****

Bilkent Universtity

Department of Computer Engineering

CS 319 Term Project

Group 1D  
Wars & Warriors

Requirement Analysis Report

Project Group Members:

**Samet DEMIR  
Ahmet MALAL  
Mahammad SHIRINOV  
İbrahim MAMMADOV  
Huseyn ALLAHYAROV**

Supervisor: Eray TUZUN

Table of Contents

[1. Introduction 4](#_Toc531125414)

[2. Overview 4](#_Toc531125415)

[2.1. Gameplay 5](#_Toc531125416)

[2.2. Map 5](#_Toc531125417)

[2.3. Walls 5](#_Toc531125418)

[2.4. Warriors 5](#_Toc531125419)

[2.5. Menus 5](#_Toc531125420)

[2.5.1. Main menu 5](#_Toc531125421)

[2.5.2. Walls Menu 6](#_Toc531125422)

[2.6. Settings 6](#_Toc531125423)

[3. Improvements summary 6](#_Toc531125424)

[3.1. Map 6](#_Toc531125425)

[3.1.1. Different Sizes 6](#_Toc531125426)

[3.1.2. Immobile Objects 6](#_Toc531125427)

[3.2. Walls 6](#_Toc531125428)

[3.2.1. New Shapes 6](#_Toc531125429)

[3.2.2. Chains 6](#_Toc531125430)

[3.2.3. Strength 7](#_Toc531125431)

[3.3. Warriors 7](#_Toc531125432)

[3.3.1. Attackers 7](#_Toc531125433)

[3.3.2. Walkers 7](#_Toc531125434)

[4. Functional Requirements 7](#_Toc531125435)

[4.1. Main menu 7](#_Toc531125436)

[4.2. Play 7](#_Toc531125437)

[4.3. How to play 8](#_Toc531125438)

[4.4. Settings 8](#_Toc531125439)

[4.5. Credits 8](#_Toc531125440)

[4.6. Quit 8](#_Toc531125441)

[5. Non-functional requirements 8](#_Toc531125442)

[5.1. Performance 8](#_Toc531125443)

[5.2. Interactivity 9](#_Toc531125444)

[5.3. Compatibility 9](#_Toc531125445)

[6. System Models 9](#_Toc531125446)

[6.1. Use Case Diagram 9](#_Toc531125447)

[6.1.1. Use Case: Play (Figure 1) 10](#_Toc531125448)

[6.1.2. Use Case: Access Game Manual (Figure 1) 11](#_Toc531125449)

[6.1.3. Use Case: Go to Credits (Figure 1) 11](#_Toc531125450)

[6.1.4. Use Case: Pause (Figure 1) 12](#_Toc531125451)

[6.1.5. Use Case: Choose Level to Play (Figure 2) 13](#_Toc531125452)

[6.1.6. Use Case: Select an Object (Figure 3) 14](#_Toc531125453)

[6.2. Dynamic Models 16](#_Toc531125454)

[6.2.1. Sequence Diagrams 16](#_Toc531125455)

[6.2.2. Activity Diagrams 20](#_Toc531125457)

[6.2.3. State Diagrams 21](#_Toc531125458)

[6.2.4. Object Model 23](#_Toc531125459)

[6.3. User Interface: Navigational Path & Screen Mockups 25](#_Toc531125460)

[7. References 30](#_Toc531125461)

# Introduction

Walls & Warriors is a board game played on a board with a castle and 2 opposing groups of warriors, and walls to be placed onto the board during the game. The main objective of the game is to place the walls in such a manner that they form one closed fortress so that the castle and one group of warriors – the blue ones – are within the borders, and the other group - the red ones – are outside. This project aims to develop a desktop version of this game with some additional features that are not present in the original version. The game is implemented in Java with the further improvements to increase the player affinity and the level of complexity for a better and more enjoyable experience.

This document is intended to describe the requirements analysis for the project. For more technical details about the project, functional, dynamic and object models have been introduced in the subsequent sections which have been made in the Visual Paradigm [1]. Technical and non-technical requirements describe the programmers’ task of implementation for the best experience in visual and non-visual aspects of the game. Mockups which have been produced by the help of Balsamiq [2], help to visualize the final game that our project intends to achieve. Proceeding parts of the project will be briefing the information about the details given above.

The classic board game of Walls & Warriors has 3 main components: map, warriors and walls. The map is a fixed size board, and the warriors are placed on the cells of the map. There are 4 different shapes of walls in the game and an immobile castle that also needs to be encapsulated within these walls. Our project improves and adds new features to the original set. In particular, the map size is not fixed; more advanced levels will have larger maps, and more soldiers. Also, unlike in the original game, where the castle is the only ‘obstacle’ through which the player cannot put a wall, there are other immobile objects. These will span a few map cells and will subside player’s freedom in placing the walls in certain ways. These immobile objects include a forest and a lake. Another function of the lake is that, in certain levels, it can contain ships of two colors, which will be further needed to be separated with chains, which carry the function of the walls in the lakes. The player should place the walls in such a manner that he/she does not cross these immobile objects. The game will have new types of warriors – attackers and walkers – in contrast with the classic game, in which all warriors are intrinsically the same. Among these, attackers are red warriors with the ability to damage, and given enough time, break the walls placed on the board. Walkers are warriors that can be members of either color, which change their places throughout the game and walk in a specified trajectory. The walkers and attackers impose a time constraint in the game for the player, as they have to be quick to act to avoid the collapsing of the walls or keeping the warriors inside (or outside) before they change their place. Last but not least, walls have new features too. There are new shapes of walls and chains as well as new strength feature of the walls which determine their immunity to the attackers, which we described briefly above. These new features add a degree of challenge and diversity to the game experience, as well as adding time constraints on certain levels.

# Overview

This section aims to explain the general overview of the game and key features associated with the gameplay. Additionally, main attributes and parts of the gameplay are explained, to describe functionalities of these attributes. Map, being the most interactive part of the game, is one of the features that will be explained, along with the added features.The objects placed on the map, such as the walls and warriors, and their functions will be described in the proceeding subsections. Menus that are used to navigate in the game will be explained with all the aspects. Finally, settings for the gameplay that are customized by the user will be described in the last subsection.

