

CS 319

Object-Oriented Software Engineering  
Fall 2014

**Conquerors**

Analysis Report

Group H  
Çağla Arıkan

Metin Enes Uçan

Can Akgün

İsmail Can Dereli

Table of Contents

[1. INTRODUCTION 4](#_Toc402383945)

[2. GAME OVERVIEW 5](#_Toc402383946)

[2.1. Hero 6](#_Toc402383947)

[2.2. Castle 7](#_Toc402383948)

[2.3. Enemies 7](#_Toc402383949)

[2.3.1. Soldiers 7](#_Toc402383950)

[2.3.2. Cannonball 8](#_Toc402383951)

[2.3.3. Walls 8](#_Toc402383952)

[2.3.4. Mine 8](#_Toc402383953)

[2.4. Bonuses 9](#_Toc402383954)

[2.4.1. Power-ups 9](#_Toc402383955)

[2.4.2. Bad surprises 9](#_Toc402383956)

[2.5. Levels 10](#_Toc402383957)

[2.5.1. Level 1 10](#_Toc402383958)

[2.5.2. Level 2 10](#_Toc402383959)

[2.5.3. Level 3 11](#_Toc402383960)

[2.5.4. Level 4 11](#_Toc402383961)

[2.5.5. Level 5 12](#_Toc402383962)

[3. FUNCTIONAL REQUIREMENTS 12](#_Toc402383963)

[3.1. Play Game 12](#_Toc402383964)

[3.2. View Credits 13](#_Toc402383965)

[3.3. View Help 13](#_Toc402383966)

[3.4. Change Settings 13](#_Toc402383967)

[3.5. High Scores 14](#_Toc402383968)

[3.6. Level Selection 14](#_Toc402383969)

[4. NON-FUNCTIONAL REQUIREMENTS 14](#_Toc402383970)

[4.1. Game Performance 14](#_Toc402383971)

[4.2. Graphical Smoothness 14](#_Toc402383972)

[4.3. User Friendly Interface 15](#_Toc402383973)

[5. SYSTEM MODELS 15](#_Toc402383974)

[5.1. Use Case Model 15](#_Toc402383975)

[5.2. Dynamic Models 23](#_Toc402383976)

[5.2.1. Play Game 23](#_Toc402383977)

[5.2.2. Bonuses 25](#_Toc402383978)

[5.2.3. Change Settings 28](#_Toc402383979)

[5.2.4. Main Game Flow 29](#_Toc402383980)

[5.3. Object and Class Model 31](#_Toc402383981)

[6. USER INTERFACE 33](#_Toc402383982)

[6.1. Navigational Path 33](#_Toc402383983)

[6.2. Screen Mock-ups 34](#_Toc402383984)

[6.2.1. Main Menu 34](#_Toc402383985)

[6.2.2. Level Selection 35](#_Toc402383986)

[6.2.3. View Help 36](#_Toc402383987)

[6.2.4. Change Settings 37](#_Toc402383988)

[6.2.5. High Scores 38](#_Toc402383989)

[6.2.6. View Credits 39](#_Toc402383990)

[6.3. Game Map 40](#_Toc402383991)

[6.4. Game Objects 41](#_Toc402383992)

[7. REFERENCES 42](#_Toc402383993)

# 1. INTRODUCTION

“Conquerors” is an arcade game which is inspired from the Atari game “*Volfied*”. The aim of this game is to conquer the castle by drawing lines before time runs out. However do not think it will be easy. During the game you will face with different enemies that will obstruct our hero to reach the goal. Following you will find the *YouTube* video of the influenced game to understand how it will work better.

<http://www.youtube.com/watch?v=RxuMVqkLD7o>

The game which we plan to develop will have many different features than the original game. Unlike the original game, there will be various bonuses, -both positive and negative, and the user will be allowed to start the game from the desired level using special codes which will be given to users when they start that level. Moreover you will be able to choose your own hero from selection of three.

Our purpose of creating that game is to make a project which will allow us to use Object-Oriented Software Programming and general programming techniques. During that process we will create an arcade game which can be played by any person who has beginner (or more) knowledge of computers. In the game there will be many different objects, levels, bonuses and time factor. Those factors make that game a good choice to work on for our course since our goal is to learn the principles of Object-Oriented Software Engineering.

In this report the overview of game, principles of the game, requirements, use-case’s models including scenarios and use-case diagrams will be explained.

# GAME OVERVIEW

The game “Conquerors” has a purpose of conquering the land where castle stands before time runs out. Similar to its ancestor the lands are occupied by drawing lines around them. After user completes the drawing process, that land will be conquered. The drawing line process is made by the “hero” (see section 2.1. Hero). The name hero is chosen since it is the protagonist character of our game. The game has a military theme so we give names to objects according to this factor.

The game is controlled by the arrows on the key board. You move the hero according to them. Moreover space bar is used to fire a bullet (see section 2.4. Bonuses).

The game has 5 different levels which will be harder than the previous one (see section 2.5. Levels). There is an option of selection of levels for the user. However to start from the level you want, you should know its code. You learn the codes of desired level when you unlock that level. Thus if user wants to start with level 3, s/he must play that game before and at least come to the level 3 and got the code. That selection is allowed at the beginning of the game. You will press the “Level Selection” button. Then you press the desired level and enter the code into the box. If it is true then you can start from that level.

The score system is based on the time. At each level different time intervals are given. After one passes a level, time left is taken and multiplied by the coefficient which is different for each level (see 2.5. Levels). You earn points for each level and that point is added to the previous level(s) points. For example you pass level 2 and you had 15 points from level 1 and you got 10 seconds left. That 10 will be multiplied by 2 which is 20 and your final point will be 35. However you can win or lose point by bonuses (see section 2.4. Bonuses). Those points will be added or subtracted from your point at the end of the level. For instance the user from the previous example got one point plus bonus. That means that his final 35 point will be 45 now. Furthermore, high-score list consists of 5 highest scores of all time.

