TOWER DEFENSE PROJECT: SOFTWARE REQUIREMENTS

Andrei Chubarau 260581375

Yordan Neshev 260587938

Dang Khoa Do 260584925

Steven Voyer 260531264

TABLE OF CONTENTS

1 PROJECT DESCRIPTION

1.1 INTRODUCTION

Tower Defence is a type of real-time strategy video game that has been around since the early 1990's. In the case of this project, the object of the game is to prevent computer controlled enemies from reaching the end of a certain path. To achieve this, the player will strategically place defensive structures or towers which will attempt to eliminate the enemies as these travel along the path. Killing (or damaging) enemies and completing levels and tasks will grant score points to the player which can be used to buy more towers or even upgrade current ones in anticipation of stronger enemies in upcoming levels. Total Scores will be recorded and displayed in the Leaderboards. In addition, the player also has the option to create or edit custom maps. This can be done by using the in-game map manager.

The purpose of this document is to outline exactly how this particular version of Tower Defence will run and the requirements and constraints imposed on the programmers of the game. A full domain model, in-depth use cases and an overview of the business rules associated with this project are also contained within this document.

1.2 CONTEXT

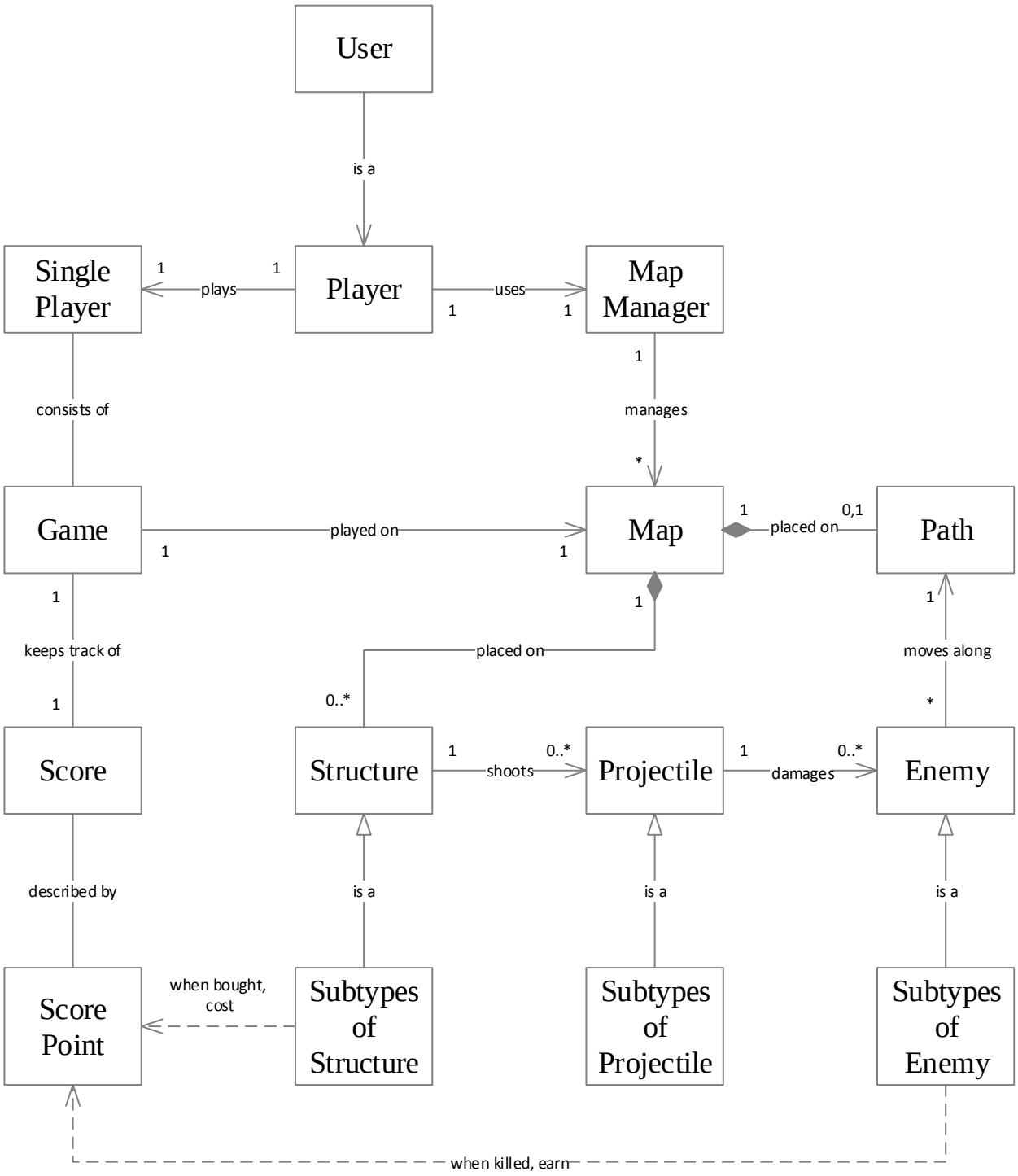
This game is designed for players of all ages and of all video game skill levels. The concept of the game and the navigation of the options are simple enough for any player to pick up and play. In addition, player-chosen difficulty levels enable any player to play the game comfortably.