## Gameplay

Walls and warriors is a strategic board game that is being implemented in 2D for the desktop environment via this project. Game is consisting of a map that is populated with warriors of different kinds and walls for keeping the enemies outside the borders of the castle of a certain kind of warriors. Main responsibility of the user is to find a way to place the walls in such a manner that they do not overlap and the final castle includes only the warriors of one kind, mainly, blue warriors. The board game has a fixed size map and 2 kinds of warriors, blue and red, and 4 types of walls with different shapeswhich a user can pick one of each at most. The game always has one solution which means there is only one way to place the walls in such a manner that the user will win. This project is based on these key features with more additional ones for increased complexity. These additional features include different sizes of maps, immobile objects and obstacles on the map, different shapes of walls, new functionalities of certain warriors and more which will be explained in more detail in the improvements summary. With the additional features, game may have more than one solution and a certain algorithm will be used to determine if the game is over or not.

## Map

Map is the fundamental part of the game that all the action is happening on. Map is the base for the walls and warriors to be placed on. With the additional features, in the proceeding levels, map may have natural immobile objects such as lakes, forest or rocks to challenge the user to come up with a solution that circumvents these obstacles.

## Walls

Walls are the fundamental parts of the final castle which will determine if the user wins or not. With many shapes that satisfies different scenarios, user will be responsible for selecting a wall from a menu to place on the map. The walls will have some strength that may be decreased with the attacks of the enemy warriors which challenges user to think and play faster before they collapse. The strength feature will be included in harder and more complex levels.

## Warriors

Warriors are the main characters of the game user interacts with. The placement of the warriors is generally fixed and determine the solution of the game. This project will be granting some warriors with new features to walk in certain directions for the blue warriors and attack the walls for the red warriors which adds a time constraint. These new features will increase the complexity in the harder levels.

## Menus

In the game there are 2 menus that a user interacts with. This subsection will be describing these 2 menus in the following paragraphs.

### Main menu

Main menu is the first screen that welcomes the player. The menu includes buttons to start playing, reading the guidelines and instructions, customize the game settings, display the credits and to quit the game. User needs to select the play option to start the game and select the level.

### Walls Menu

In the game, there will be a walls menu that displays the different kinds of walls a user may pick. This menu may present more options in accordance with the complexity of the current level.

## Settings

Settings menu will allow the user to customize the settings for the desired gameplay. The customizable settings will include adjustment options for the game sounds (SFX) and background music.

# Improvements summary

To challenge the user and increase the complexity of the game, many additional features have been added to the classical board game. These features will be explained in the proceeding subsections which will describe the additions to the map, walls and the warriors.

## Map

### Different Sizes

Unlike the board game, this game comes with different sizes of the maps to increase the complexity. Levels will start easy and advance accordingly. One of the advancements include the increase in the map’s size which will allow it to house more warriors and therefore, more walls and obstacles.

### Immobile Objects

As the map expands, immobile objects may be generated accordingly to increase complexity and challenge the player. These objects include lakes, forest and rocks that cannot be moved and need to be ignored to build the castle. Lakes may house ships of different warriors. Blue ships should be in the castle’s area while the red ships should remain outside. In that scenario, walls may contain the blue ship by the chains that are attached to the walls.

## Walls

### New Shapes

In the board game there are 4 wall shapes and they all can be placed on a single fixed-size map. Because of the varying sizes of maps in this game, new wall shapes are also introduced accordingly.

### Chains

Like the walls on surface, there may be chains available to replace the walls’ functionality in the water. In certain scenarios where there is a lake on the map and that houses 2 different kinds of ships, chains may be used to house the blue ship in the castle’s field. These chains will appear in these scenarios on the walls menu and should be attached to the walls by the both ends.

### Strength

Walls in the harder levels will have some strength levels that can be decreased by attackers on the red ships. This same rule applies to the chains as well which may be damaged and collapsed by the attackers. This rule adds a virtual time constraint and challenges the user think and decide faster to win the game. If the strength of the walls decreases to 0, game is over and player loses.

## Warriors

### Attackers

As mentioned in the previous subsections, in the maps that house a lake with red ships, there will be red warriors, in other words enemies, who will attack and try to collapse the walls or the chains. These warriors will be seen in the harder levels.

### Walkers

Rather than being in the same location and fixed place, some blue soldiers will be walking in certain direction. Player needs to place the wall accordingly to contain these walking warriors. Like the attackers, these warriors will appear in the more complex levels.

# Functional Requirements

This section includes the requirements and expectations for the key features of the game that a user will be interacting with. These features include main menu, play, how to play, settings, credits and quit options. Design and functionalities of the menus and the game have been designed carefully with a simple, user-friendly and concise structure in mind to avoid confusion and dissatisfaction.

## Main menu

Main menu is the main screen that will be encountered by the user at start. This screen should have a user-friendly UI and should be simple to use. A new user should be able to get the grasp of the menu and the game in the first run. This menu will let user to choose one of the options that are being displayed and needs to respond accordingly. The main menu will be displaying play, how to play, settings, credits and quit options and will respond with the pre-defined functions for different scenarios.

## Play

Before launching the gameplay, the user should be able to select among the levels that are available to him to start the game. The levels should be maintained in such a way that the user will be able to access all the levels they have completed, and the latest level they haven’t. All other levels should be locked away from the user and will be unlocked as the user advances through the levels in a consecutive manner. After the player has completed a level, the next level should be made available in the game, and the current progress of the user saved. Player may either immediately start playing the next level or come back to the levels menu to see which levels have been made available. A simple menu will be displayed in the end of the game to choose one of these 2 options.

When a user chooses a level to play, the system will set up the map (both logically and graphically) with the warriors and the castle placed, and will display it to the user. The components should be recognizable to the user. For that reason, walls menu will have its name on it. Warriors will be distinguishable based on their corresponding colors and shapes. When the user attempts to place a wall on the map, the game must decide whether the selected wall can be placed on the specified location by the user. The game will be periodically checking if the user has won yet or not and will respond immediately if the game is over either by a win or because of collapse of the walls, with the accompaniment of sound effects. If user loses the game because of collapse of the walls, a simple menu should pop up informing the user that the walls collapsed, and asking for a restart or return to the main menu.