To win the game you should pass all 5 levels. If time runs out, before conquering the castle, game will end. Moreover when any of the enemy will touch to the hero or the unfinished lines the game will be over too (game over criteria for each enemy is given at section 2.3. Enemies).

## Hero

The hero is the object that user controls in the game with the keyboard (arrows for the movement and space bar for firing a bullet). User is allowed to choose his/her hero from three options: a sword, a shield or a helmet. Selection of hero is allowed at the beginning of the game. To choose the hero you should click the “Settings” button and then user can see the options. Hero will perform the drawing lines process. At the beginning of each level, hero will locate at the leftmost bottom corner of the screen. It cannot move directly to the other corners which are at the top. You should first conquer bottom parts to move up, which makes the game harder.

## Castle

The castle object is the key for passing a level or winning the game. When hero conquers the land of the castle user wins. Its location is fixed for each level. When you go to next level its location changes and it will locate a place which is harder to conquer. We decided to put it in the middle of the map at the first level. Then we will move it up a little for each level and at level 5 it will be at the top (see section 2.5. Levels).

## Enemies

We decided to create different objects as enemies to make our game harder and more fun. There are four different enemy types which will act differently.

### Soldiers

Soldiers are the primer enemies in our game. Since we are making a game which has military elements, we think that we should give this name to one of the enemies. Those soldiers will be in the game in all of the levels (see section 2.5. Levels). They will move randomly on the unconquered lands but they are not allowed to pass the occupied lands. In order to get rid of them, you can shoot them if you have fire bonus (see section 2.4.1. Power-ups) or you can conquer the land they are walking on by drawing lines around that place. When your hero or the line you are drawing will touch one of them you will die if you did not get any extra life bonuses (see section 2.4.1. Power-ups). If you have that bonus you will lose one of your lives.

### Cannonball

Like soldiers cannonball objects are inspired by the military elements. They are not able to move but they will throw balls which will move in a horizontal line. They will throw a ball every 5 seconds. They will locate at the rightmost or leftmost places in the screen. Their amount will change according to the level (see section 2.5. Levels). They are fatal just like soldiers. You can destroy them by occupying the land that cannonball sits or by shooting them with fire bonus.

### Walls

Walls will prevent our hero move freely. They will be the obstacles on its way. They will be placed in random places and they are not able to move. Their existence will differ from one level to another (see section 2.5. Levels). Touching them will make hero to lose its life. User can break them by shooting if you have fire bonus or user can conquer the area they will stand. In level 4, you need to shoot them once to break them and in level 5, 2 shoots are required for destroying them.

### Mine

Mine is the worst enemy of our hero. It cannot be destroyed with bonuses other than enemies gone bonus (see section 2.4.1. Power-ups). Its position will be fixed (near to the castle). If you think conquering the land, which mine stays, will destroy it, you are wrong. When you occupy that land, you will lose the game and having life bonuses does not matter. Thus user needs to run from the mine if s/he wants to win.

## Bonuses

We decided to have both positive and negative bonuses in our game. As levels go up their ratio of positive to negative will change (see section 2.5. Levels). 10 bonuses will appear at each game. They will come up at a random time of the game and will appear at only 8 seconds. Two or more bonuses can appear at the same time, there is no limitation for that. Their location will be random too. They will be in treasure chests. If user wants to get them their location should be occupied. After that, user can get either a power-up (positive bonus) or a bad surprise (negative bonus). It will be the user’s luck.

### Power-ups

* **Life Bonus:** As it can be understood from its name, it will add a life to our hero. If they hit an enemy they will not die instantly thanks to that bonus.
* **Bigger Hero:** When user gets that bonus, the hero’s size will be bigger. That feature will allow it to conquer more lands easily.
* **Faster Hero:** That feature will help our hero to move faster, which will help it to run away from enemies easily and conquer lands in a smaller time interval.
* **Time Plus:** It will increase the game time by 5 seconds.
* **Point Plus:** It will increase user’s point by 10 points.
* **Enemies Gone:** All enemies are destroyed when this bonus is taken.

### Bad surprises

* **Smaller Hero:** That bonus will decrease the size of the hero which will make it harder to conquer lands.
* **Time Minus:** It will decrease the game time by 5 seconds.
* **Point Minus:** It will decrease user’s point by 10 points.
* **More Walls:** After getting that bonus, suddenly walls will be built around the castle. When walls are built, the land which should be occupied to pass the level will be bigger. That will make hero’s work harder.

## Levels

As mentioned before the game will consists of 5 levels. Each level will be harder than the previous one. All of them have the same purpose: conquering castle’s land.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **LEVEL 1** | **LEVEL 2** | **LEVEL 3** | **LEVEL 4** | **LEVEL 5** |
| **GAME LENGTH** | 210 seconds | 180 seconds | 150 seconds | 120 seconds | 90 seconds |
| **AMOUNT OF SOLDIER** | 2 | 2 | 2 | 3 | 3 |
| **AMOUNT OF CANNON** | 0 | 1 | 2 | 2 | 2 |
| **AMOUNT OF WALLS** | 0 | 0 | 0 | 3 | 3 |
| **AMOUNT OF MINE** | 0 | 0 | 0 | 0 | 1 |
| **AMOUNT OF POWER-UP** | 8 | 7 | 6 | 5 | 4 |
| **AMOUNT OF BAD SURPRISES** | 2 | 3 | 4 | 5 | 6 |
| **COEFFICIENT FOR FINAL POINTS** | X1 | X2 | X3 | X4 | X5 |

**Table 1:** Table which shows the properties of all of the levels

# 

# 3. FUNCTIONAL REQUIREMENTS

## 3.1. Play Game

The player will be able to use the arrow keys to draw lines, trying to conquer the lands in a given time interval. There will be random power-ups and bad surprises which makes the game easier / harder. When any of the enemy will touch to the hero or the unfinished lines the game will be over. The player will be able to change the appearance of the hero from the settings screen.