1.3 BUSINESS GOALS

The goal of this project is to recapture the timeless charm and integrity of Tower Defence games while adding some new features to keep the players on their toes. Among other aspects, the application must be stable, easy to use, enjoyable, and fun. These criteria will be achieved by delivering an optimized and intuitive User-Interface, simple yet addicting gameplay elements, as well as satisfying graphical fidelity.

The game is not created for particular users; it is open for everybody to play and enjoy. The variable difficulty level will help in drawing in players of all ages and skill levels to maximize the user-base.

1.4 DOMAIN MODEL



As can be observed from the domain model, our user is a Player. A Player can play a Game on a Map of his choice and earn a Score. In a Game, Player purchases and places Structures on the Map. Structures act as defenses: they shoot Projectiles that damage Enemies that move along the Path. When an Enemy is killed, Player earns a given amount of Score Points that he can use to purchase Structures.

There are many types of Structures, Projectiles and Enemies. Each type has specific parameters that make it unique.

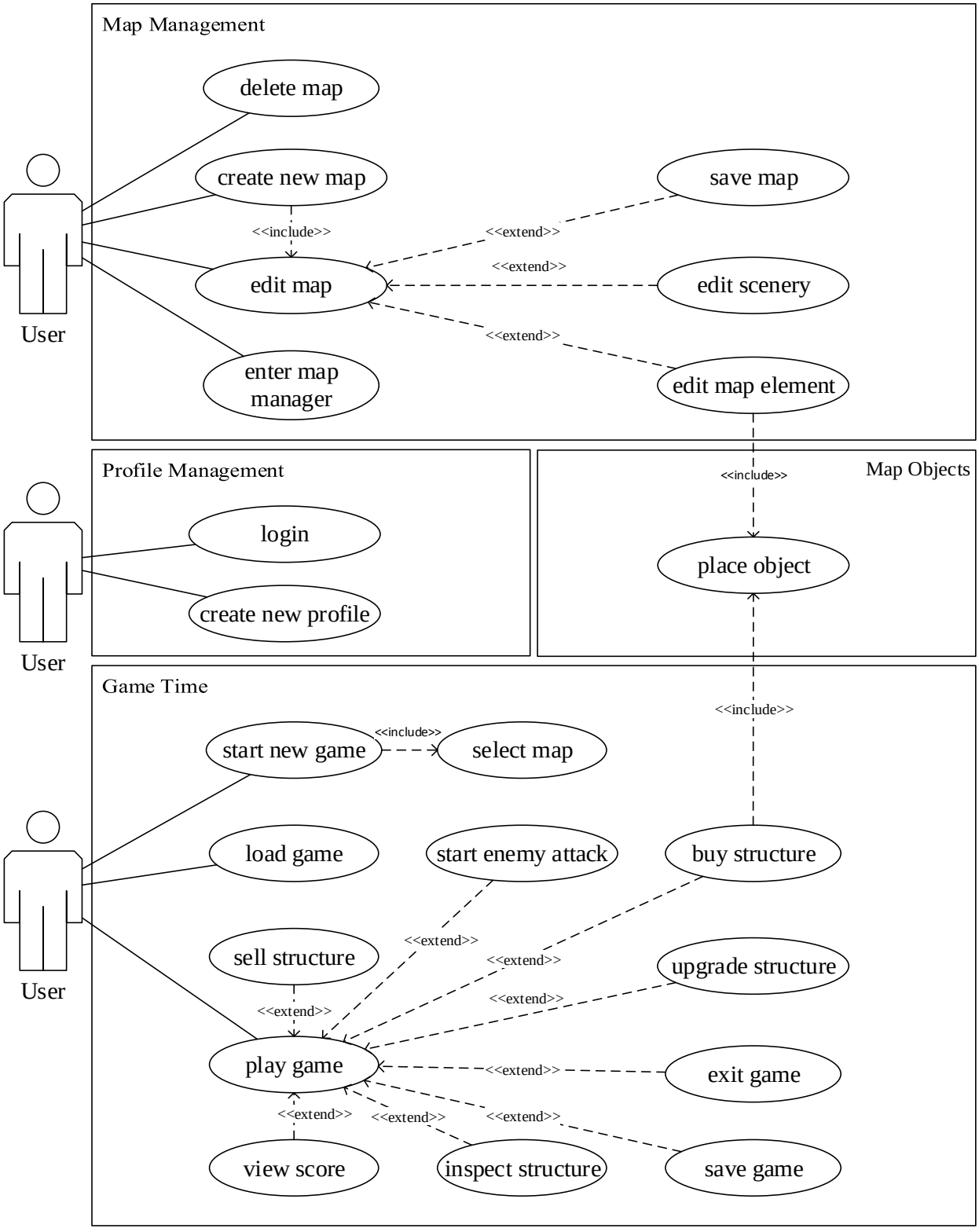
Player can also use Map Manager to manage Maps. Player can create, edit or delete Maps.