## How to play

How to play section should describe the gameplay, rules and instructions as its name suggests. Basic rules of the board game and features of the walls and warriors with the additional ones will be provided in form of text, explaining the gameplay thoroughly. The official rule set for the game will be added with the pictures that explains the scenarios and the guidelines about them.

## Settings

A settings section should be available to allow the user to customize the game setting for a desired experience. These settings will allow the adjustment of the game sounds (SFX) and background music volume.

## Credits

Credits will display the information about the developers and the contributors of the project. Their names will be displayed on the screen. Some additional information about the developers may also be included in addition to the names such as linkedin accounts or github repository of the project.

## Quit

As the name suggests, quit option will allow the user to quit and terminate the game from the main menu. A second pop-up menu will be displayed to confirm the termination of the game. Placement and the naming of this option have been decided carefully for the simplicity reasons, thus, this option will be last one in the menu. This option will appear only in main menu but a terminate button will also work any time to close the window and terminate the process immediately.

# Non-functional requirements

Main intention of this section is to explain the non-functional requirements which serve the developer responsibilities. These responsibilities include performance, interactivity and compatibility issues of the game.

## Performance

The game should not waste resources of the system such as RAM or storage. Game should allocate 25MB of RAM and the runnable JAR file should be 1MB. For the compactness reason, sound files used should be within 70KB limit. Lags and performance decrease or FPS drops should be solved for the final product. Main reason to choose the Java language for the implementation is to use tools and features of the JVM to tackle these issues with the help of garbage collector, for example. Screen tearing should be avoided. In other words, game should maintain a FPS rate accordance with the display in use to not to respond with a higher refresh rate that is used by the displaying device. An ideal 30-60 FPS rate should be maintained to fix any probable bugs related to this issue as it is the case with most of the monitors in use and considering the idleness of the game. We intend to tackle the visual challenges easily with the help of documentation on the swing and awt classes we use for this purpose. In our tests, we have not experienced any problems regarding the graphics. CPU usage should also be reasonable for a smooth experience and sound production. In our tests CPU usage did not come close to 1%.

## Interactivity

Game’s responsiveness is another challenge to be tackled for a smooth user experience. User interface also should be as simple as possible to increase satisfaction of the customer. Any operation or animation (such as dragging a wall to the map, or removal of the wall to its place upon invalid move) should provide a ‘smooth’ feel – they must take no longer that 0.05 seconds.

## Compatibility

The game should be compatible with all the desktop platforms and operations systems in use. Java’s JVM tackles this issue as it comes with its own VM to run the Java programs.

# System Models

## Use Case Diagram

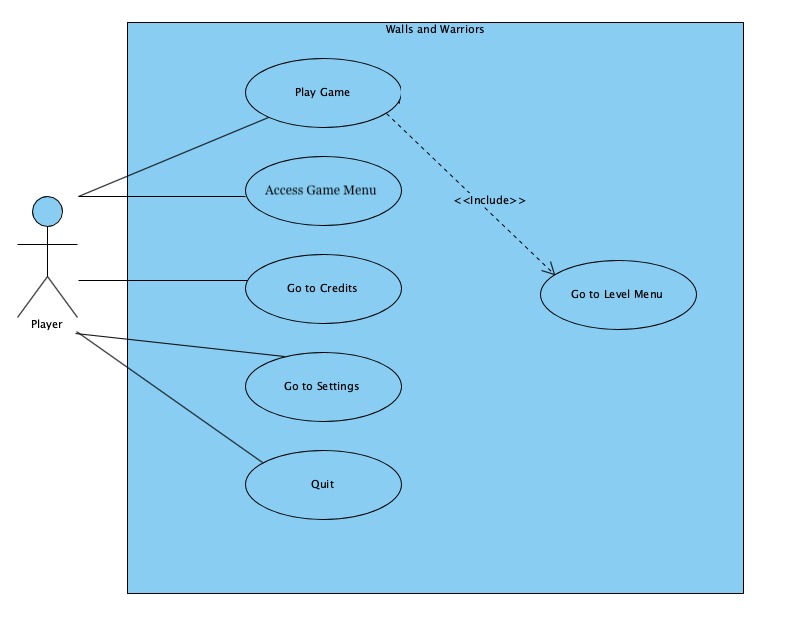


Figure Main Menu Use Case

### Use Case: Play (Figure 1)

Primary Actor: Player

Interests: 1. Player wants to play the game.

2. System launches the game.

Pre-Conditions: 1. Player has started the game.

2. Player is in main menu.

Entry-Conditions: 1. Player clicks “Play” button in the main menu.

Exit-Conditions: 1. Player clicks “Pause” button in the game menu.

2. Player clicks “Back” button in the level menu.

Success scenario event flow:

1. Player clicks “Play” in the main menu.
2. Player selects an unlocked level from level menu.
3. Player chooses a wall and changes its positions by using direction keys on the keyboard.
4. Player drops the selected wall in suitable place.
5. Player chooses a wall and changes its positions by using mouse.
6. Repeat 4.
7. Player chooses a chain and changes its positions by using direction keys on the keyboard.
8. Player drops the selected chain in suitable place.
9. Player chooses a chain and changes its positions by using mouse.
10. Repeat 8.
11. Repeat 3-10.
12. Level is completed.

Alternative Event flow:

1. Player presses pause key while in game.
2. Player selects quit game option.
3. Walls & Warriors closes the game.

### Use Case: Access Game Manual (Figure 1)

Primary Actor: Player

Interests: 1. Player wants to know how to play the game.

Pre-Conditions: 1. Player is in the main menu.

Entry-Conditions: 1. Player clicks “How to Play” button in the main menu.

Exit-Conditions: 1. Player clicks “Back” button in the how to play screen

Success scenario event flow:

1. Player presses How to Play button.
2. Player views tutorials to learn how to play Walls & Warriors
3. Player presses Back button
4. Player goes to main menu.

