player chooses one of the Tower upgrade options. | |
| **6** | System validates Player resources. | According to GL-Player |
| **7** | System upgrades Tower attributes and modifies Player resources. | According to GL-Tower |
| **8** | Use case ends successfully. | |

*Alternative Flows*

*4a No upgrades are available for the selected tower:*

| Step | Action | Notes |
|---|---|---|
| **4a.1** | System displays to the Player that there are no Tower Upgrades for the selected Tower. | |
| **4a.2** | Player acknowledges this fact | |
| **4a.2** | System returns Player to the Game Screen. | |
| **4a.3** | Use case ends unsuccessfully. | |

*6a Not enough Resources are available to apply the selected upgrade to the tower:*

| Step | Action | Notes |
|---|---|---|
| **6a.1** | System displays to the Player that there are not enough Resources to apply the selected Tower upgrade option. | According to GL-Tower |
| **6a.2** | Player acknowledges this fact | |
| **6a.3** | Return to Main Success Scenario Step 4. | |

**Use Case: Sell Tower**

*Successful Outcomes: The Player sells an existing Tower to prepare for the next Wave.*

| | |
|---|---|
| **Use Case Package** | Tower Management |
| **ID** | UC-TM-3 |
| **Use Case Goal** | The primary actor sells an existing tower |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player is preparing for a Wave and selects to sell a Tower. |
| **Domain Entities** | Tower, Map |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to sell a Tower. | |
| **2** | System prompts Player to select a Tower to sell | |
| **3** | Player selects the tower to sell | |
| **4** | System displays Tower Sell Price | According to GL-Tower |
| **5** | Player chooses to sell the Tower. | |
| **6** | System deletes Tower and modifies Player resources accordingly | According to GL-Tower |
| **7** | Use case ends successfully. | |

**a. Player chooses to not sell the Tower:*

| Step | Action | Notes |
|---|---|---|
| ***a.1** | System displays to Player that the Tower was not sold | |
| ***a.2** | Use case ends unsuccessfully. | |

**Use Case: Play Wave**

*Successful Outcomes: The Player plays one Wave of the Game without letting critters through*

| | |
|---|---|
| **Use Case Package** | Tower Defense |
| **ID** | UC-TD-PW |
| **Use Case Goal** | The primary actor completes a wave of the game without letting critters through |
| **Actor(s)** | Primary Actor: Player |
| **Level** | User-level goal |
| **Precondition** | The Player has already prepared for the Wave and selects to play the Wave. |
| **Domain Entities** | Tower, Player, Critter |

*Main Success Scenario:*

| Step | Action | Notes |
|---|---|---|
| **1** | Player indicates intent to start a Wave. | |
| **2** | System generates a finite amount of Critter to move on the Path | |
| **3** | Towers fire on Critters to prevent them from reaching the end of the Path. | |
| **4** | All critters are killed, and the system modifies player attributes according to the types and amounts of critters killed | According to GL-Critter |
| **6** | Use case ends successfully. | |

**a. A critter reaches the end of the path:*

| Step | Action | Notes |
|---|---|---|
| ***a1** | System modifies the Player attributes. | According to GL-Player |
| ***a2** | Use case ends unsuccessfully. | |

## Game Logic

### GL-Tower

| Tower Name | **Laser Tower** | **Ice Beam Tower** | **Fire Tower** | **Bomb Tower** |
|---|---|---|---|---|
| Tower Description | Average | Slowing, weak | Sets critters on fire for Damage Over Time (DOT) | Fires bombs which damage critters in an area |
| Build Price | 100 | 150 | 200 | 250 |
| Upgrade Price (level 2, 3, 4) | 300, 400, 500 | 200, 300, 400 | 400, 500, 600 | 400, 500, 600 |
| Sell Price | 50+(upgrade money spent)/2 | 75+(upgrade money spent)/2 | 100+(upgrade money spent)/2 | 125 + 0.5*(upgrade money spent) |
| Range | 150 | 200 | 125 | 100 |
| Damage | 25 | 12.5 | 25 | 50 |
| Slow Factor (out of 1.0) | 0.0 | 0.7 | 0.2 | 0.0 |
| Rate of Fire (beams/s) | 2.0 | 1.5 | 1.0 | 0.5 |
| Special Attributes | None | Slowing (as specified by slow factor) | Sets critters on fire for 25 HP/s for 5 seconds (total of 125 damage) | Hits target critter (10 damage); hits critters in 25 unit radius (2.5 damage) |
| Color | Blue | Green | Red | Yellow |

### GL-Player

| Game Mode | Starting Money | Number of Lives | Critter Money Multiplier | Maximum Time between waves |
|---|---|---|---|---|
| **Zen** | 1000000 | Infinite | 0.0 | Infinite |
| **Easy** | 600 | 20 | 1.5 | 60 seconds |
| **Medium** | 300 | 10 | 1.0 | 30 seconds |
| **Hard** | 200 | 5 | 0.5 | 15 seconds |

