CS 319 - Object-Oriented Software Engineering

Analysis Report

Medieval Tower Defense

Group 1A

Abdullah Mahir Özer

Deniz Başaran

H.Buğra Aydın

Berk Ataç

1. Introduction

Medieval Tower Defense is a basic tower defense type video game we decided to develop. There are lots of different tower defense games with different objectives. The main purpose of tower defense games is to prevent enemies from reaching a certain point or destroying a base at that point, by marching along a predetermined path. In order to achieve this, the player sets up a defense of their own design by using the tools provided to them by the system. This is usually done by building towers with resources, customizing them in a way of their choosing according to their strategy.

The game we were influenced by is following game mods from the game Warcraft III.

https://www.epicwar.com/maps/159097/

http://www.moddb.com/mods/element-tower-defense/news/element-td-40-released

Medieval Tower Defense (MTD) will be developed with different features. In MTD we will add new types of waves like boss fights and various difficulty settings.

Classic tower defense game rules will apply. The player will face fifty waves of enemies with the defense they built and prevent them from reaching to the exit point. The player will be able to face the next wave after cleaning wave that is coming. The game is designed to be reusable, since the players will want to defeat the game and obtain a spot in the high score list.

The game will be a desktop application and it will be controlled by a mouse.

2. Overview

MTD is a medieval themed tower defense game. All tools, waves and bosses will be from medieval times. Player will face a wave with towers built by spending resources provided to them at start. After killing each enemy, player will gain different amount of resources and player can choose to build more towers or upgrade towers they have already built. Goal of MTD is clearing all waves, killing bosses to face next wave and next boss until waves are finished.

2.1 Gameplay

Player will need a mouse to play the game. Keyboard will be available to perform some actions such as entering a nickname for high score screen when the game ends. Player will choose and place tower to the map with mouse. Towers can be placed on top of the areas which are made available to player. These areas will usually be near the path which enemies are walking on. After given time finishes, waves will start to come and player's towers will kill them or fail to kill them and enemies will pass exit. There are 50 waves. Wave difficulty will change with difficulty settings. Player is allowed to miss a certain amount of enemies but if that amount is reached, player will lose and start again from the beginning. Players will be able to customize tower according to their strategies and type of enemies.

Game mods are as follows.

- Normal mod : 50 waves, 5 bosses
- Survival mod : Infinite waves, infinite bosses

2.2 Waves

Game will progress one wave at a time. Meaning second wave will not start before all enemies from first wave are killed or missed. A brief waiting time will be given to player between waves. Player will be able to fast forward the game. A player who has established a fine defense which can hold on for several waves can use fast forward so that waiting time for those waves are shorter.

2.2.1 Boss waves

Bosses will come from one of two lanes in boss waves. Boss waves will be more challenging since they will require unique strategies to defeat.

2.3 Building and Customizing Towers

Player's defense will consist of towers they build. The player will place towers on areas they are allowed to and they will be able to customize them with game progress and resource gain.

2.3.1 Tower Types

There will be different tower types and each type will have their own specifications. Tower specs are as follows.

- Slow Rating
- Range
- Damage (siege, infantry)
- Fire Rate

Each tower type will specialize in different ways. Tower types are a follows.

Tower name	**Range/Attack Frequency**	**Attack Point/Damage Type**
Arrow Tower	Medium/High	Low/single target
Canon Tower	High/Low	Medium/splash damage
Ice Tower	Medium/Medium	low/slow debuff
Fire Tower	Low/Low	Medium/bonus against siege engines
Oil Tower	Very Low/Low	High/area damage
Poison Tower	Medium/High	Medium/Victim bleeds
Arcane Tower	High/Medium	Very low/Armor reducement

Ballista Tower	Very High/very low	Very High/single target
Mortar Tower	High/low	Medium/single target

2.4 Enemies

Each wave will have types of enemies with their own specs.

- Speed
- Armor
- Health

2.4.1 Enemy Types

Invader Name	**Speed**	**Health**	**Armour**
Footman	Medium	Low	Low
Catapult	Low	Medium	Medium
Knight	Low	Medium	High
Light Cavalry	High	Low	Low
Elephant Rider	Low	High	Medium
Trojan Horse	Low	Very High	High
Chengiz Khan & riders	High	Very High	medium
Battering Ram	Low	High	Very High
Jester	Very High	Medium	Medium
Saint John’s Knigths	Low	High	Very High

Pope	Medium	Very High	Low

3. Requirement Specificaton

3.1 Functional Requirements

3.1.1 Play Game

The game will start after play game button is pushed. The isometric game map that includes a path for AI invaders and free sites to build towers will be loaded. The AI invaders will be on the path to the final location and this final location is the keep that player has to protect with the towers built. In addition, any invader who manages to trespass the keep will cost the player a health point and player will have 50 health points at total. There will be various types of towers with different characteristics. Towers can be upgraded for better attack points, and better crowd control ability, also new towers can be purchased with sufficient amount of coin, which can be gathered by eliminating invaders with existing towers. Since Medieval-Tower-Defense is a wave-based game, there will be a number of 50 waves. At each 7 or 8 level player will face with a boss invader, which is considerably harder to beat, yet generates a lot of coins when defeated.