### Use Case: Go to Credits (Figure 1)

Primary Actor: Player

Interests: 1. Player wants to know credits.

Pre-Conditions: 1. Player is in the main menu.

Entry-Conditions: 1. Player clicks “Credits” button in the main menu.

Exit-Conditions: 1. Player clicks “Back” button in the how to play screen

Success scenario event flow:

1. Player presses Credits button.
2. Player views the developers.
3. Player presses Back button.
4. Player goes to main menu.

### Use Case: Pause (Figure 1)

Primary Actor: Player

Interests: 1. Player wants to change the settings.

Pre-Conditions: 1. Player is in a level.

Entry-Conditions: 1. Player clicks “Pause” button in the game menu.

Exit-Conditions: 1. Player clicks “Back” button in the pause screen.

Success scenario event flow:

1. Player presses Pause button.
2. Walls & Warriors pauses the game screen
3. Walls & Warriors displays the pause menu.
4. Player selects to continue level.
5. Walls & Warriors continues the game.

Alternative Event flow:

1. Player presses Pause button.
2. Walls & Warriors pauses the game screen
3. Walls & Warriors displays the pause menu.
4. Player selects to continue level.
5. Walls & Warriors continues the game.
6. Player presses sound up button.
7. Player presses sound down button.
8. Player presses music up button.
9. Player presses music down button.
10. Player presses return the main menu.
11. Walls & Warriors display the main menu.

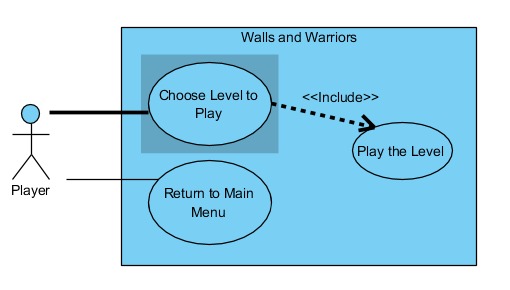


Figure Level Menu Use Case

### Use Case: Choose Level to Play (Figure 2)

Primary Actor: Player

Interests: 1. Player wants to choose a level.

Pre-Conditions: 1. Player is in the level menu.

Entry-Conditions: 1. Player chooses a level among the listed levels using a mouse.

Exit-Conditions: 1. Player chooses a valid level.

Success scenario event flow:

1. Player clicks a valid level.
2. Walls & Warriors opens the level.

Alternative Event flow:

1. Player clicks an invalid level.
2. Walls & Warriors doesn’t open the level.

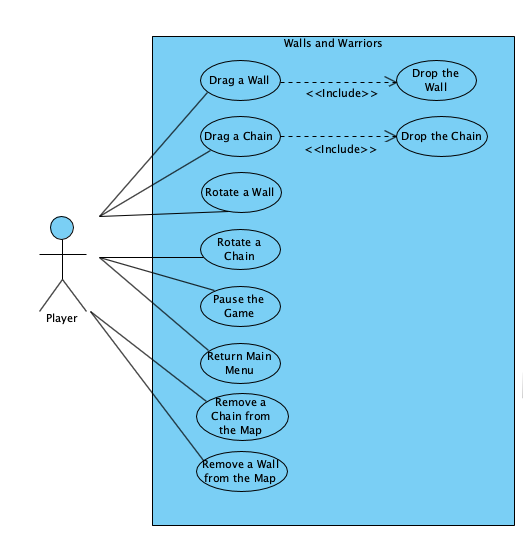


Figure Game Menu Use Case

### Use Case: Select an Object (Figure 3)

Primary Actor: Player

Interests: 1. Player wants to pick an object like a wall or chain.  
2. Player wants to change the position of an object it is already in the map.

Pre-Conditions: 1. Player is in a level.

Entry-Conditions: 1. Player clicks on an object using a mouse.  
2. Player clicks on the specific keys on the keyboard, as specified by the game manual.

Exit-Conditions: 1. Player releases the mouse.  
2. Player releases pressed key.

Success scenario event flow:

1. Player clicks an object in the wall/chain panel.
2. Player clicks the rotate button on the specific object.
3. Walls & Warriors rotates the selected object
4. Player presses the selected object and drops into the map.
5. Walls & Warriors changes the position of the selected object.

Alternative Event flow:

1. Player clicks an object in the wall/chain panel.
2. Player clicks the rotate button on the specific object.
3. Walls & Warriors rotates the selected object
4. Player presses the selected object and drops into the map.
5. Walls & Warriors does not change the position of the selected object.

## Dynamic Models

### Sequence Diagrams

### Iteration2/seq1.jpeg

Figure Discoverıng the Maın Menu Sequence Dıagram

#### Scenario 1 - Discovering The Main Menu

Entry Condition:

Player is on main menu

Exit Condition:

Player wants to exit the game

Main Flow of Events:

1- Player clicks on Play button.

2- View Controller displays the level menu.

3- Player clicks on the first level.

4- Game Controller checks if level 1 is currently open.

5- Game Controller tells the view to display level 1.

6- Player wants to go to main menu.

7- Player wants to go to How to Play screen.

8- How to Play is displayed.

9- Player presses the back button to go to main menu.

10- Player exits the game

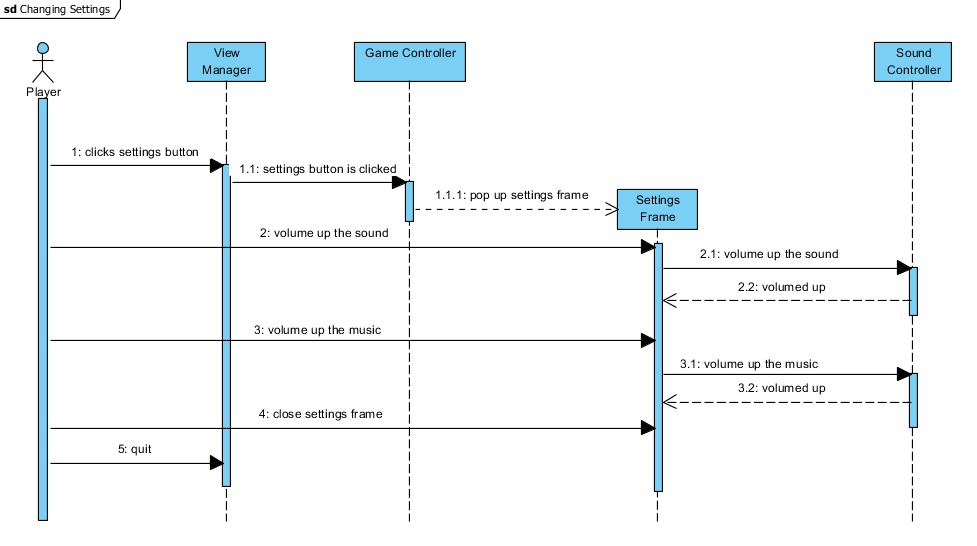


Figure Changıng Sound Volume Sequence Dıagram

#### Scenario 2 – Changing Sounds Volume

Entry Condition:

Player is in a level or in the Main Menu

Exit Condition:

Player closes the settings pane

Main Flow of Events:

1- Player clicks on the Settings button

2- Game Controller will show a pop-up for Settings

3- Player will turn off the music

4- Game Controller will update the Sound Controller

5- Player will increase the volume of the sound of the game

6- Game Controller will update the Sound Controller

7- Player will close the Settings Frame

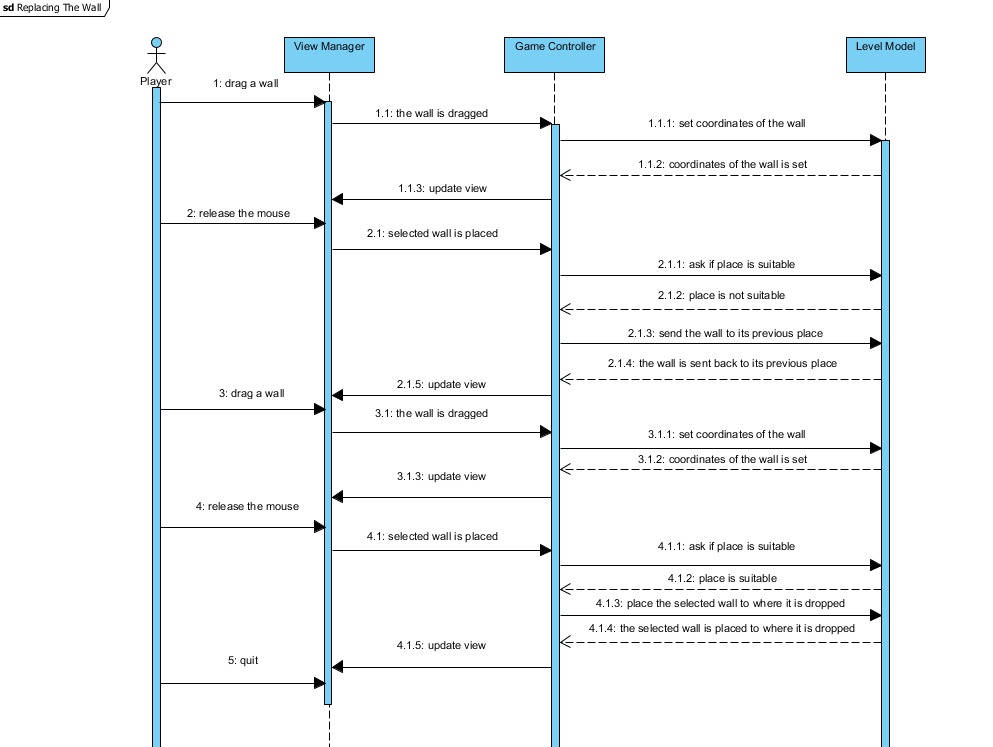


Figure Placıng the Wall Sequence Dıagram

#### Scenario 3 - Placing the Wall

Entry Condition:

Player is in a level

Exit Condition:

Player wants to go back to level menu

Main Flow of Events:

1- Player drags a wall and places it on an obstacle such as lake, tree and another wall

2- Game Controller observes the game state from the LevelModel and determines that the move is not valid

3- Game Controller sends the wall back to its previous place

4- Player drags a wall and places it outside of the board

5- Game Controller removes it from the board

6- Player drags the wall and places it to a suitable place

7- Game Controller observes the game state from the LevelModel and determines that the move is valid

8- Game Controller puts the wall into the specified place and updates LevelModel

### Activity Diagrams

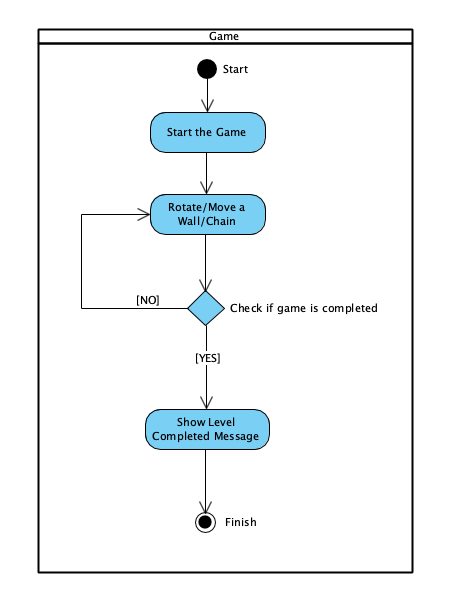


Figure Game Actıvıty Dıagram

- Player wants to play the game.   
- Player changes the positions of walls and chains.   
- If the positions of whole walls and chains are correct, Walls & Warriors displays the level completed message.

### State Diagrams

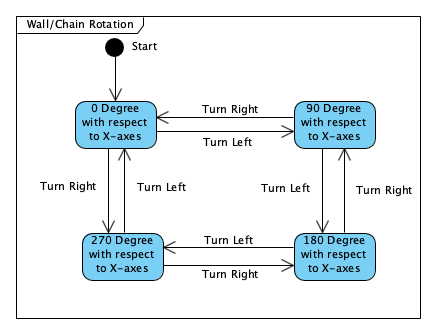


Figure Wall/Chaın Rotatıon State Dıagram

This state diagram shows the rotation angle states. User is able to rotate with 90 degrees each time.