1.5 ACTORS

Player is a User who plays the game.

2 FUNCTIONAL REQUIREMENTS

2.1 OVERVIEW



2.2 USE CASES

2.2.1 Use Case: Login

Successful Outcomes: Primary Actor logs in

Use Case Package	Profile Management
ID	UC-PM-01
Use Case Goal	Primary Actor successfully enters Single Player by providing correct Login information
Actors	Primary Actor: Player
Level	User Goal
Precondition	Existing Profile (BR11).
Domain Entities	Player, SinglePlayer

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to enter the game with his existing profile.	
2	System prompts Primary Actor for his username and password.	
3	Primary Actor enters profile credentials	
4	System validates user entries	
5	Primary Actor enters the game	
6	*Use case ends successfully.*	

Alternative Flows:

4a Primary Actor enters incorrect profile credentials

Step	Action	Notes
4a.1	Primary Actor types in a wrong username or password.	
4a.2	System prompts re-try entering his credentials.	
4a.3	Primary Actor confirms.	
4a.4	*Use case resumes at Main Success Scenario step 3.*	

2.2.2 Use Case: Create New Profile

Successful Outcomes: Primary Actor creates a new profile

Use Case Package	Profile Management
ID	UC-PM-02
Use Case Goal	Primary Actor successfully creates new profile by providing profile credentials that are valid for new profile creation
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has launched the game
Domain Entities	Player, SinglePlayer

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to create a new profile and enter the game as a new user.	
2	System prompts Primary Actor to enter new profile's credentials	
3	Primary Actor types new profile credentials	
4	System verifies entered profile credentials for correctness (BR9, BR10).	
5	Primary Actor enters the game with newly created profile	
6	*Use case ends successfully.*	

Alternative Flows:

4a Primary Actor entered an already existing username

Step	Action	Notes
4a.1	System informs Primary Actor that the entered username already exists and suggests creating an account with a different username.	
4a.2	Primary Actor confirms.	
4a.3	*Use case resumes at Main Success Scenario step 2.*	

4b Primary Actor entered an invalid password

Step	Action	Notes
4b.1	System informs Primary Actor that the entered password is invalid and suggests creating an account with a different password.	
4b.2	Primary Actor confirms.	
4b.3	*Use case resumes at Main Success Scenario step 2.*	

2.2.3 Use Case: Enter Map Manager

Successful Outcomes: Primary Actor successfully enters Map Manager

Use Case Package	Map Management
ID	UC-MM-01
Use Case Goal	Primary actor enters Map Manager
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has launched the game
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to enter Map Manager	
2	System prompts Primary Actor for confirmation	
3	Primary Actor confirms	
4	System launches Map Manager	
5	*Use case end successfully*	

2.2.4 Use Case: Create new map

Successful Outcomes: Primary Actor successfully creates new map

Use Case Package	Map Management
ID	UC-MM-02
Use Case Goal	Primary actor creates new Map by specifying valid map parameters
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor performed Enter Map Manager (UC-MM-01).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention of creating a new Map	
2	System prompts Primary Actor to enter map properties.	
3	Primary actor enters map properties and confirms (BR7).	
4	System initializes and saves new map according to entered properties.	
5	*Use case ends successfully.*	

Alternative Flows:

3a The chosen map properties are invalid

Step	Action	Notes
3a.1	The map properties chosen are invalid.	
3a.2	System informs Primary Actor of the error and prompts re-try.	
3a.3	*Use case resumes at Main Success Scenario step 2.*	