There will also be different types of invaders that have varying attributes of speed, health, and armor. There is another list available that shows the invader types. First 5 are minions and remainder 7 are boss units, which are more challenging special units.

3.1.2 Settings

Gameplay and music sounds are the features which can be set by the player. In each time when the game application is started, volume of the gameplay and music sounds, and visual quality are predefined in default settings. In order to change these default settings, user

may enter the settings page from the main menu and adjust any two of the sound types with the two separate slide bars as desired.

3.1.3 High Scores

Through High Scores bar user will be able to see the list of the highest scores, yet chart is accessible only if the user has internet connection. User's own score and other users with top scores will be visible in High Scores section. In this case, there will be a more competitive environment which means more game activity. The high score list will be implemented using a database. It is required to update and store this list.

3.1.4 Pause Button

The user can pause the game in progress any time desired by pressing pause button. Game time will be frozen as long as the pause button is pressed again. Button will be located at the top left corner during gameplay. If the player turns back to the main, game will be auto-paused, and player will have to click continue game in order to proceed more in the current game session. User will lose the achieved progress if application's closed during pause stance.

3.1.5 Help and Information

This section serves as a manual which represents the overall game concepts. The user can get information about: map pattern, game purpose, health, resource gathering, invader types/attributes, and tower types/attributes. Help document is accessed through the main menu, or anytime gameplay is in progress. There may also be a subsection in which game tips can be accessed.

Medieval Tower Defense game will have historical figures such as Chengiz Khan and St. John's Knights in it. Player's will be able to read information about historical characters and technologies from the information screen(tech tree).

3.1.6 Credits

The User may visualize the list of developers by pushing the credits bar of the main menu. Contact information for communicating the developers will be available as well. Suggestions, comments, and bug reports may be sent directly to the developers with information available in this section.

3.1.7 Exit

Exit button is pretty self-explanatory. The application will be closed completely when the button's pushed. Paused game session will be lost and settings will be set on the default mode in the next game launch of application.

3.2 Non-Functional Requirements

• Game graphics will be designed in order to be appealing. There will be no flickering and movement of the units and the projectiles will be smooth. Images of graphical objects will be chosen to give the players better gameplay experience.

• Control mechanisms of the game will have low response time, which enables users to play with minimal delay.

• The code will be efficient, allowing for no framerate drops during gameplay.

• The game will be readable, the player should understand what to do as long as they enter the game.

• The game art should not be too confusing to make it easier for the player to easily see what's going on on the map.

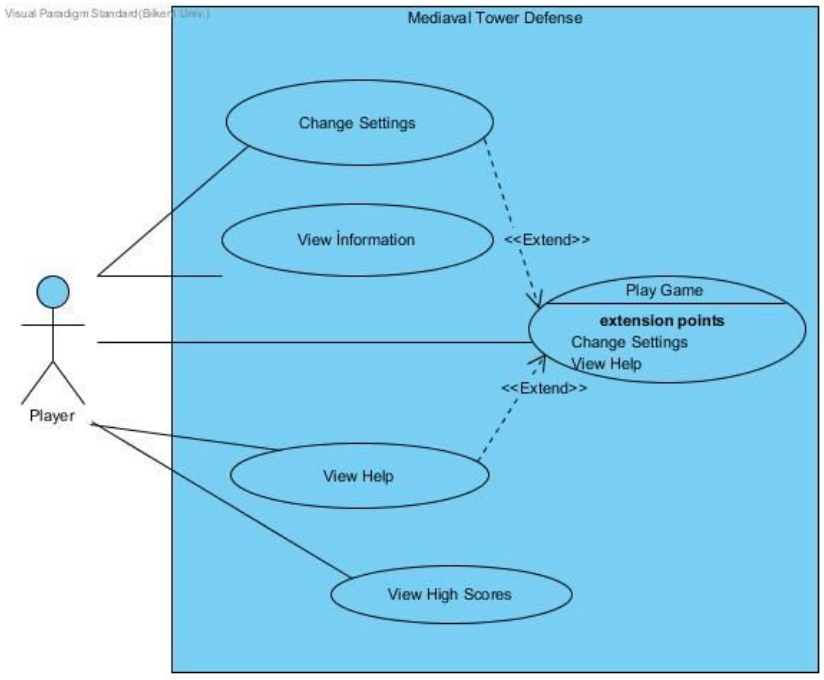
3.3 Pseudo Functional Requirements

• The game will be implemented in Java.

• Game art will be done using Adobe® Photoshop CC, Balzamiq.

4. System Model

This section provides information about the main use case model of MTD game, detailed use case explanations are included below.