If the player conquers the castle in given time successfully, s/he will have access to the next level. In order to win the game, the player has to complete 5 levels which have different types of enemies and objects which makes the game harder. These extra features makes game much more challenging and enjoyable. However, having good time is not only positive effect of this game. The game “Conquerors” also increases hand-eye coordination of a person.

## 3.2. View Credits

The player will be able to reach the game developers’ contact information in this section. Therefore the player will have a chance to communicate with the developers and most importantly help them to develop game by using their creative ideas.

## 

## 3.3. View Help

The player will be able to get the information about the game with very detailed explanation. These explanations are

* Rules and main purpose of the game.
* Player controls.
* Power ups and power downs that can be gained by Player during game.

The player is given chances to learn rules and game controls. When the player learns and uses all these information during game, he will have much more fun.

## 3.4. Change Settings

The player will be able to change some of the game settings according to his own demands. These are

1-sound on/off

2-hero type

The player will be able to turn off or turn on sound of game from Change Settings. Additionally, the player is provided by 3 choices for hero types. Player can select hero type before starting game from Change Settings.

## 3.5. High Scores

The player is able to see highest 5 scores of all time are recorded in the High Score List with ranked player names.

## 3.6. Level Selection

The game will consist of 5 levels. Each level will be harder than the previous one. All of them have the same purpose: conquering castle’s land. From level 1 to 5, time and power-ups are decreased and number of enemy and power-downs are increased. In addition, if player finishes the level, before the next level vision comes, the code appears which is used for skipping levels.

# 4. NON-FUNCTIONAL REQUIREMENTS

## 4.1. Game Performance

The Conquerors game will work as high as possible in terms of performance. As the game has some dynamic displays, sound and some objects that move randomly, we want these features not to effect speed of the game. In addition, system requirements will be kept as low possible since it is new version of old fashioned game.

## 4.2. Graphical Smoothness

We will try to make our game work good and look good at the same time. There are a lot of smooth dynamic graphics such as death of enemies and movement of object randomly. We plan to keep these graphic as smooth as possible.

## 4.3. User Friendly Interface

Playing and understanding shouldn’t be very hard for players. And most importantly feeling comfortable during playing the game is one of our goals. Therefore the game will include very friendly and understandable interface.

# 5. SYSTEM MODELS

## 5.1. Use Case Model

This section is to give detailed information about the use cases of the Conquerors Game as well as to demonstrate the UML Use Case Diagram that represents the capabilities of this game system.

**Use Case1:**

**Use Case Name:** View Help  
  
**Primary Actor:** Player  
  
**Stakeholders and Interests:**  
- Player can learn how to play game.  
- System shows an image that explains the purpose of game, controls of game, bonuses and enemies.  
  
**Pre-conditions:** Player should be in Main Menu.  
**Post-condition:** -  
  
**Entry Condition:** Player selects button “View Help” from Main Menu.  
**Exit Condition:** Player selects button “Back” to return master page.  
  
**The Main Success Scenario:**

1. Player selects button “View Help” from Main Menu.  
2. System displays an image giving instructions about controls, bonuses and enemies of the game.

**Alternative Scenarios:**

A. If player wants to return Main Menu; player selects “Back” button to return main menu.

**Use Case 2:**

**Use Case Name:** Change Game Settings  
  
**Primary Actor**: Player  
  
**Stakeholders and Interests**:  
-Player wants to change game settings that are disabling or enabling game sound and changing the hero type.  
-System updates the new settings which are changed by player.  
  
**Pre-condition:** Player should be in Main Menu.   
**Post-condition:** Game settings are updated.

**Entry Condition:** Player selects button “Change Settings” from Main Menu.  
**Exit Condition:** Player chooses button “Back” to return Main Menu.  
  
**The Main Success Scenario:**

1. Player selects button “Change Game Settings” from Main Menu.  
2. Game settings are displayed by the system.  
3. Player arranges the settings.  
4. System updates the game settings accordingly.

**Alternative Scenarios:**

A. If player wants to return Main Menu; player selects “Back” button to return main menu.

**Use Case 3:**

**Use Case Name:** Play Game

**Primary Actor:** Player  
  
**Stakeholders and Interests:**   
- System aims to provide an entertaining game environment with visually enriched graphics for the player. Besides that the game system calculates the score of the player till the end of game in order to evaluate player’s score for list of the high scores.  
- Player wants to spent enjoyable time while playing the game and aims to successfully finish all levels and rank among the high scores list.  
  
**Pre-condition:** Unless the player changes the game settings, game starts with the predefined features which are image of the hero and game music. If player changes the game settings, the system will accordingly adjust these settings and then starts the game with these new features.  
**Post-condition:** System records the score of the player among the high scores if the player gets a sufficient score, which is higher than one or more scores in the list, from the game.  
  
**Entry Condition:** Player chooses the option of ‘Play Game’ from master page.

**Exit Condition:**

1. Player chooses the option of ‘Main Menu’ from the game screen.

2. Player loses all lives in any level.  
 3. Player chooses to click ‘Exit’ button at any time in the game.

**The Main Success Scenario:**

1. Player chooses the option of ‘Play Game’ from the ‘Main Menu’ screen.  
2. System initiates a new game starting from the first level.