2.2.5 Use Case: Edit map

Successful Outcomes: Primary Actor successfully edits parameters of a map

Use Case Package	Map Management
ID	UC-MM-03
Use Case Goal	Primary Actor edits parameters of a map by specifying new map parameters that are valid
Actors	Primary Actor: Player
Level	Summary Goal
Precondition	Primary Actor performed Enter Map Manager (UC-MM-01).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to edit an existing map.	
2	System prompts Primary Actor to select an existing map to edit.	
3	Primary Actor selects existing map and confirms.	
4	System loads selected map and presents Primary Actor with editing options.	Extended functionality is available after this step (more specific map editing options)
5	*Use case ends successfully.*	

Alternative Flows:

2 System cannot locate any existing maps valid for editing.

Step	Action	Notes
2.1	System informs Primary Actor that no maps are available for editing.	
2.2	*Use case ends unsuccessfully.*	

2.2.6 Use Case: Edit scenery

<<extends>> Edit map @ Step 4

Successful Outcomes:

Use Case Package	Map Management
ID	UC-MM-04
Use Case Goal	Primary Actor edits scenery of Map by specifying valid scenery parameters
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor performed Enter Map Manager (UC-MM-01). Primary Actor is currently editing a map (UC-MM-03).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to edit the scenery of an existing map.	
2	System presents Primary Actor with scenery editing options.	
3	Primary Actor edits scenery parameters of the map.	
4	*Use case ends successfully.*	

Alternative Flows:

3a Primary Actor enters invalid scenery parameters

Step	Action	Notes
3a.1	System resets scenery parameters	
3a.2	System notifies Primary Actor of failure to edit scenery	
3a.3	*Use case ends unsuccessfully*	

2.2.7 Use Case: Edit scenery elements

<<extends>> Edit map @ Step 4 <<includes>> Place Object

Successful Outcomes: Primary Actor successfully edits scenery elements

Use Case Package	Map Management
ID	UC-MM-05
Use Case Goal	Primary Actor edits scenery elements by specifying valid scenery elements parameters
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor performed Enter Map Manager (UC-MM-01). Primary Actor is currently editing a map (UC-MM-03).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to edit scenery elements	
2	System presents Primary Actor with scenery element editing options	Options: edit scenery, add scenery element, move scenery element, edit path
3	Primary Actor edits scenery	
4	System validates and applies new scenery parameters	
5	*Use case ends successfully.*	

Alternative Flows:

3a Primary Actor desires to move existing scenery element

Step	Action	Notes
3a.1	Primary Actor selects existing scenery element which he desires to move	
3a.2	Primary Actor performs Place Object (UC-OI-01)	
3a.3	*Use case ends successfully*	

3aa Place Object is performed unsuccessfully

Step	Action	Notes
3aa.1	System resets position of scenery element	
3aa.2	System notifies Primary Actor of failure to place scenery object	
3aa.3	*Use case ends unsuccessfully*	

3b Primary Actor desires to add a new scenery element

Step	Action	Notes
3b.1	Primary Actor indicates intention to add a new scenery element	
3b.2	System presents Primary Actor with available scenery elements	
3b.3	Primary Actor selects a scenery element	
3b.4	Primary Actor performs Place Object (UC-OI-01)	
3b.5	*Use case ends successfully*	

3bbPlace Object is performed unsuccessfully

Step	Action	Notes
3aa.1	System resets position of scenery element	
3aa.2	System notifies Primary Actor of failure to place scenery object	
3aa.3	*Use case ends unsuccessfully*	

4a System fails to validate new scenery parameters

Step	Action	Notes
4a.1	System resets scenery parameters	
4a.2	System notifies Primary Actor of failure to edit scenery	
4a.3	*Use case ends unsuccessfully*	

2.2.8 Use Case: Edit path

<<extends>>Edit scenery elements @ Step 2

Successful Outcomes: Primary Actor edits parameters of path

Use Case Package	Map Management
ID	UC-MM-06
Use Case Goal	Primary Actor edits path by specifying valid path parameters
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using Map Manager (UC-MM-01). Primary Actor is currently editing a map (UC-MM-03). Primary Actor desires to edit a Path element during Edit Scenery Elements (UC-MM-04)
Domain Entities	Player, Map Manager, Map, Path

Main Success Scenario:

Step	Action	Notes
1	System presents Primary Actor with path element editing options.	
2	Primary Actor edits path parameters.	
3	System validates new path parameters (BR6)	
4	*Use case ends successfully.*	