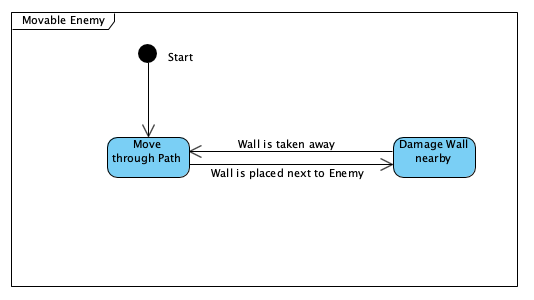


Figure Movable Enemy State Dıagram

This state diagram shows situation of movable enemy. Movable Enemy can damage the wall nearby or move through the given path.

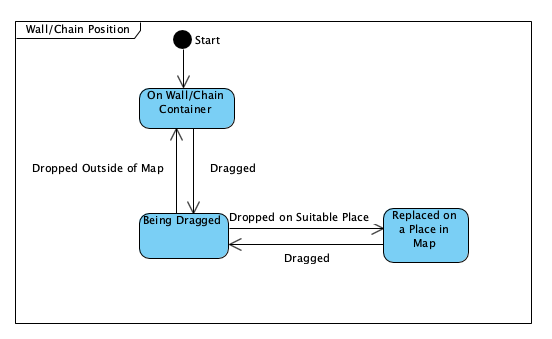


Figure Wall/Chaın Posıtıon State Dıagram

This state diagram shows that the position states of Wall/Chain.

### Object Model

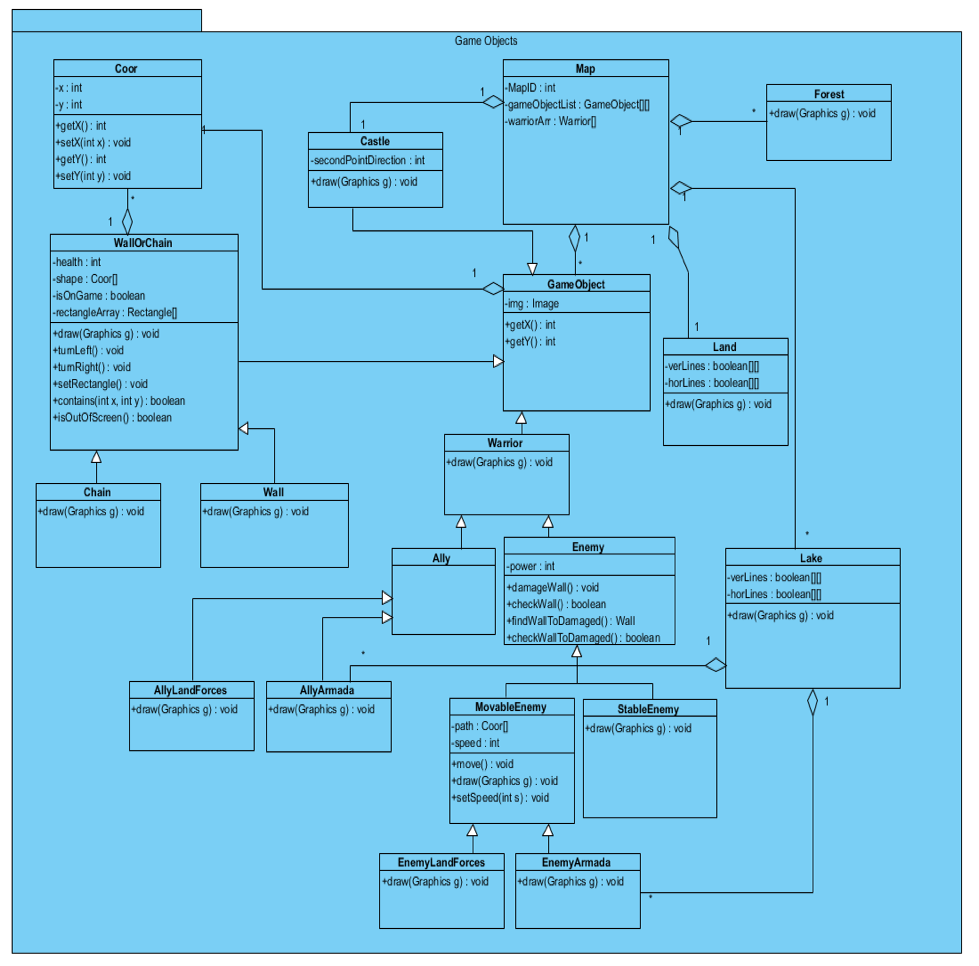


Figure Game Objects Class Dıagram

#### Entity Classes

**Warrior**: It is the parent class of Enemy and Ally classes.

**Enemy**: It is the warrior of enemy of the player. Some of them will be able to break the wall when they are near any wall or able to move through a specified path.

**Ally**: It is the warrior of the player. As the aim of the game implies, Allies should be inside the walls.

**Coor**: It simply keeps the x and y coordinates

**Obstacle**: It is the parent class of Forest and Lake classes. This class will not allow the wall/chain to be placed on that place.

**Forest**: It is a type of obstacle

**Lake**: It is a lake in which there can be enemy ships and ally ships. The player is supposed to separate them by using chains.

**Castle**: It is the castle of player around which player has to wrap by walls

**WallOrChain**: It is the parent class of Wall and Chain

**Chain**: It separates ally ships from enemy ships

**Wall**: It separates ally warriors from enemy warriors

**Land**: It is a land in which there can be enemy and ally warriors. The player is supposed to separate them by using walls.