3. When each level is successfully completed system displays a pop-up screen including a code of that recently passed level in order to enable player to start from that level instead of beginning from level 1 when the game is over.   
4. Player passes all of five levels by conquering the castle in each level.  
5. System calculates and demonstrates the final score of the player on a new pop-up page:

a. If the player’s score is high enough to rank among the high scores list, system asks player for entering his/her name, then records that name and score in the high scores list .   
 b. If the player’s score is not high enough to rank among the high scores list, system returns to the master page of the game when the button of ‘OK’ is clicked by the player.

**Alternative Scenarios:**

A. While the game is running, player loses all lives by getting a hit from an enemy, then the system displays a pop-up page that tells to player that the game is over and returns to the master page of the game when the button of ‘OK’ is clicked by the player.

B. In the stream of the game player conquerors the land which is including mine, then system displays a pop-up page that tells to player that the game is over and returns to the master page of the game when the button of ‘OK’ is clicked by the player.

C. At any time player clicks on the button of ‘Exit’ while the game is running, system automatically closes down the game.

**Use Case 4:**

**Use Case Name:** Select Level  
  
**Primary Actor**: Player

**Stakeholders and Interests:**  
-Player wants to play any desired level of the game.  
-System starts the desired level.  
  
**Pre-conditions:** Game always starts from first level. If player enters the code of the desired level, game starts from that level.  
**Post-condition:** Player must know the code of desired level which is given by the system.  
  
**Entry Condition:** Player chooses “Select Level” from Main Menu.  
**Exit Condition:** Player selects “Back” to return Main Menu.  
  
**The Main Success Scenario:**   
  
1. Player clicks the button “Select Level” from the Main Menu

2. System asks for a level code and a particular level.

3. System starts the game from the desired level.

**Alternative Scenarios:**

A. If player wants to return main menu at any time, player selects “Back” button to return Main Menu.

B. If given code is not appropriate for desired level; system displays a warning message pop-up in order to inform the player.

**Use Case 5:**

**Use Case Name:** View High Scores  
  
**Primary Actor**: Player

**Stakeholders and Interests:**  
-Player wants to see top scores with names.  
-System shows the list containing top five scores with names.  
  
**Pre-conditions:** System registers top five scores.  
**Post-condition: -**  
  
**Entry Condition:** Player selects “View High Scores” from Main Menu.  
**Exit Condition:** Player selects “Back” to return Main Menu.  
  
**The Main Success Scenario:**

1. Player clicks the button ‘View High Scores’ from the Main Menu screen.  
2. System displays top five scores with names.  
  
**Alternative Scenarios:**

A. If player wants to return Main Menu; player selects “Back” button to return main menu.

**Use Case 6:**

**Use Case Name:** View Credits  
  
**Primary Actor**: Player

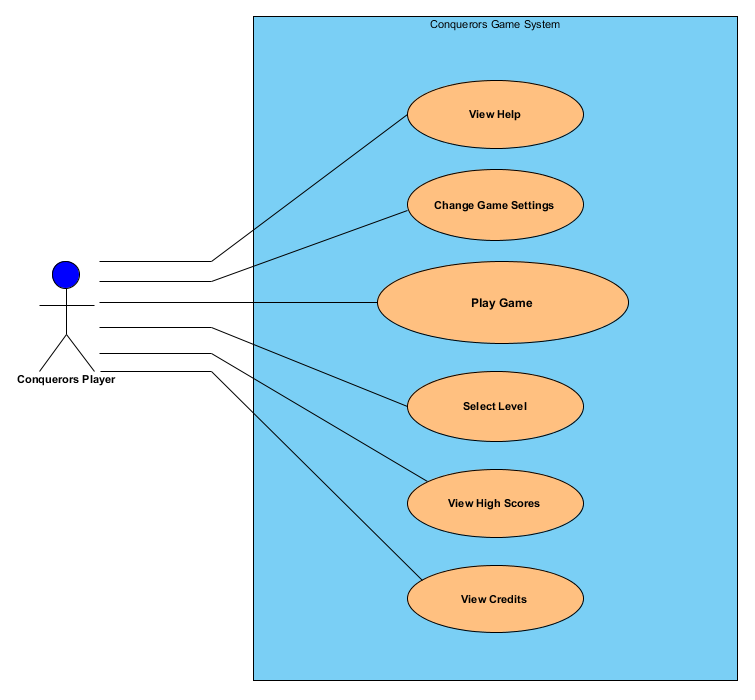
**Stakeholders and Interests**:  
- Player wants to learn names who developed the game and contact information.  
- System shows the developers contact information.  
  
**Pre-conditions**: Player should be in Main Menu.  
**Post-condition**: -  
  
**Entry Condition**: Player chooses button “View Credits” from Main Menu.  
**Exit Condition**: Player selects button “Back” to return master page.  
  
**The Main Success Scenario:**

1. Player chooses button “View Credits” from Main Menu.

2. System displays contact information of developers of Conquerors.  
  
**Alternative Scenarios:**

A. If player wants to return Main Menu; player selects “Back” button to return that menu.

The following diagram illustrates the indicated use cases in detail above as a summary.