Alternative Flows

4a System fails to validate new path parameters

Step	Action	Notes
4a.1	System resets path parameters	
4a.2	System notifies Primary Actor of failure to edit path	
4a.3	*Use case ends unsuccessfully*	

2.2.9 Use Case: Save map

<<extends>> Edit map@ Step 5

Successful Outcomes: Primary Actor successfully saves currently open map

Use Case Package	Map Management
ID	UC-MM-07
Use Case Goal	Primary Actor saves currently open map
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using Map Manager (UC-MM-01). Primary Actor is currently editing a map (UC-MM-03).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to save currently open map	
2	System prompts Primary Actor to enter a map name	
3	Primary Actor enters map name and confirms	
4	System validates map parameters and entered map name (BR7)	
5	*Use case ends successfully.*	

Alternative Flows

4a System fails to validate map parameters

Step	Action	Notes
4a.2	System notifies Primary Actor of failure to save map	
4a.3	*Use case ends unsuccessfully*	

2.2.10 Use Case: Delete map

<<includes>> Enter MapManager

Successful Outcomes: Primary Actor successfully deletes an existing map

Use Case Package	Map Management
ID	UC-MM-08
Use Case Goal	Primary Actor deletes an existing map by selecting an existing map and confirming selection
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using Map Manager (UC-MM-01).
Domain Entities	Player, Map Manager, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to delete an existing map.	
2	System prompts Primary Actor to select a map from existing maps	
3	Primary Actor selects map to delete	
4	System prompts Primary Actor for confirmation	
5	Primary Actor confirms	
6	System deletes selected map	
7	*Use case ends successfully*.	

Alternative Flows:

4a Primary Actor cancels the deletion of map

Step	Action	Notes
4a.1	Primary Actor indicates intention to cancel the deletion of map	
4a.2	System aborts the deletion of map	
4a.3	*Use case ends unsuccessfully*	

2.2.11 Use Case: Start new game

<<includes>> Select map

Successful Outcomes: Primary Actor successfully starts a new game

Use Case Package	Game Time
ID	UC-GT-01
Use Case Goal	Primary Actor starts a new game by selecting a map
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has performed Login (UC-PM-01).
Domain Entities	Player, SinglePlayer, Game, Map

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to start a new game.	
2	System prompts Primary Actor to select map.	
3	Primary Actor performs Select map (UC-GT-02)	
4	System responds by presenting the properties of the new Game and prompts Primary Actor for confirmation.	
5	Primary Actor confirms.	
6	System initializes a new game.	
7	*Use case ends successfully.*	

Alternative Flows:

3a Select Map ends unsuccessfully.

Step	Action	Notes
3a.1	System notifies Primary Actor that a new game could not be started.	
3a.2	Primary Actor confirms.	
3a.3	*Use case ends unsuccessfully.*	

4b Primary Actor decides to re-edit game parameters

Step	Action	Notes
4a.1	Primary Actor does not confirm game selection	
4a.3	*Use case resumes at Step 2*	

2.2.12 Use Case: Select map

Successful Outcomes: Primary Actor selects map to play on

Use Case Package	Game Time
ID	UC-GT-02
Use Case Goal	Primary Actor successfully selects map
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has performed login (UC-PM-01).
Domain Entities	Player, SinglePlayer, Game

Main Success Scenario:

Step	Action	Notes
1	System notifies Primary Actor of available maps.	
2	Primary Actor indicates intention to select a specific Map.	
3	System responds by presenting the properties of the selected Map and prompts Primary Actor for confirmation.	
4	Primary Actor confirms.	
5	*Use case ends successfully.*	

Alternative Flows:

1a System cannot locate any existing maps.

Step	Action	Notes
1a.1	System notifies Primary Actor that no maps exist.	
1a.2	Primary Actor confirms.	
1a.3	*Use case ends unsuccessfully.*	

2.2.13 Use Case: Load game

Successful Outcomes: Primary Actor loads a saved game

Use Case Package	Game Time
ID	UC-GT-03
Use Case Goal	Primary Actor successfully loads a game
Actors	Primary Actor: Player
Level	Summary Goal
Precondition	Primary Actor has performed login (UC-PM-01).Player has saved games (BR 8).
Domain Entities	Player, SinglePlayer, Map, Game

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to load a game.	
2	System prompts Primary Actor to select a game to load from a list.	
3	Primary Actor selects and confirms.	
4	System loads map and starts up a Game.	
5	*Use case ends successfully.*	