4.1 Use Case Model

4.1.1 Play Game

Use case 1: Play Game

Actors involved: Player

Aim and System Response:

-Player aims to choose and place towers efficiently to not lose any lives. Losing a life means letting an enemy through, therefore also losing gold and reducing the chance of win.

-System keeps the score of the Player.

Entry condition: Player has opened the game and selected Play Game from the main menu.

Exit condition:

1. Player has won the game with or without achieving a high score, OR

2. Player has lost, i.e., letting through more enemies than his/her lives, OR

3. Player has exited the game from the in-game menu.

Preconditions: Player defined settings are used when starting a game if the player hasn't visited & changed the settings tab from the main menu, pre-defined settings are used when Play Game is selected.

Post-conditions: If the player's score is high enough to be on the leaderboard, it is added to the highest scores list.

Event Flow:

1. The player starts the game from the first level with the default life count.

2. Player constructs his/her tower system in the given 30 second time before each wave appears. The player can also add towers when the wave is spawn.

Flow A:

3A. Player successfully finishes all the levels, i.e., not letting through more enemies than his given lives.

4A. The player's score is displayed on screen. If the player is among the top ten scorers, he enters his name to be added to the high scores list.

5A. Player returns to the main menu.

Flow B:

3B. Player lets through more enemies than his lives.

4B. The player's score is displayed on screen. If the player is among the top ten scorers, he enters his name to be added to the high scores list.

5B. Player returns to the main menu.

Flow C:

3C. Player pauses the game while playing, accessing the in-game menu.

4C. Player decides to quit from the in-game menu before finishing the game.

Flow D:

3D. Player decides to continue playing.(Back to step 2)

4.1.2 Change Settings

Use Case 2: Change Settings

Actors involved: Player

Aim and System Response:

-Players want to change the settings such as difficulty, sound volume, etc.

Pre-condition: Default game settings are given by the system each time the game is started.

Post-condition: Game settings are changed by the player.

Entry Condition: Player selects the "Settings" button from the main menu or from the in-game menu.

Exit Condition: Player selects "Back" to return to the menu

Event Flow:

1. Player clicks on the "Settings" button from the menu.

2. Game settings are displayed with interactable buttons.

Flow 1:

3A. Player changes settings.

4A. Player returns to the menu by pressing "Back." The changes are saved.

Flow 2:

3B. Player returns to the menu by pressing "Back." No new settings have been saved.

4.1.3 View Help

Use Case 3: View Help

Primary Actor: Player

Stakeholders and Interests:

- Player needs help in how to play the game.

- Player is given explanations on how certain aspects of the game work, the information is given in text format by the system.

Entry Condition: Player selects "View Help" from Main Menu or in-game Menu.

Exit Condition: Player selects "Back" to return to either main menu or the in-game menu.

Pre-conditions: Player is in the main menu or in-game menu.

Post-condition: -

Event Flow:

1. Player click on “Help” on the menu.

2. A text explaining the controls and the aim of the game is shown.

4.1.4 View Credits

Use Case 4: View Credits

Primary Actor: Player

Aim and System Response:

- Player aims to get information about the developers.

- Name and contact information of the developers is displayed.

Pre-conditions: Player should be in the Main Menu .

Post-condition: -

Entry Condition: Player hits the “Credits” button from main menu.

Exit Condition: Player hits the “Back” button to return back to the main menu.

Event Flow:

1.Player clicks on the “Credits” button from the menu.

2.Information regarding the developers is displayed.

3.Player returns back to the menu by hitting the “Back” button.

4.1.5 View High Scores

Use Case 5: View High Scores

Primary Actor: Player

Aims and System Response:

-Player aims to see the leaderboard.

-The system shows the top ten scorers with their saved nicknames.

Pre-conditions: System should save the high scorers after every game instance is ended.

Post-condition: -

Entry Condition: Player hits the “High Scores” button from the main menu.