**Figure 1:** A UML Diagram of Use Cases made by Visual Paradigm

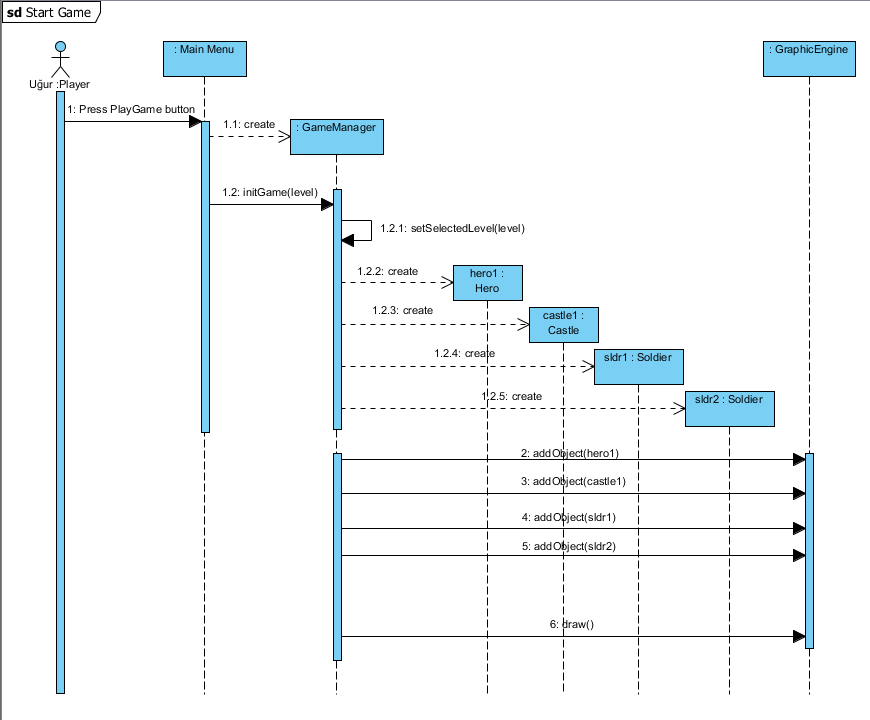
## 5.2. Dynamic Models

In this part we will talk about dynamic models of the game.

### 5.2.1. Play Game

In order to start the game “Play Game” button should be chosen to execute the game. If any level selection is not made then level 1 is opened by the system. System reads the object distribution from a text file and places the object according to that. For each level objects’ amount and places differ. However for all levels hero starts from the leftmost bottom corner of the window and timer is at the same place . Lastly, system enters the game loop in order to continuously update the game. In order to show that process we aim to create a scenario in which the player starts the game at level 1 and we will tell what he saw on the screen.

**Scenario 1:** Player Uğur opens the game. He clicks on Play Game button which will make him go to level 1. After he clicked that button system creates a window and the map. He sees the entire objects; a hero at the left bottom corner, a castle at the middle of the map and two soldiers in the map.



**Figure 2:** Sequence Diagram of Start Game

### 5.2.2. Bonuses

Bonuses will appear at the random places at random times. Where they will locate will be determined from the text file too. We aim to talk about how bonuses can be gained by player and tell how some bonuses affect the game. That’s why we first create a scenario for telling how to get a bonus, another one to talk about “life” bonus and another for “enemies gone” bonus. We thought that those two bonuses will be important to show game process.

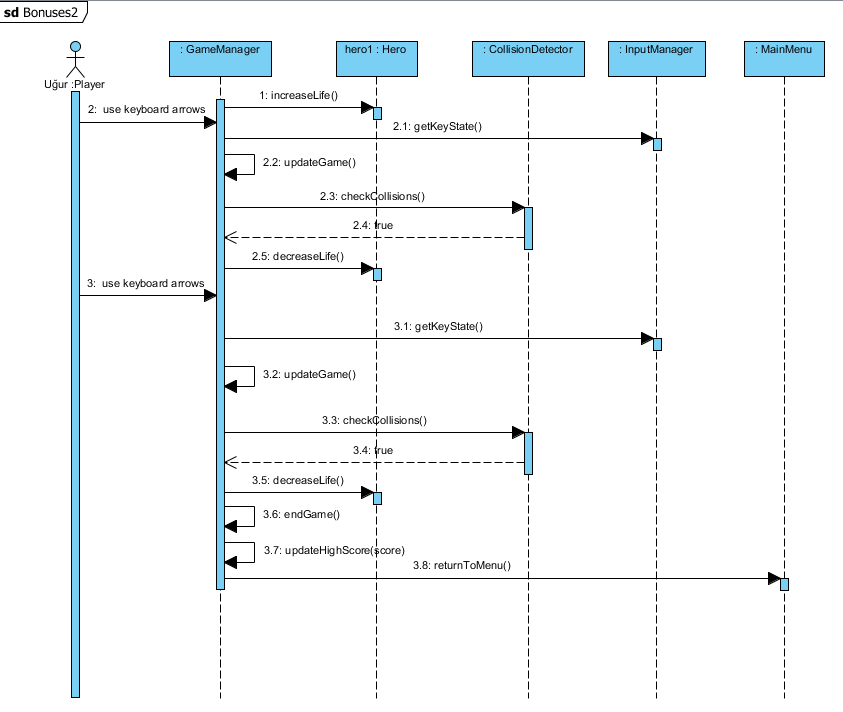
**Scenario 1**: After executing the game (still level 1), Uğur sees a bonus objects on the screen and decides to get it. Using keyboard arrows Uğur draw lines around the bonus object and conquers that area. Bonus is got by that action.