Alternative Flows:

2a. No saved games exist

Step	Action	Notes
2a.1	System notifies Primary Actor that no saves games exist.	
2a.2	Primary Actor confirms.	
2a.3	*Use case ends unsuccessfully.*	

2.2.14 Use Case: Play game

Successful Outcomes: Primary Actor plays a game

Use Case Package	Game Time
ID	UC-GT-04
Use Case Goal	Primary Actor successfully plays a game
Actors	Primary Actor: Player
Level	Summary Goal
Precondition	Primary Actor has performed login (UC-PM-01). Primary Actor has performed Load Game (UC-GT-03) or Start New Game (UC-GT-01).
Domain Entities	Player, SinglePlayer, Map, Game

Main Success Scenario:

Step	Action	Notes
1	System provides gameplay options for Primary Actor to select.	Gameplay options consists of other Game Time use cases
2	Primary Actor performs game play options	
3	Primary Actor performs Exit Game (UC-GT-09)	
4	*Use case ends successfully.*	

2.2.15 Use Case: Start enemy attack

<<extends>> Play game @ Step 2

Successful Outcomes: Primary Actor starts an enemy attack

Use Case Package	Game Time
ID	UC-GT-04
Use Case Goal	Primary Actor successfully starts an enemy attack
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using SinglePlayer (UC-GT-01). Primary Actor is playing a game (UC-GT-03).
Domain Entities	Player, SinglePlayer, Game, Enemy

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to start enemy attack.	
2	System verifies if map contains defensive Structures.	
3	System starts an enemy attack.	
4	*Use case ends successfully.*	

Alternative Flows:

2a. System discovers that no defensive structures exist on map

Step	Action	Notes
2a.1	System informs Primary Actor that there are no defensive structure placed.	
2a.2	Primary Actor confirms.	
2a.3	System aborts enemy attack.	
2a.4	*Use case ends unsuccessfully.*	

2.2.16 Use Case: Sell structure

<<extends>> Play game @ Step 2

Successful Outcomes: Primary Actor sells an existing structure

Use Case Package	Game Time
ID	UC-GT-06
Use Case Goal	Primary Actor successfully sells an existing Structure
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using SinglePlayer (UC-GT-01). Primary Actor is playing a game (UC-GT-03). Structure exists in Game (BR2).
Domain Entities	Player, SinglePlayer, Game, Structure

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention of selling a specific structure.	
2	System prompts Primary Actor to confirm the sale.	
3	Primary Actor confirms.	
4	System removes the structure and updates user's resources with its resell value.	
5	*Use case ends successfully.*	

2.2.17 Use Case: Buy structure

<<extends>> Play game @ Step 2 <<includes>> Place Object

Successful Outcomes: Primary Actor buys a structure

Use Case Package	Game Time
ID	UC-GT-06
Use Case Goal	Primary Actor successfully buys a Structure
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using SinglePlayer (UC-GT-01). Primary Actor is playing a game (UC-GT-03). Player has enough Score Points (BR3).
Domain Entities	Player, SinglePlayer, Game, Structure

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention of buying a structure.	
2	System verifies the available resources of the Primary Actor. Then prompts the Primary Actor to confirm.	
3	Primary Actor confirms.	
4	Primary Actor performs Place Object (UC-GT-07)	
	Use case ends successfully.	

Alternative Flows:

2a. Not Enough Available Resources

Step	Action	Notes
2a.1	System informs Primary Actor that there is not enough available resources.	
2a.2	Primary Actor aborts.	
2a.3	*Use case ends unsuccessfully.*	

2.2.18 Use Case: Place object

Successful Outcomes: Primary Actor successfully places a object

Use Case Package	Object Interaction
ID	UC-OI-01
Use Case Goal	Primary Actor places an object
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is currently performing Play Game (UC-GT-03) or Edit Map (UC-MM-03).
Domain Entities	Player, SinglePlayer, Game, Structure

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates where to put the Object on the Map.	
2	System verifies if the selected area if appropriate.	
3	System places the Structure at the area specified by the Primary Actor.	
4	*Use case ends successfully.*	

Alternative Flows:

2a. Selected Area is Invalid

Step	Action	Notes
2a.1	System informs Primary Actor that the selected area is invalid and prompts re-try.	
2a.2	*Use case resumes at Main Success Scenario step 1.*	

2a. Selected Area is Invalid

Step	Action	Notes
2a.1	System informs Primary Actor that the selected area is invalid and prompts re-try.	
2a.2	Primary Actor aborts.	
2b.3	*Use case ends unsuccessfully.*	

