Bilkent University

Department of Computer Engineering

CS319 Term Project

RISK

Analysis Report

Group 2I

Ahmet Kaan Uğuralp – 21803844

Selcen Kaya – 21801731

Alperen Can – 21601740

Şükrü Can Erçoban – 21601437

Selin Kırmacı – 21802177

Instructor: Eray Tüzün

Teaching Assistant(s): Barış Ardıç, Emre Sülün and Elgun Jabrayilzade

Table of Contents

1. Introduction

Many kings tried to conquer the world and be the overall ruler of the Earth for centuries. Some came close but none could actually achieve this goal. Risk is a board game in which the purpose is to achieve world domination and it allows people to achieve the goal in a fun and mostly friendly environment. Every player starts with the same conditions and equal amount of possessions but only the one with the best strategy can achieve the glorious end.

Our project is an interpretation of this board game in a digital environment as a computer game. The game is enhanced with colorful graphics, easy and fun user interfaces and exciting sound effects. It offers the players to have an amazing time with their “enemies” in a friendly competitive environment.

The goal is to implement this game in such a way that it will have an efficient backend as well as an interesting and user-friendly front end design. New and wonderful features will be added which will enhance the entertainment of the original game. We will use Java language to implement the game with object oriented design principles as all of the team members are comfortable with it because of previous courses and projects.

2. Overview

Risk is a multiplayer 2D game which can be player with 2 to 4 players. After starting the executable file, the users will view the main menu where they have the options to start a new game, see how to play the game, quit or see the credits.

2.1 Gameplay and Controls

At the beginning of each game, each player selects unique avatars or emperors, colors and nicknames to represent themselves. Subsequently, system distributes territories and different troops equally to each player. The order of the players is determined by rolling dice, with highest rolling player going first. The players get an extra amount of troops at the start of each turn, depending on the number of regions under their control. Moreover, players are provided additional troops by taking the control of a whole continent, exchanging territory cards, or using the relevant curse card while attacking. During a turn a player can attack a territory adjacent to one of their own. The winner of the war is determined by dice roll between defending and attacking players as well as strength of troops on their territories. Alliance is an important

feature at the attack phase of the game, which enables players to send troops to strengthen the offense or to support the defense of a territory. After the war, fortify phase begins. Players can move troops from one of their territory to another, in order to enhance the strength of another, possibly more important territory. Afterwards, other players take turns in order. Ultimately, the game ends when a player wins by conquering all territories in the map.

The game is controlled by mouse. Players will be able to click on a territory to attack that territory or to strengthen territories at the fortify phase. Additionally, exchanging territory cards, using curse cards and accepting or refusing alliance offers are performed by clicking the relevant buttons presented on pop-up screens by the mouse.

2.2 Map

The game will be played on a map divided into continents which are then divided into territories. Players can attack enemy territories or move troops to their own territories, if there is a border between these regions. The map consists of six continents, formed by different numbers of territories. Taking control of an entire continent provides additional troops to player at the beginning of every turn. Number of the additional troops obtained varies for different continents depending on their size.

2.3 Troops

The different troop types in the game consist of infantry, cavalry and artillery. The troops located on a territory represent the strength of that territory. To clarify, an infantry is equivalent to one unit strength, where the cavalry equals to five and the artillery corresponds to 10 unit strengths. The players lose some amount of troops after each war depending on their dice roll.

2.4 Cards

The cards provide specific advantages to players. Cards are evenly distributed to players in the beginning of the game. They can also be obtained during the game with special conditions such as conquering a whole continent or capturing a special territory, the location of which changes for each game.

2.4.1 Territory Cards

If a player successfully captures a territory, a territory card will be gained regardless of the number of the captured territories in a single turn. There are three different territory cards. A player should collect three proper territory cards to form a set, in order to exchange the set with additional troops at the beginning of the turn. The number of troops to be gained depends on the number of former card exchanges:

- 1st set exchange: provides 4 additional troops
- 2nd set exchange: provides 6 additional troops
- 3rd set exchange: provides 8 additional troops
- 4th set exchange: provides 10 additional troops
- 5th set exchange: provides 12 additional troops
- 6th set exchange: provides 15 additional troops

For more than six exchanges, the number of extra troops provided will be increased by five. For example, if total six set of exchanges have already done by players, the player exchanging the 7th set will receive 20 extra troops.

If a player has 6 or more proper territory cards forming a set, the cards must be exchanged at the start of the turn to lower that number to at most 4, as per the board game rules.