**Figure 3:** Sequence Diagram of Scenario 1

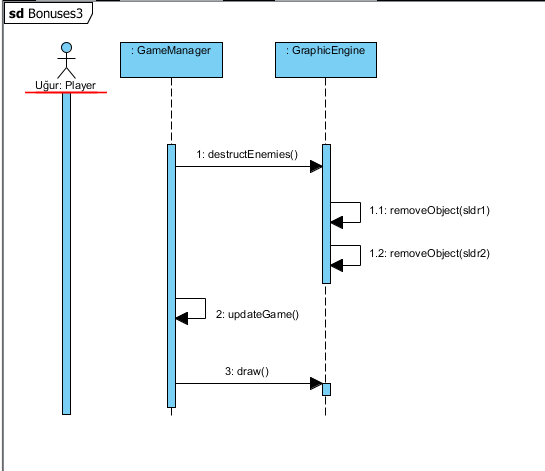
**Scenario 2:** After getting the bonus Uğur sees that it is a life bonus. He sees that “Lives: 0” phrase turned into “Lives : 1”. Then he continues to play. He staH

He starts drawing lines but before finishing the conquering processes the “Hero” collides with a soldier. Since he got a life, game is not over but “Lives: 1” turns into “Lives: 0” again. Then he continues to play and he hits another soldier. Then he sees “Game Over” phrase and the system turns to Main Menu.



**Figure 4:** Sequence Diagram of Scenario 2

**Scenario 3:** After getting the bonus Uğur realizes that no enemy exists on the screen. Suddenly all of the soldiers are deleted. He gets that that is the “Enemies Gone” bonus.

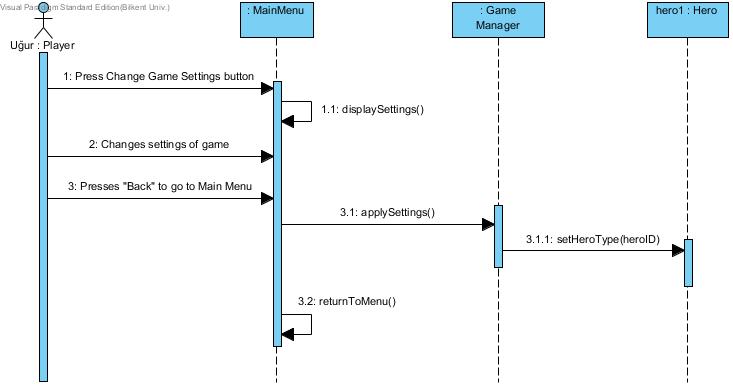


**Figure 5:** Sequence Diagram of Scenario 3

### 5.2.3. Change Settings

In order to change the game settings “Change Settings” button should be chosen from the Main Menu. In there the user is allowed to change the hero type and make sound on and off.

**Scenario 1:** Uğur clicks on “Change Settings” button from Main Menu. He sees Hero types. He decides to change the hero. He sees a sword image, a helmet image and a shield image. Then he clicks on “helmet image” so his new hero is “helmet”. Then he clicks to “Back” button so Main Menu is shown.

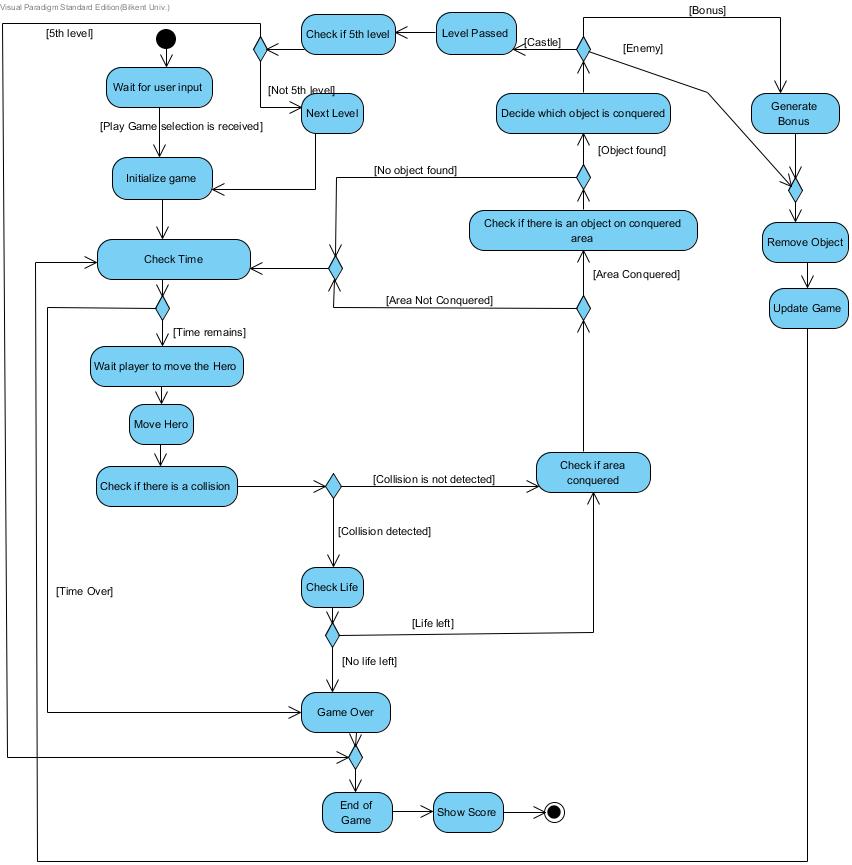


**Figure 6:** Sequence Diagram of Change Settings

**Description of process:** After player finishes changing hero process, GameManger sends hero type to Hero class. Then new hero will be set. The desired event is successfully accomplished.