2.2.19 Use Case: Exit game

<<extends>> Play Game

Successful Outcomes: Primary Actor exits Game

Use Case Package	Game Time
ID	UC-GT-09
Use Case Goal	Primary Actor successfully exits the Game
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor is using SinglePlayer (UC-GT-01). Primary Actor is playing a game (UC-GT-03).
Domain Entities	Player, SinglePlayer, Game

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention of exiting the Game.	
2	System prompts the Primary Actor for confirmation	
3	Primary Actor confirms.	
4	System sends Primary Actor to the main menu.	
5	*Use case ends successfully.*	

2.2.20 Use Case: Save game

Successful Outcomes: Primary Actor saves a Game

Use Case Package	Game Time
ID	UC-GT-10
Use Case Goal	Primary Actor successfully saves a Game by specifying valid saved game parameters
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has loaded a game (UC-GT-01). Primary Actor is playing a game (UC-GT-03).
Domain Entities	Player, SinglePlayer, Game

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to save the current Game.	
2	System prompts Primary Actor to enter saved game parameters and to confirm.	
3	Primary Actor enters saved map parameters confirms.	
4	System validates entered saved game parameters and saves the game	
5	*Use case ends successfully.*	

Alternative Flows:

2a Primary Actor enters invalid saved game parameters

Step	Action	Notes
2a.2	System informs Primary Actor of the error and prompts re-try.	
2a.3	*Use case resumes at Main Success Scenario step 2.*	

2.2.21 Use Case: View score

<<extends>> Play Game

Successful Outcomes: Primary Actor views Score

Use Case Package	Game Time
ID	UC-GT-12
Use Case Goal	Primary Actor successfully view score
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has loaded a game (UC-GT-01). Primary Actor is playing a game (UC-GT-03).
Domain Entities	Player, SinglePlayer, Score, Score Point

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to view Score.	
2	System responds by presenting a list of Scores.	
3	Primary Actor indicates intention to stop viewing Scores	
4	System resumes game	
5	*Use case ends successfully.*	

2.2.22 Use Case: Inspect Structure

Successful Outcomes: Primary Actor inspects a Structure

Use Case Package	Game Time
ID	UC-GT-13
Use Case Goal	Primary Actor successfully inspects a Structure
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has loaded a game (UC-GT-01). Primary Actor is playing a game (UC-GT-03). Primary Actor has previously performed Buy Structure (UC-GT-07).
Domain Entities	Player, SinglePlayer, Game, Structure

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to inspect a specific Structure.	
2	System responds by presenting the properties of the selected Structure.	
3	*Use case ends successfully.*	

Alternative Flows:

3a Primary Actor Inspects Another Structure

Step	Action	Notes
3a.1	Primary Actor wished to continue inspecting Structures.	
3a.2	*Use case restarts at step 1.*	

2.2.23 Use Case: Upgrade Structure

Successful Outcomes: Primary Actor inspects a Structure

Use Case Package	Game Time
ID	UC-GT-14
Use Case Goal	Primary Actor successfully upgrades a Structure
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has loaded a game (UC-GT-01). Primary Actor is playing a game (UC-GT-03). Primary Actor has previously performed Buy Structure (UC-GT-07).
Domain Entities	Player, SinglePlayer, Game, Structure

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to upgrade a specific Structure.	
2	System verifies the available resources of the Primary Actor (BR3) and prompts the Primary Actor to confirm.	
3	Primary Actor confirms.	
4	*Use case ends successfully.*	

Alternative Flows:

2a. Not Enough Available Resources

Step	Action	Notes
2a.1	System informs Primary Actor that there is not enough available resources.	
2a.2	Primary Actor confirms	
2a.3	*Use case ends unsuccessfully.*	

2.2.24 Use Case: Inspect Enemy

Successful Outcomes: Primary Actor inspects a Structure

Use Case Package	Game Time
ID	UC-GT-15
Use Case Goal	Primary Actor successfully inspects a Structure
Actors	Primary Actor: Player
Level	User Goal
Precondition	Primary Actor has loaded a game (UC-GT-01). Primary Actor is playing a game (UC-GT-03). Primary Actor has started an enemy attack (UC-GT-04).
Domain Entities	Player, SinglePlayer, Game, Enemy

Main Success Scenario:

Step	Action	Notes
1	Primary Actor indicates intention to inspect a specific Enemy.	
2	System responds by presenting the properties of the selected Enemy.	
3	*Use case ends successfully.*	

Alternative Flows:

3a Primary Actor Inspects Another Structure

Step	Action	Notes
3a.1	Primary Actor wished to continue inspecting Enemies.	
3a.2	*Use case restarts at step 1.*	

2.3 USER STORY CONVERSION

2.4 BUSINESS RULES

Number	Label	Business Rule	Notes
BR2	Structure exists in Game	Before starting an attack, the System will check if there are any Structure in Game, so that the Primary Actor can defend itself. Also, if the Player wants to sell a Structure, there should be an existing one in Game.	
BR3	Player has enough Score Points	In order to buy or upgrade a Structure, an amount of Score Points are required. Once the Structure has been successfully bought or upgraded, the required Score Points will be deducted.	
BR4	Valid selected area for Structure	Structures must be placed in an area on the Map that is not part of the Path or not already occupied by other Structures.	
BR5	Valid Game Name	For the name of the saved Game to be valid, it must be between 1 and 30 characters, and only contain numbers ['0'...'9'], lowercase ['a'...'z'] and upper case letters ['A'...'Z'], dashes['-'], and underscores ['_']. The name must also be unique.	
BR6	Valid Path	The path created on the map has to be continuous and contain no loop. It must have a starting and an ending point.	
BR7	Valid Map	A valid map name must be between 1 and 30 characters, and only contain numbers ['0'...'9'], lowercase ['a'...'z'] and upper case letters ['A'...'Z'], dashes ['-'], and underscores ['_']. The size of the map must be [10-30]x[10-30]. In the presence of a path, path must be valid.	
BR8	Saved Game	To load a Game, there must be an existing saved game.	
BR9	Valid Username	For the username to be valid, it must be unoccupied by another user.	
BR10	Valid Password	For the password of the user to be valid, it must be between 5 and 15 characters, and only contain numbers [‘0’...’9’], lowercase [‘a’...’z’] and upper case letters [‘A’...’Z’], dashes[‘-‘], and underscores [‘_’].	
BR11	Existing Profile	In order to login a profile, there must be an existing one.	

3 NON-FUNCTIONAL REQUIREMENTS

With regards to the non-functional requirements, it is essential that the game is user friendly. This includes menus and options that are easily navigable and that are simple to understand. The game must look nice and presentable and in-game options must be organized in a way which will not confuse the user.

In addition, the game will be in English. It is thus imperative to avoid any spelling or grammar mistakes in the text of the game.

The game must also be visually appealing which will be achieved through use of colorful and original art.

4 DESIGN CONSTRAINTS

In terms of constraints, the first and most important thing to note is that this game must be completed by the first week of April. It is essential that this time constraint be respected.

It is also required that this game be programmed in JAVA and that all in-game sprites be designed using the Swing library and public-use images. While little intricate features of the game are open to programmer imagination, the integrity of the original Tower Defence game must be maintained. The following features must be present:

- Main menu and Leaderboards complete with a user login feature.
- A functional map editor which can differentiate between valid and non-valid maps. The path and the scenery must be separate (path must have one entry and one exit point)
- Currency system to buy different types of towers with different attributes (Status ailment, range, strength, projectile type etc…)
- Enemies (Wave-based) that will attack the player. There will be several types of enemies each with their own unique attributes (hit points, speed, kill worth etc…)
- Towers can only be put on the scenery and enemies can only travel on the path.

NOTE: while the above features are currently deemed to be necessary, the final design can also include additional features or iterate on the ones hereby described.

5 GLOSSARY

5.1 BUSINESS ENTITIES

Business Entity	Description
Player	User of the application able to play games and create or edit maps.
Map Manager	System that allows Player to create new or edit existing maps.
Singleplayer	System that allows Player to play on existing maps.
Game	Description of the progress of Player playing Singleplayer.
Map	Visual and logical description of the game world on which Player can play on.
Path	Sequence of tiles on Map that Enemies are allowed to move along.
Score	Total score achieved by Player while playing Game.
Score Point	In-game currency that can be earned by eliminating Enemies and spent on purchasing Structures.
Structure/Tower	Purchasable defenses placed on Map by Player during Game. Structures shoot Projectiles at Enemies.
Projectile	In-game objects that damage and eliminate Enemies thus earning Score Points to add to Player's Score.
Enemy	Physical objects that move along Path on Game's Map and can damage Player's base upon reaching it.

6 REFERENCES

Daniel Sinnig PhD Lecture Slides, ECSE-321. McGill University Winter 2015

Martin Fowler UML Distilled: A Brief Guide to the Standard Object Modelling Language.