**Game Object:** It is an upper class of all kinds of objects that are used in a level.

**Map:** This class keeps the information of the positions of all object.

**AllyLandForces:** This class is specific type of ally warriors which are knights on the land.

**AllyArmada:** This class is specific type of ally warriors which are ships on the lake.

**MovableEnemy:** This class is specific type of enemy warriors which are able to move through a pre-determined path.

**StableEnemy:** This class is specific type of enemy warriors which are not able to move.

**EnemyLandForces:** This class is specific type of enemy warriors which are knights on the land.

**EnemyArmada:** This class is specific type of enemy warriors which are ships on the lake.

## User Interface: Navigational Path & Screen Mockups

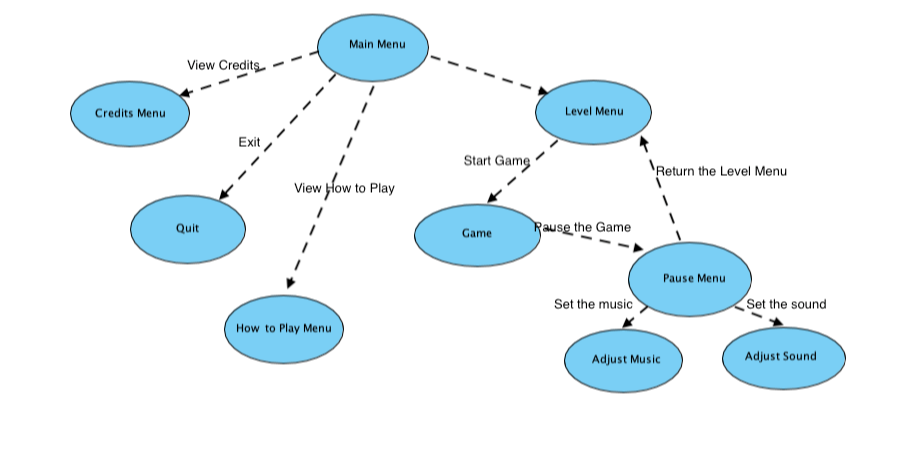
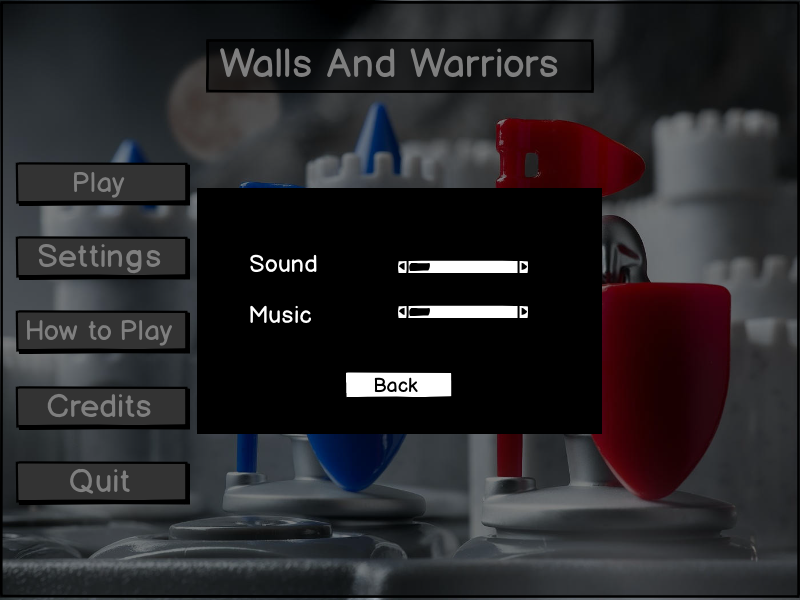


Figure Navıgatıonal Path

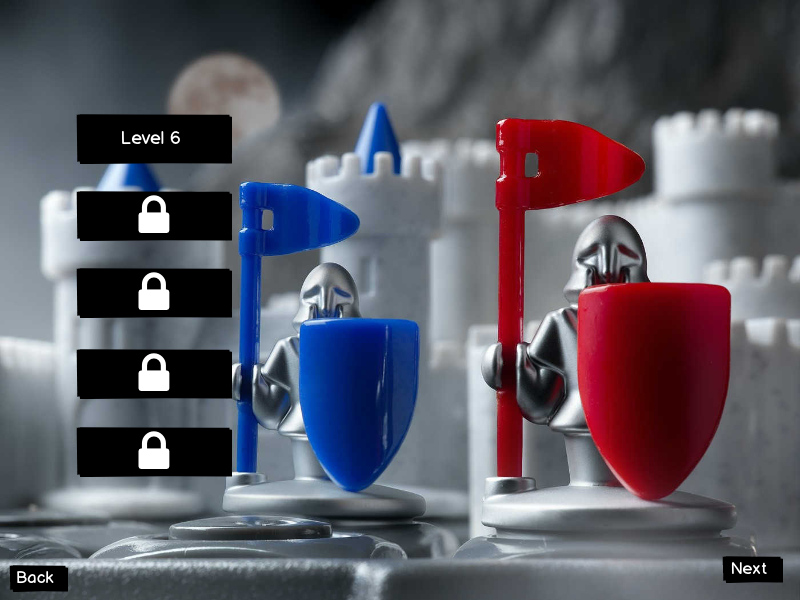
The main menu which is displayed at startup contains buttons to start and quit the same and learn about its creators or about the game. The “Settings” button will open the Settings page (below).

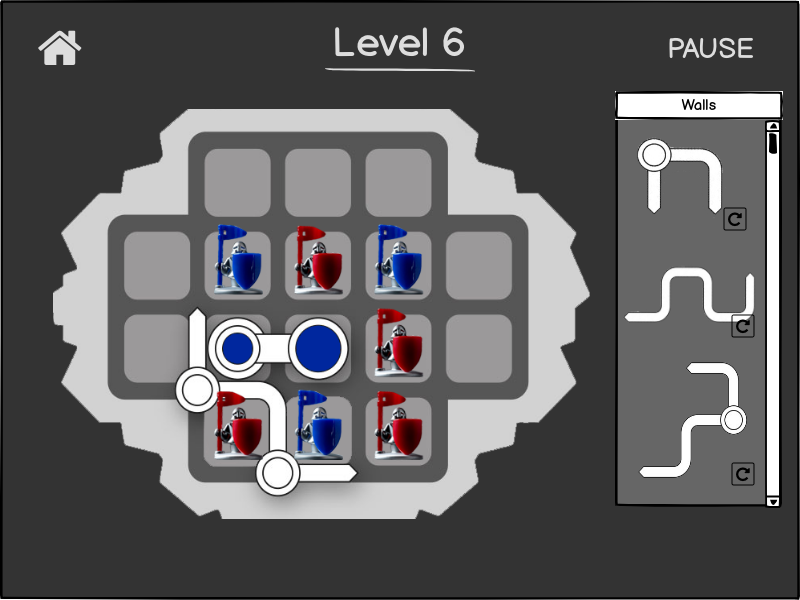


In Settings menu, you can change the volume of background music and sound effects.