2.4.2 Curse Cards

Curse cards are special rare cards that can be used in the attack phase. It provides advantages such as decreasing the number of the defending troops, or prohibiting the enemy's ally to send reinforcements. It might be received by capturing a territory. However, there is a low possibility to obtain one.

2.5 Attack Phase

Players may declare a war to another player's territory through their own territories if these regions are sharing a land border or a sea border. The players can also chose to not attack and skip to the fortify phase.

2.5.1 Alliance

When a player declares war, attacking and defending players can offer to have an alliance to other players, not including each other, if other players share a border with target region. If an alliance offer is accepted, attacking and defending players can receive reinforcement from their allies supplied by nearby territories.

2.5.2 Capturing a Territory

The result of the war is determined by rolling dice. The aggressor can choose to roll one to three dice while attacking, where the defender can roll maximum two dice. The highest dice of players are compared. If the die of the aggressor is higher, the defender loses a unit strength. Similarly, if the die of the defender is higher, then the aggressor loses a unit strength. If more than two dice are rolled, second highest pair is compared as well as the first highest pair. Additionally, if the attacker rolls one or two more dice than the defender, the dice which cannot be compared are ignored.

The war lasts until there are no more troops to fight in that region or until the aggressor decides to retreat from the war. Moreover, it should be noted that the attacking player can use as many units from its region. However, at least one unit has to stay in its territory, that is, the player cannot fully abandon its region. After a territory war ends, the player whose turn it is can declare war on another territory. Attack phase lasts until the aggressor skips to fortify phase or loses its all units, or the defender surrenders the territory.

2.6 Fortify Phase

Fortify phase begins after the attack phase ends. Player whose turn it is may send troops from one of their territories to another one in order to increase the strength of that region. Players can perform only one fortify move in a turn. When a player completes or skips the fortify phase, player's turn ends.

3. Requirements

The functional and non-functional requirements of the project are described below in separate subheadings.

3.1 Functional Requirements

The functional requirements which are starting a new game, viewing how to play page, changing settings and seeing credits are explained below.

3.1.1 Start New Game

When the program is opened, the initial page displays the main menu. “Start New Game” option can be seen on the top of the menu and when the user clicks this button, the game starts after asking the initial conditions and statements. The user is directed to a page which asks the number of players as well as the names, colors and preferred avatars of the players. The desired number of players is selected using a drop-down menu with options 2, 3 or 4. Each player rolls dice to determine the turn order and the ascending orders of score from the dice determines who will start and who will continue. Then the game evenly distributes all of the territories on the map to the players randomly. Afterwards, the armies and the cards are distributed to each player and the game is ready to start.

3.1.2 How to Play

In the main menu and the pause menu during a game, there is a “How to Play” option which displays essential information about the game to the user containing rules of the game, controls and game explanations such as win conditions. This option helps first time players of the game as well as people who played other Risk game implementations by clarifying the differences.

3.1.3 Settings

The user can see the “Settings” option in the main menu and pause menu. This option allows the user to customize sound and music options which can either be adjusted or muted.

3.1.4 Credits

This option can be accessed from the main menu and when the user clicks this button, they see a screen displaying information, such as names, about the developers of the game.

3.2 Non-Functional Requirements

The non-functional requirements are explained in the following subheadings including interface, performance, maintenance and extendibility.

3.2.1 Interface

The user interface is one of the most important features of a game since it can have critical effects on the users even with a first impression. It determines users' involvement and can also improve continuity and cohesion of the game. Thus, to make a user-friendly interface, we will make all the buttons and options distinct and obvious, so accessibility and visibility will improve. In addition, we will design our graphics carefully in terms of intelligibility and tonality. These features make our game more attractive and captivating.

3.2.2 Performance

In the game, we will focus on response time and interface transition time since slow performance in terms of reaction time may push users to quit because of their exhaustion and annoyance. Moreover, to get more performance and for compatibility with every computer, the system requirements will be at the lowest possible level. We also add some features which do not affect the performance such as sound and music, in order to get the highest performance along with the right ambiance and pleasantness.

3.2.3 Extendibility and Maintenance

The class models were designed to easily make adjustments and add new features for any improvement we might want to make in the future. For example, a feature we are planning to add involves certain advantages and disadvantages given to each player depending on their choices at the beginning of the game such as the capital and the emperor they chose. Also, object oriented programming concepts will be used in order to make extendibility convenient. For maintenance, we will pay attention to the implementation and design of our codes since it should be clearly understood by others. In addition, we will try to clear any ambiguous sections by writing coherent comments. Moreover, the game can be improved with artificial intelligence (AI) since with AI, a player can play the game against the computer.