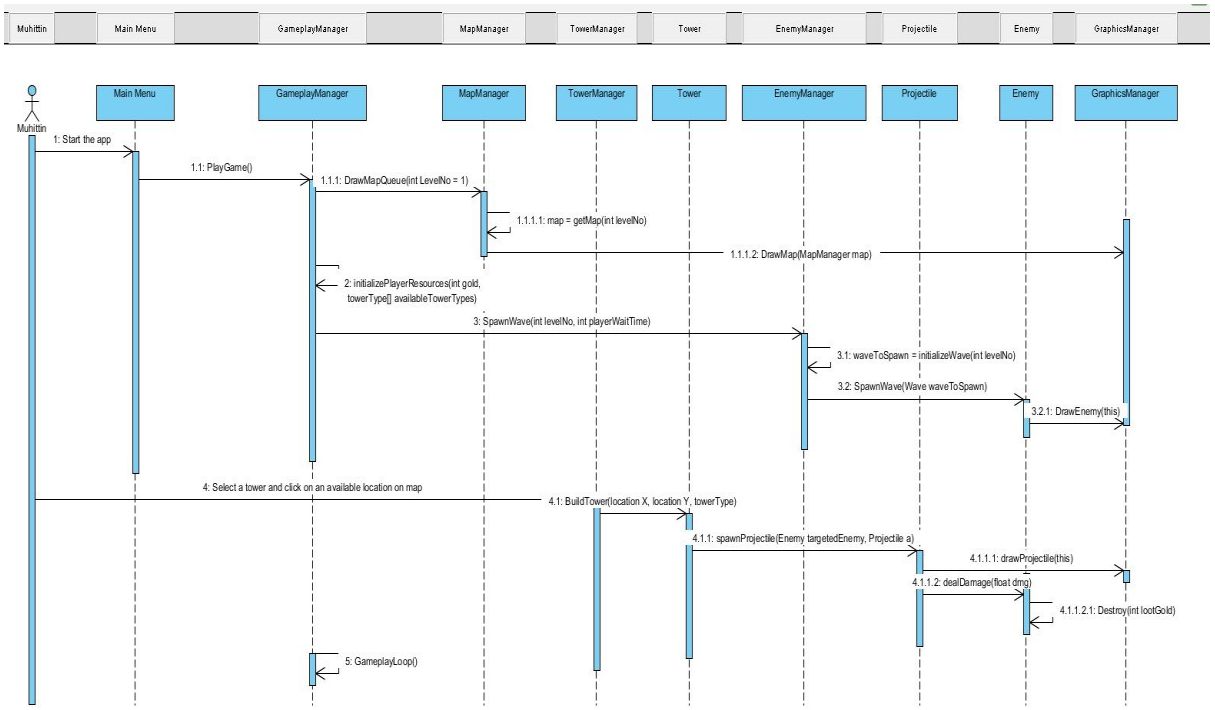
Exit Condition: Player hits the “Back” button to return back to the main menu.

Event Flow:

1. The player hits the “High Scores” button.
2. The system shows the nicknames of the top ten scorers.
3. The player hits the “Back” button when he is done examining the high scorers.

4.2 Dynamic Models

4.2.1 Start Game



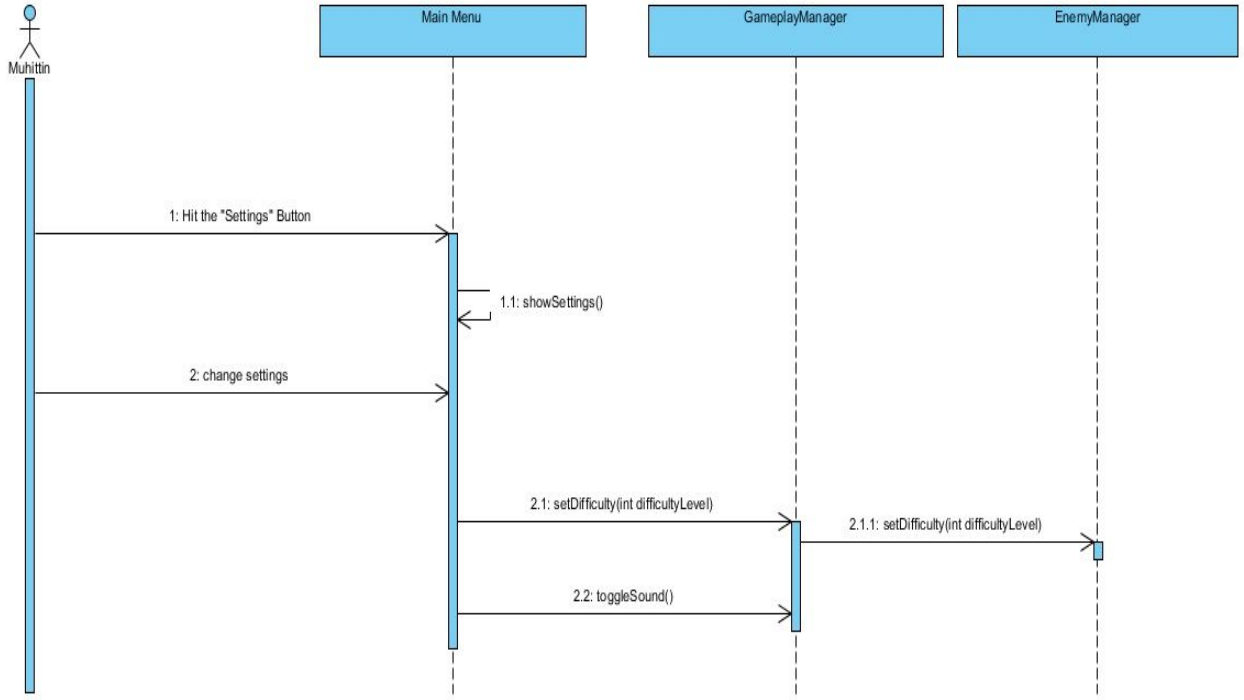
Event Flow:

Player Muhittin clicks the Play Game button to start playing. Then the MapManager system draws the pre-defined level and the pathway for enemies, then MapManager calls the GraphicsManager to display the graphics objects for these elements. Then the system gives the instructions and waits 30 seconds for the player to build towers. The player can then build the available towers with his resources.