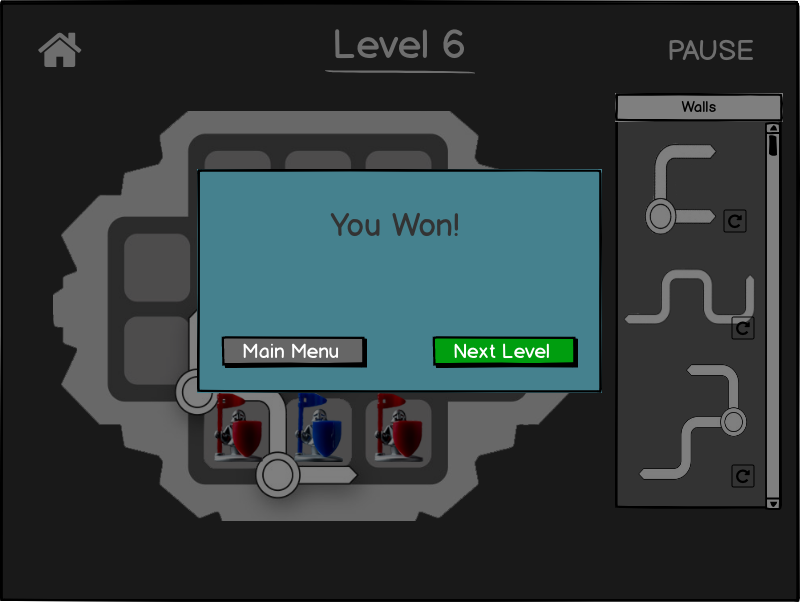


When “Play” button is pressed in main menu, the user is directed to the “Selects Levels” menu, to choose which level they want to play.

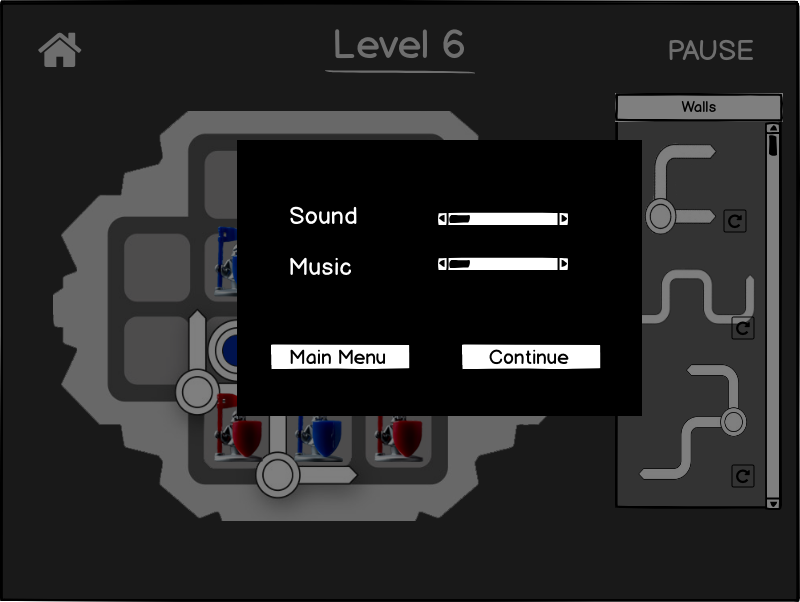


Choosing a level brings one to the game screen. The game screen displays the game board and the walls the player can select from. The player can rotate the walls by 90 degrees using the ‘rotate’ button next to each wall item. It also has buttons to go back to the main menu or pause the game (see below).

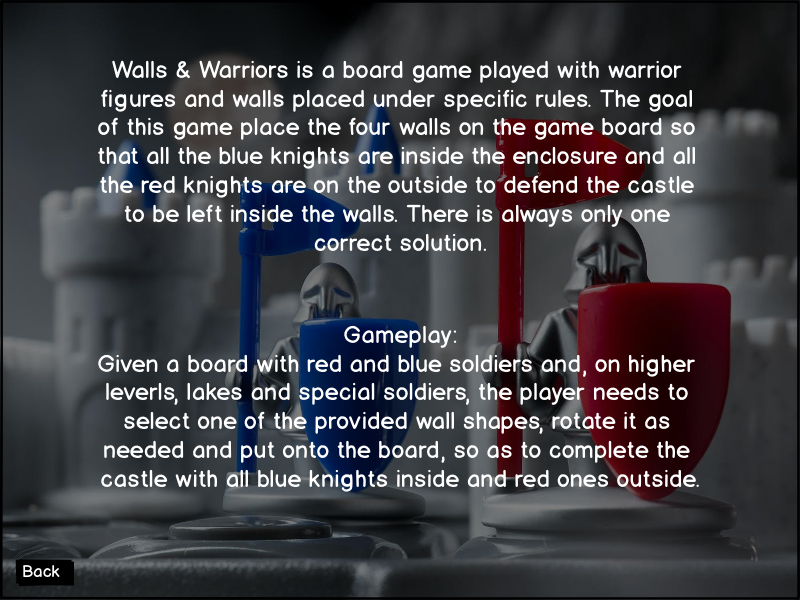
When you complete the challenge, a pop-up appears and you can proceed to the next level.



When the game is paused, the player can change some basic settings or return to the main menu.



The “How To Play” screen gives the user information about the game and instructions about how to play.



Finally, the “Credits” screen gives a list of the creators of the game. Some additional information can be given here.



# References

Balsamiq – Mockup Software

<https://balsamiq.com>

Visual Paradigm – Diagram Design Software

<https://visual-aparadigm.com>