4. System Models

The use-case model and descriptions, object and class diagram, sequence diagrams, activity diagrams and user interface mock-ups are given in their respective subheading below.

4.1 Use Case Model

The use case diagram is given in the figure (Figure 1) and the descriptions of five use-cases are given below it in their respective subheadings.

4.1.1 Use Case Diagram

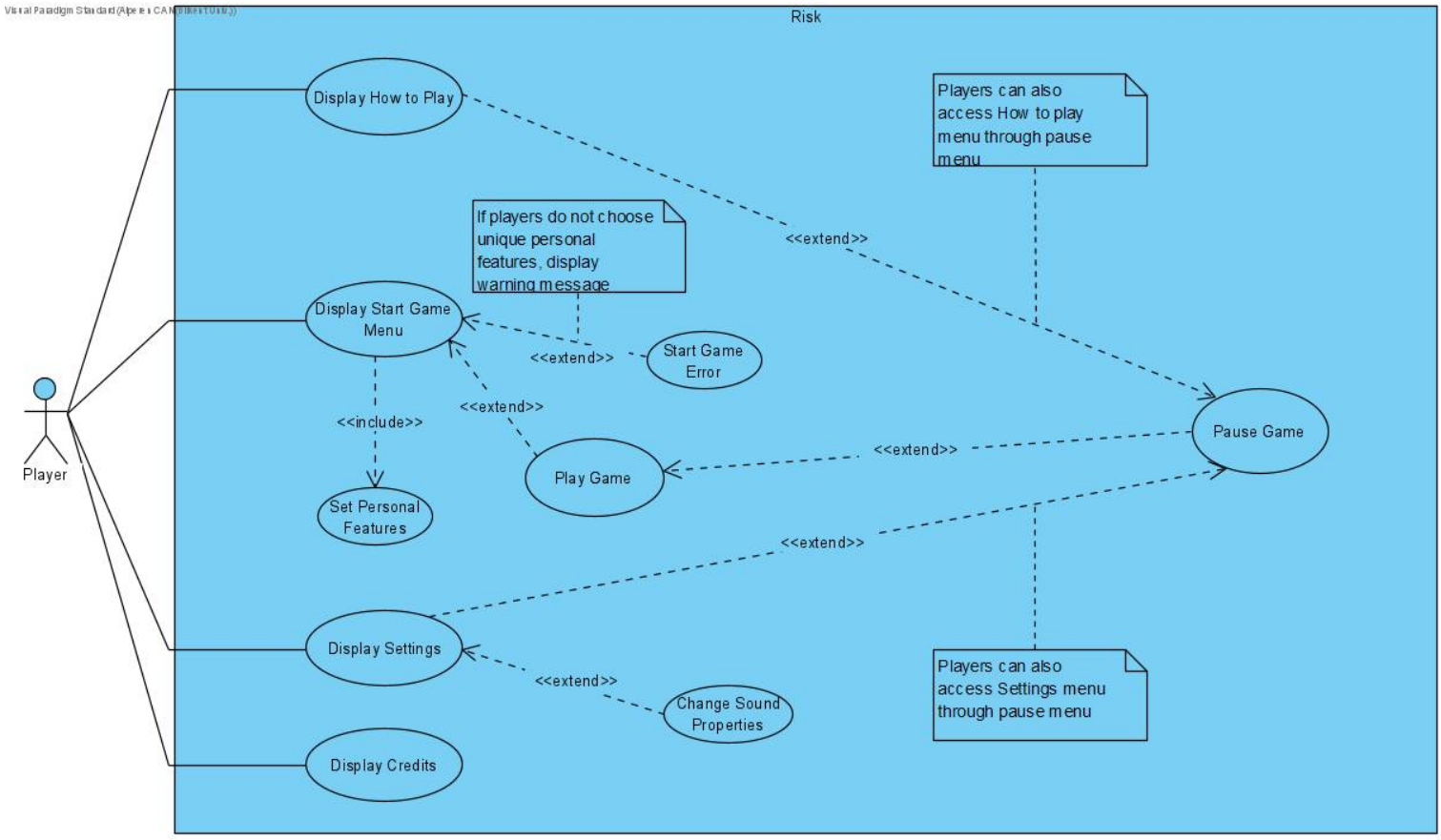

Figure 1: Use case model of the project.