EnemyManager creates the Enemy instances, each enemy instance calls the GraphicsManager to be drawn on the map. In the same manner, each tower instance calls the GraphicsManager to be drawn. For each enemy destroyed, the player receives resources to build/upgrade new towers. The amount of resources given to the player depends on the attributes of the enemy that is destroyed. Once all the enemies of a particular wave is either destroyed by the player or has crossed the end of the pathway, the system then goes into the gameplay loop for the next wave and again waits 30 seconds for the player to prepare new towers.

Once the player wins or loses the game, the system asks for the nickname of the player if his score is in the top 10 among all scores so far.

4.2.2 Change Settings



Muhittin wants to change the settings. He clicks on the settings from the main menu. The system displays the settings screen. Once Muhittin is done playing with the settings, the changes are saved on the GameplayManager object.

4.2.3 View Help

Event Flow:

Muhittin wants to get information about the game before starting to play. He clicks on the "Help" button. The system then displays info on most of the gameplay elements.

4.2.4 View Credits

Event Flow:

Here Muhittin wants to get information about the developers. The system displays the names and contact info about the developers.

4.2.5 View High Scores

Event Flow:

Here Muhittin wants to see the highest scorers. He clicks on the “High scores” button. The system gets the high scores from the saved location and displays them on screen.

5. User Interface

5.1 Navigational Path

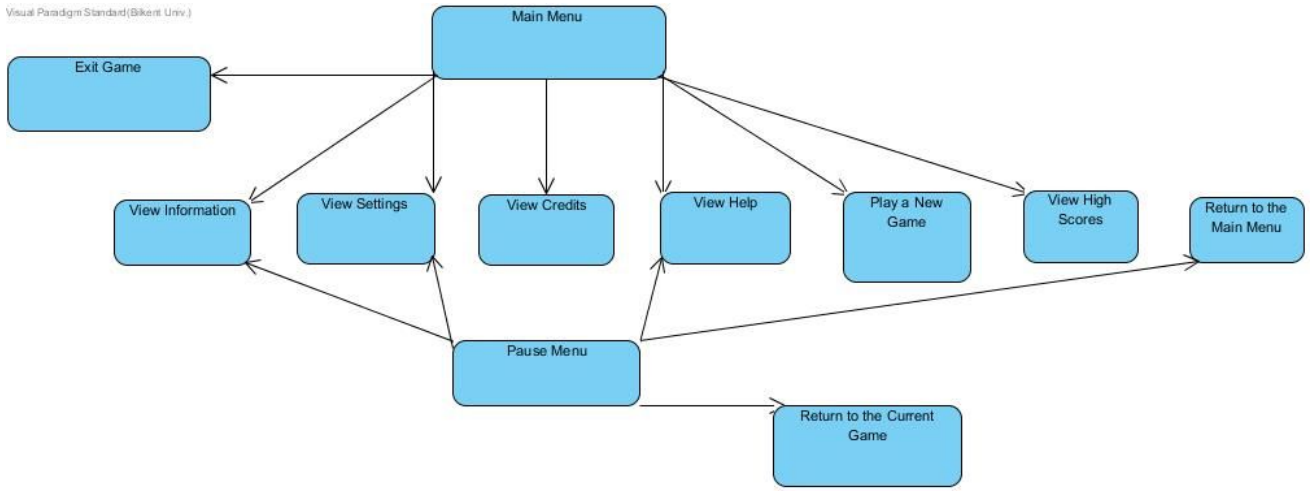


Figure 5.1 Illustrates the Navigational Path of Medieval Tower Defense

5.2 Screen Mock-ups

5.2.1 Main Menu

When the game is started, player will see the main menu screen. Main menu contains six options as buttons and an exit icon. Player will be able to choose between playing the game, change settings, get information about the units in the game, get help, see his/her and other player's high scores, see credits and exit the game. (Figure 5.2.1)

Figure 5.2.1 A Balsamiq mock-up of the main menu

- Play Game:

If the user selects the play game option, game starts with the selected settings. (Figure 5.2.1.1)

Figure 5.2.1.1: Balsamic mock-up of the gameplay

- Settings:

When the player selects the settings button, a screen containing several options appears on the screen. (Figure 5.2.1.2) These settings can be changed to the desired levels by the player. Automatically, default settings are selected. Players can change game difficulty and turn the volume up and down.