### 5.2.4. Main Game Flow

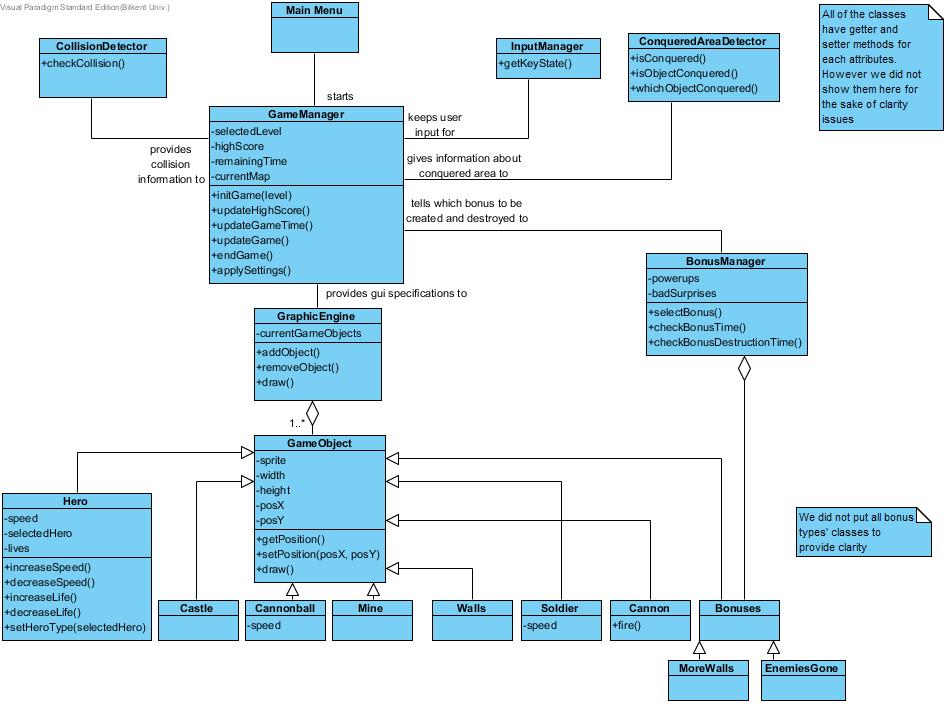
It is decided that an activity diagram should be drawn to illustrate the main game flow. The activity diagram is shown below.



**Figure 7:** Activity Diagram of Flow of Game Events

The game will open the Main Menu at the beginning and the system will wait to start the game until “Play Game” button is pressed. After that, game will be initialized. The objects will properly set. Game GUI will be created. The system will check whether there is time left or not. If no time left, that means game is over. However if there is time, it will wait player input then move the hero, and then checks a collision. If hero hits an enemy while moving then it needs to check the life status. If player has 0 lives then game is over. On the other hand if player has 1 or more lives it will continue to move. If movement is successful then system will check if the area conquering process is finished. If the answer is no, the system goes back to check time and does the same things. However if the area is conquered then it will check whether there is an object inside or not. If no object is detected by the system then it will simply go back to waiting for new input phase too. When an object is found, the system will decide what kind of object lays there. If it is a castle, it means that level is completed. Then system has to check if it is the 5th level or not. If it is not the 5th one then the system will turn to initialize game phase as set the game according to next level’s properties. If it is the 5th level then it is the end of the game. System shows the score and exits. On the other hand if the conquered object is an enemy then the game will remove it and update the game then goes back to check time phase. However, if the object is a bonus object then the system will generates the bonus and deletes the object from game map and goes back to check time phase. When game is over, the system will show the score and exits. This is the main flow of events in the game.

## 5.3. Object and Class Model



**Figure 8:** Object and Class Model of the game *Conqueror*

The class diagram of the game Conquerors are illustrated above. In this phase we decided to make the class hierarchy as shown above.

**MainMenu** class is the parent of GameManager class. This class is designed for managing the “Main Menu” of the game.

**GameManager** class has a purpose of organizing and managing the game dynamics by working with other classes.

**CollisionDetector** is the class in which we check if hero collides with any enemy. We need that since when an enemy hits the hero game will be over. This class provides collision information to GameManager.

**ConqueredAreaDetector** class does three main things. First of all it checks whether hero completed the conquering process or not. Secondly (if the area is conquered) it checks whether there is an object under that area or not. Lastly (if there is an object) it decides what kind of object lays there. This class provides all of that information to GameManager.

**InputManager** simply keeps user input for GameManager.

**BonusManager** class is designed for managing bonus dynamics of the game. It will decide which bonuses will be chosen for each level. It will keep track for how long they will exist, when they should be created and when they should be destroyed. It will use “Bonus” class’ objects. It will reports those facts to GameManager.

**GraphicEngine** will provide GUI specifications to GameManager. It will tell it to add or remove an object and it has a draw() method in it.

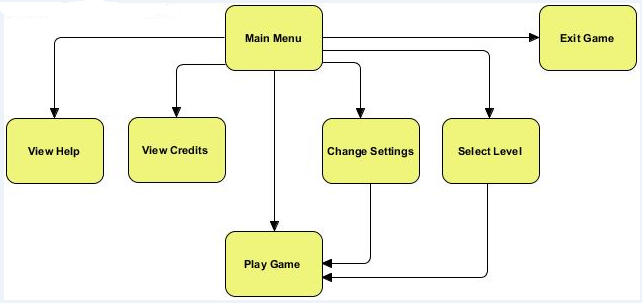
**GameObject** class is the parent of classes shown above. It will keep information about their position, sprite, width and height of all objects in the game. It is responsible for getting and setting their position too.

All objects will have different characteristics of their own as seen from the diagram. We decided to make a Bonus class which will be the parent of all bonus types so that we will create a polymorphism there too.

We tried to make that as neat and clear as possible. That’s why we did not put all the getter setter methods and did not put all of the bonus types’ classes.

# 6. USER INTERFACE

## 6.1. Navigational Path



**Figure 9:** Navigational Path of the game

When the game is opened, first thing that system opens is Main Menu. The player can choose View Help, View Credits, Change Settings, Select Level, Play Game and Exit Game. The actor can go to Play Game section from Change Settings and Select Level sections too.

## 6.2. Screen Mock-ups

All of the mock-ups are designed by developer, only background image is taken1.