**GL-Critter**

| Critter Name | Critter Description | Health Points (HP) | Health Regeneration (HP/s) | Speed (units/s) | Resistance (out of 1.0) | Loot Money | Special attributes |
|---|---|---|---|---|---|---|---|
| **Circle Critter** | Average | 100 | 1.0 | 15 | 0.0 | 10 | None |
| **Arrow Critter** | Fast but weak | 30 | 0.0 | 30 | 0.0 | 10 | None |
| **Shuriken Critter** | Strong but slow | 200 | 2.0 | 5 | 0.3 | 10 | None |
| **Square Critter** | Weak but travel in large groups | 50 | 0.0 | 10 | 0.0 | 5 | Travel in groups of 10, 15, 25 |
| **Diamond Critter** | Regenerating critter | 100 | 5.0 | 15 | 0.3 | 15 | None |
| **X Critter** | Critter that is immune to slows and Damage Over Time (DOT) | 100 | 1.0 | 10 | 1.0 | 20 | Resistance of 1.0, so immune to Damage Over Time (DOT) + slow |
| **Infinity Critter** | Strong and Fast (boss) | 200 | 3.0 | 20 | 0.5 | 30 | None |

**GL-Map**

The user must adhere to certain conditions with respect to the creation of the map. The system then validates these rules before allowing the map to be saved.

These conditions are:

1. The grid size can be a minimum of 5x5, and a maximum of 30x30.
2. The path tiles must start on the perimeter, and end on the perimeter
3. There must be no breaks in the path tiles from beginning to end
4. There must be at least one tower tile (although more are recommended)

If these conditions are not met, the map will not be saved (see use case).

## Non-functional Requirements

### Software Quality Requirements

There should be few to no bugs, and the game should not crash upon appropriate use.

### Performance Requirements

The gameplay should be smooth, and without any noticeable Lag**.**

## Design Constraints

The program must be implemented in Java with graphical libraries. The program must be created by the 31st of March.

## Glossary

| Keyword | Description |
|---|---|
| **Build Price** | The cost to build a certain tower, i.e. create and place the tower on the map |
| **Critter** | The entities that travel along the map path in an attempt to make it through without being destroyed by the towers. Critter details can be found in the Game Logic Section. |
| **Critter Money Multiplier** | Determines how much of the loot money the user actually receives when he/she kills a critter. For killing a creep with Loot Money L, in a game mode with CMM C, the player receives money = C*M. For example, in Medium mode, CMM = 1.0, and the player will exactly get the loot money. In easy mode, the player will get more money. In hard mode, the player will get less money. In Zen, the player will not get money. |
| **Damage** | This is the amount of Health Points that an attack from a tower will take away from a critter |
| **Damage Over Time (DOT)** | This is a property of a tower that can harm critters over time, not just upon hitting them. |
| **Game Mode** | There are four game modes: Zen, Easy, Medium, and hard. The user can select which mode he/she would like to play. This changes various game properties (money earned, timers, etc.) that can be found in the Game Logic section |
| **Grid Size** | The dimensions of the map. A grid size of 5x5 means the map is 5 tiles by 5 tiles. |
| **Health Points** | The amount of health that a critter has. If this reaches 0 the critter dies |
| **Health Regeneration** | How much a critter regains health over time |
| **Lag** | When animation appears choppy |
| **Lives** | The amount of critters that the player can allow to reach the end of the path before he/she loses the game |
| **Loot Money** | The amount of money that a critter gives to the player upon dying (before being multiplied by the critter money multiplier). |
| **Map** | What determines where towers can be placed, and on what path the critters will travel |

| | |
|---|---|
| **Map Editor** | The software entity that allows the player to create a new map |
| **Path** | The place where the critters can travel on the map. The critters follow the path from the beginning to the end. |
| **Player** | The primary actor. The player is in control of the game |
| **Range** | The amount of distance units from which the tower can hit critters |
| **Rate of Fire** | The amount of damage instances a tower can do in a set amount of time |
| **Resistance** | A value from 0.0 to 1.0 that determines how a certain critter type is affected by slows/DOT. Reduces the effect by the factor = Resistance. Thus, a value of 1 means that they are not affected at all (effect – 1.0*effect), whereas a value of 0 means that they are completely affected (will not mitigate effect) |
| **Sell Price** | The sell price of a tower is how much money the player will receive if he/she sells the tower. |
| **Slow Factor** | A value from 0.0 to 1.0 that determines how much a towers slows critters. Speed of critter after being hit by tower = (original speed of critter)*(Slow Factor of tower). |
| **Speed** | How many units the critter will move in a set amount of time |
| **Tower Defense Game (TDG)** | The main product for which this Software Requirements Specification has been created |
| **Tower** | The entity that damages or adversely affects critters to stop them from travelling through the path |
| **Upgrade Price** | The amount of money the player has to spend to upgrade a certain tower. There are three available upgrades for each tower, each with its own price |
| **Wave** | One preset (finite) amount of critters that travel on the path periodically. The towers clear these critters, and then the user can manage his/her towers. |

## References

Some aspects of document adapted from:
http://brazos.cs.tcu.edu/0910/deliverables/SRS_v1.0.pdf