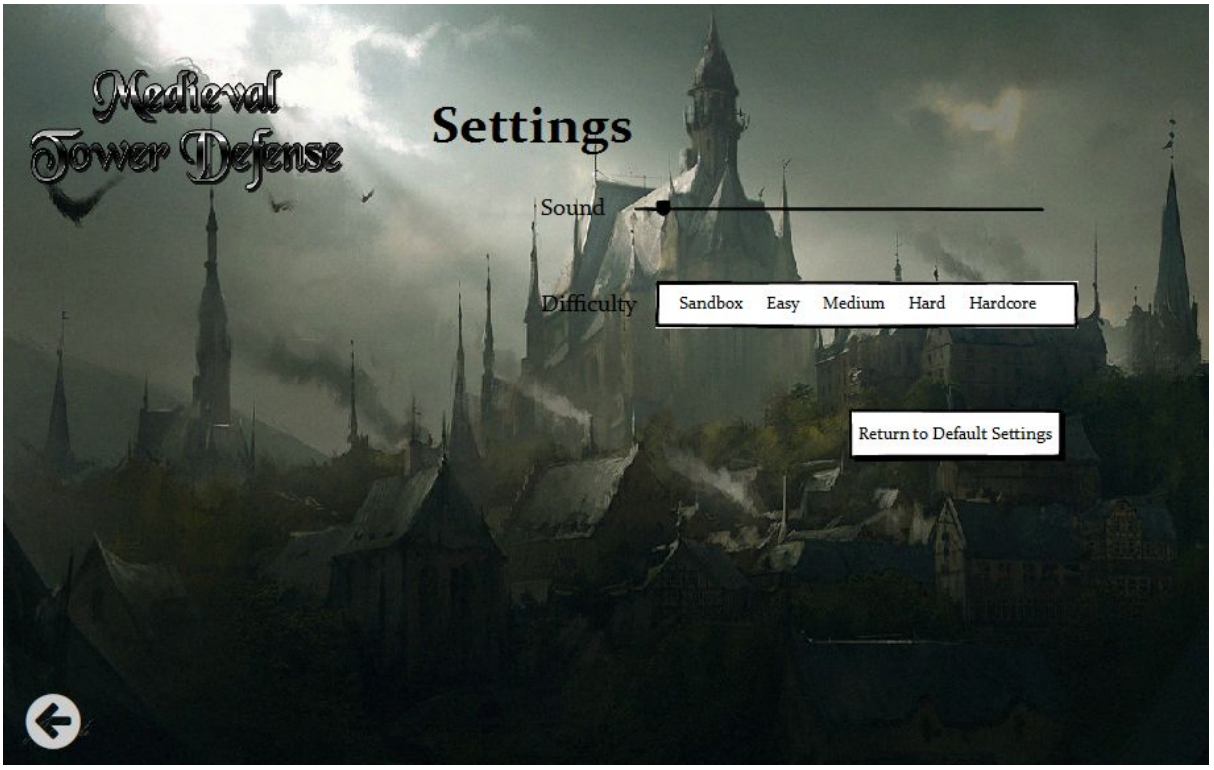


Figure 5.2.1.2: Balsamiq mock-up of the change settings

- Information:

When the player selects the information button, a screen containing the tech-tree of the units in the game appears.(Figure 5.2.1.3) The player can read about the attributes of the enemies, towers, waves and gather information about their strategies in this window.

Figure 5.2.1.3: Balsamiq mock-up of the information screen

- Help:

When the player selects the help button, a screen containing the controls, basic concepts and aims of the game appears. (Figure 5.2.1.4) The player can read about the controls and how to play the game in this screen.