### 6.2.1. Main Menu

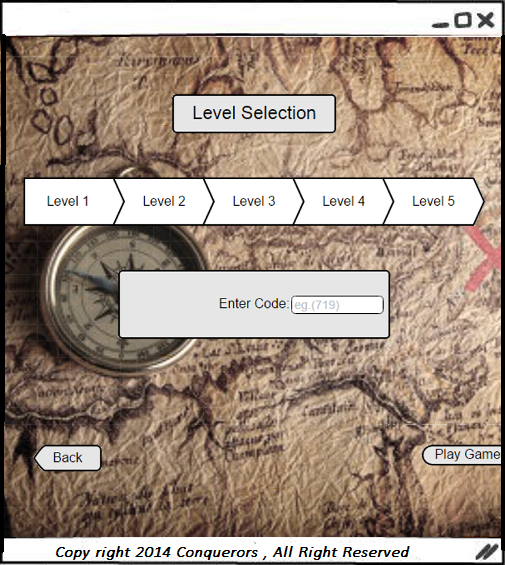
When the system starts, main menu screen appears. Player can start the game, change settings, view credits and help, select level.



**Figure 10:** Screen Mock-up of Main Menu

### 6.2.2. Level Selection

If the player chooses Level Selection button, the player should click on the desired level button. Then the player should write the code to the box shown below to activate the desired level. The player can go to Play Game Section from this section.



**Figure 11:** Screen Mock-up of Level Selection

### 6.2.3. View Help

If player clicks the button View Help, explanation of the game and the objects are shown on the screen. Player is suggested to click here in order to feel comfortable during the game.



**Figure 12:** Screen Mock-up of View Help

### 6.2.4. Change Settings

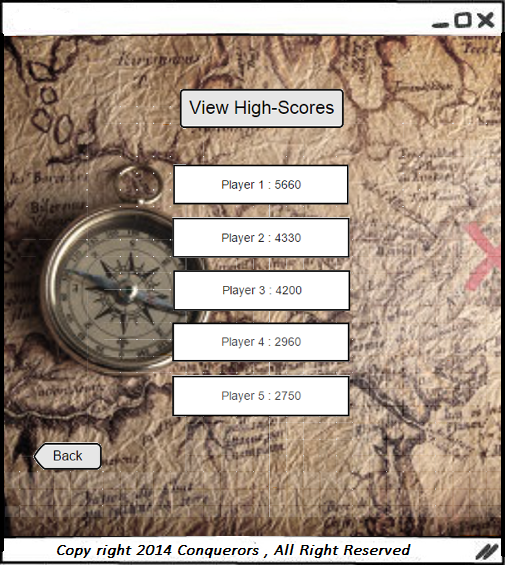
Player can select the hero type and make sound on/off from here. If player does not make any change in settings, the default settings are used by system. Player can also start the game from here.



**Figure 13:** Screen Mock-up of Change Settings

### 6.2.5. High Scores

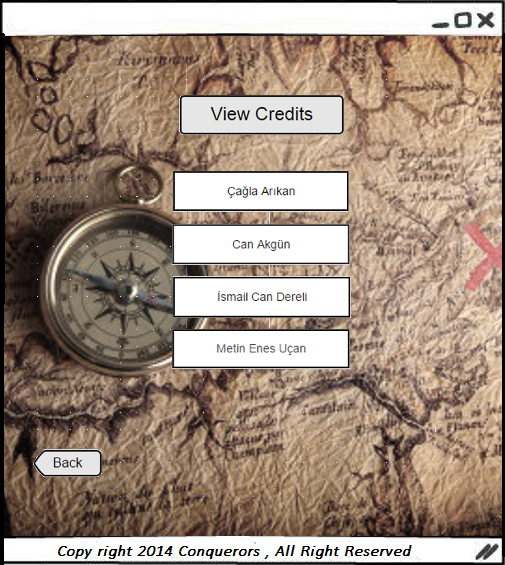
When high score button is pressed, the system shows the highest 5 scores of players. Also if a player can beat the one of the high-score, s/he will be in the list.



**Figure 14:** Screen Mock-up of High Scores

### 6.2.6. View Credits

In View Credits, player can find the contact information and names of the game developers.



**Figure 15:** Screen Mock-up of View Credits

## 6.3. Game Map

The mock-up below illustrates an example of game map.



**Figure 16:** Screen Mock-up of Game Map

## 6.4. Game Objects

 Cannonball2

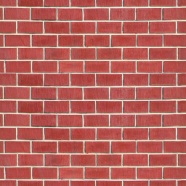
 Castle3

 Bonuses4

 Shield5

 Helmet6

 Sword7

 Wall8

 Soldier9

C:\Users\Mettobil\Desktop\319images\mine1.png Mine10

# 7. REFERENCES

1. http://1.bp.blogspot.com/MrZj7jtTZAg/UaaGzdeku8I/AAAAAAAANPw/gLu\_4Udzvy0/s1600/hazine\_define\_harita.jpg
2. http://graphicriver.net/item/human-cannonball/4307140
3. http://www.shutterstock.com/s/castle/search.html
4. http://www.shutterstock.com/pic-112253219/stock-vector-a-treasure-chest-on-a-white-background-vector-illustration.html
5. http://pixgood.com/cartoon-shield.html
6. http://clubpenguin.wikia.com/wiki/Viking\_Helmet?file=RedVikingHelmet.png
7. http://www.clker.com/clipart-6800.html
8. http://www.battlefieldheroes.com/en/forum/showthread.php?tid=331058&page=39
9. http://lol-rofl.com/soldier-cartoon/

10- http://4vector.com/free-vector/cartoon-sea-mine-clip-art-107708