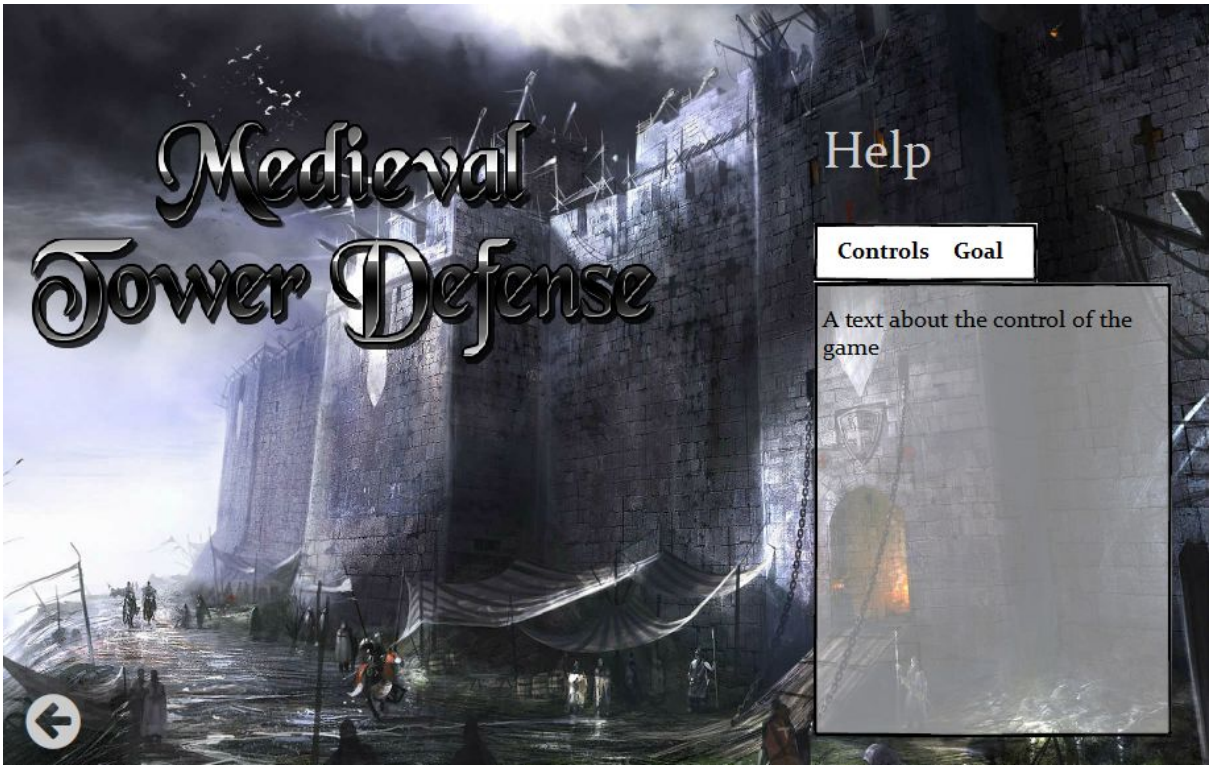


Figure 5.2.1.4: Balsamiq mock-up of the help screen

- High Scores:

When the player selects the high scores button, a screen that shows the top 10 high scores of the game appears. (Figure 5.2.1.5) These top 10 high scores are arranged according to the wave progress of the players. It priorities the wave progress, but if two players' wave progress is same, then it chooses the one with less time spent.

Figure 5.2.1.5: Balsamiq mock-up of the high scores screen

- Credits:

When the player selects the credits button, a screen containing the information about the developers appears (Figure 5.2.1.6)

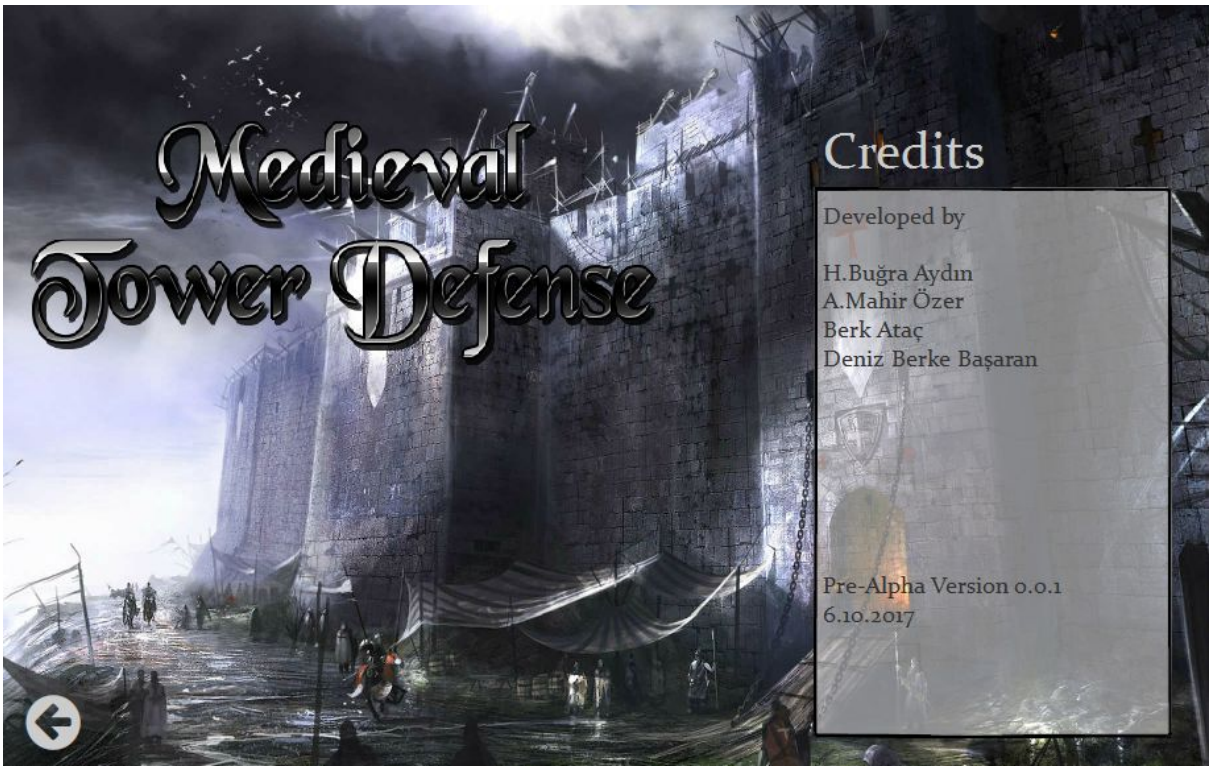


Figure 5.2.1.6: Balsamiq mock-up of the credits screen

- Pause Menu

If the player decides to pause the game during play game feature, a screen containing several options appears. (Figure 5.2.1.7)

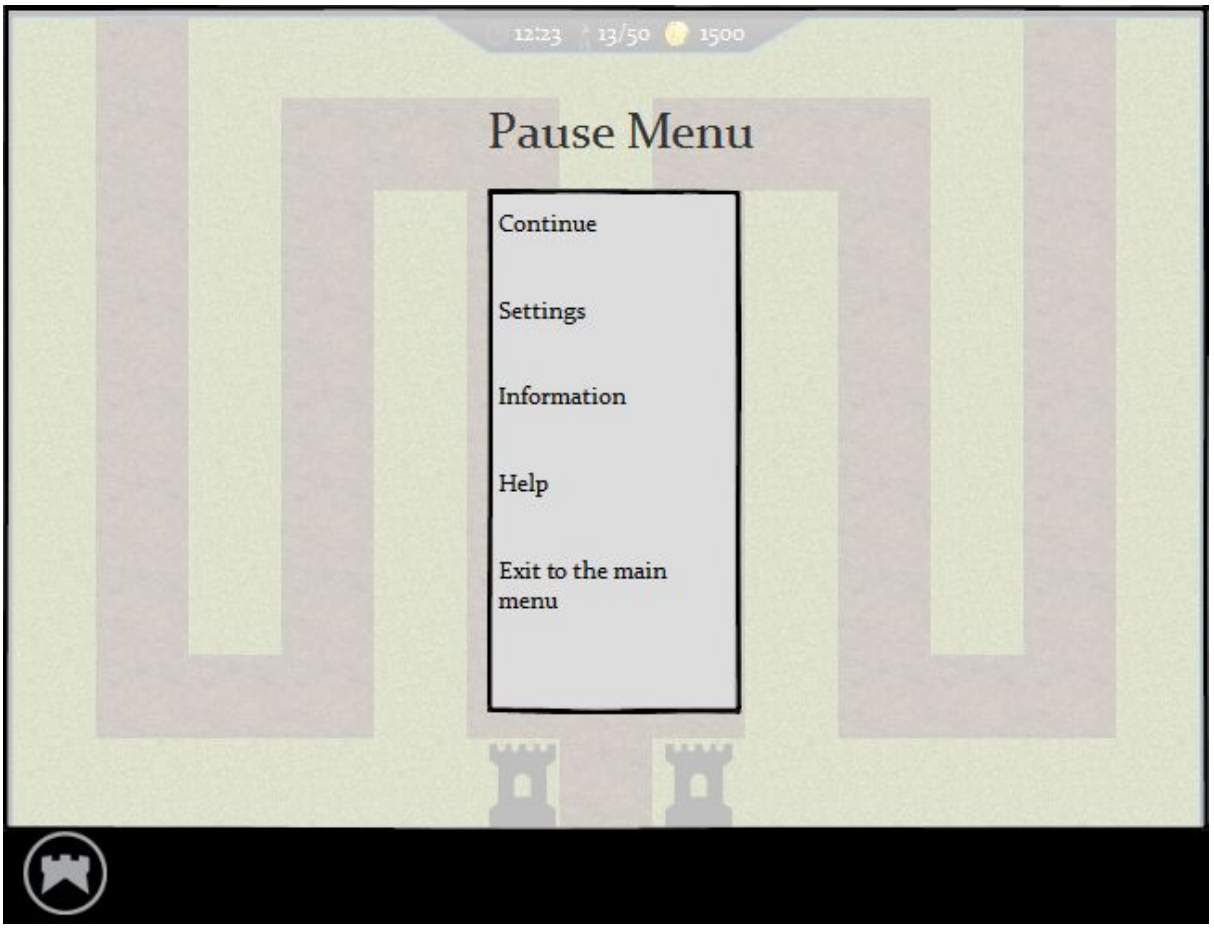


Figure 5.2.1.7: Balsamiq mock-up of the pause menu

6. Conclusion

This report contains our analysis of the medieval tower defense game we are creating. In this report, the requirements and conditions were specified, then the system models were given.

The system models were use case models, dynamic models, the class model and the user interface. We tried to create the models as simple as possible in order to avoid making things more complicated than needed when creating the game.

When determining the requirements, we thought about what the player could do in the game, from changing the settings to actually playing the game. The requirements showed clearly what the play is able to do, and with that knowledge we created the use case models.

7. References

https://sites.google.com/site/knightsvszombies/
http://www.cs.bilkent.edu.tr/~ugur/teaching/cs319/proj/11_2C/analysis.doc