4.1.2 Use Case Descriptions

The five use cases which are named start new game, how to play, settings, credits and pause are described below.

4.1.2.1 Use Case Name: Start New Game

Participating Actor: Players

Stakeholders and Interests: Players want to start the game.

Entry Conditions:

- Players clicks *Start New Game* button.
- Player selects the number of players.
- Players choose avatars, nicknames and colors to represent them.
- Players clicks *Continue* button.

Exit Conditions:

- Player clicks *Quit* button from pause menu, OR
- Player conquers all territories in the map to win the game.

Main Flow of Events:

1. Player select the number of players.
2. Players enter nicknames and choose their avatars and colors.
3. Player starts game.
4. System sets the distribution of the troops and territories for each player.
5. Player whose turn it is adds new troops to their territory(s) at the beginning of the turn.
6. Player may attack other players' territories.
7. Player may move troops to their other territories.
8. Other players take turns.
9. A Player wins the game by conquering all territories.
10. The winner is demonstrated on the screen at the end of the game.

Alternative Flows of Events:

2.1 Players choose unique avatars, nicknames and colors.

a. Player starts game.

2.2 Players do not choose unique avatars, nicknames and colors.

a. Warning message is displayed in order to urge players to choose unique avatars, nicknames and colors.

7.1 Player attacks to other players' territories.

7.1.1 Aggressor and defender players offer alliance to other players.

7.1.1.1 Other players accept the alliance.

7.1.1.1.1 Allies attack and defend territories together by throwing dice.

a. Aggressor conquers the territory.

b. Aggressor fails to conquer the territory.

7.1.1.2 Other players reject the alliance.

7.1.1.2.1. Attacker and defender clash alone by throwing dice.

a. Aggressor conquers the territory.

b. Aggressor fails to conquer the territory.

7.1.2 Aggressor and defender players do not offer alliance to other players.

7.1.2.1 Attacker and defender clash alone by throwing dice.

a. Aggressor conquers the territory.

b. Aggressor fails to conquer the territory.

7.2 Player does not attack other players' territories.

a. Player skips the attack phase to begin fortify phase.

8.1 Player moves troops to their other territories.

a. Player increases the strength of their other territories.

8.2 Player does not move troops to their other territories.

a. Player skips the fortify phase to let other players take turn.

4.1.2.2 Use Case Name: How to Play

Participating actor: Player

Stakeholders and Interests: Player wants to be informed about the game rules and mechanics.

Entry Conditions:

- Player clicks *How to Play* button from main menu, OR
- Player clicks *How to Play* button from pause menu.

Exit Condition:

- Player clicks *Back* button.

Main Flow of Events:

1. Player clicks *How to Play* button.
2. System displays how to play screen.

4.1.2.3 Use Case Name: Settings

Participating actor: Player

Stakeholders and Interests: Player wants to change the music volume and enable/disable sound effects.

Entry Conditions:

- Player clicks *Settings* button from main menu, OR
- Player clicks *Settings* button from pause menu.

Exit Conditions:

- Player clicks *Back* button.

Main Flow of Events:

1. Player clicks *Settings* button.
2. Player adjusts the volume, alters the status of sound effect checkbox.

3. Player clicks *Save* button.

4. Music and sound properties will be changed.

Alternative Flow of Events:

2.1 Player does not change the music and sound options.

a. Music and sound properties remain same.

3.1 Player returns menu without clicking *Save* button.

a. Music and sound properties remain same.

4.1.2.4 Use Case Name: Credits

Participating actor: Player

Stakeholders and Interests: Player wants to know the developers of the game.

Entry Condition:

- Player clicks *Credits* button from main menu.

Exit Condition:

- Player clicks *Back* button.

Main Flow of Events:

1. Player clicks *Credits* button.
2. The contributors of the game are shown on the screen.

4.1.2.5 Use Case Name: Pause

Participating actor: Player

Stakeholders and Interests: Player wants to pause the game.

Pre-condition:

- The game must have started.

Entry Condition:

- Player clicks *Pause* button.

Exit Conditions:

- Player clicks *Continue* button, OR
- Player clicks *Quit* button.

Main Flow of Events:

1. Player clicks Pause button.
2. The game is paused.

4.2 Object and Class Diagram

The object and class diagram is given in the figure (Figure 2) and descriptions of each class is located under it.

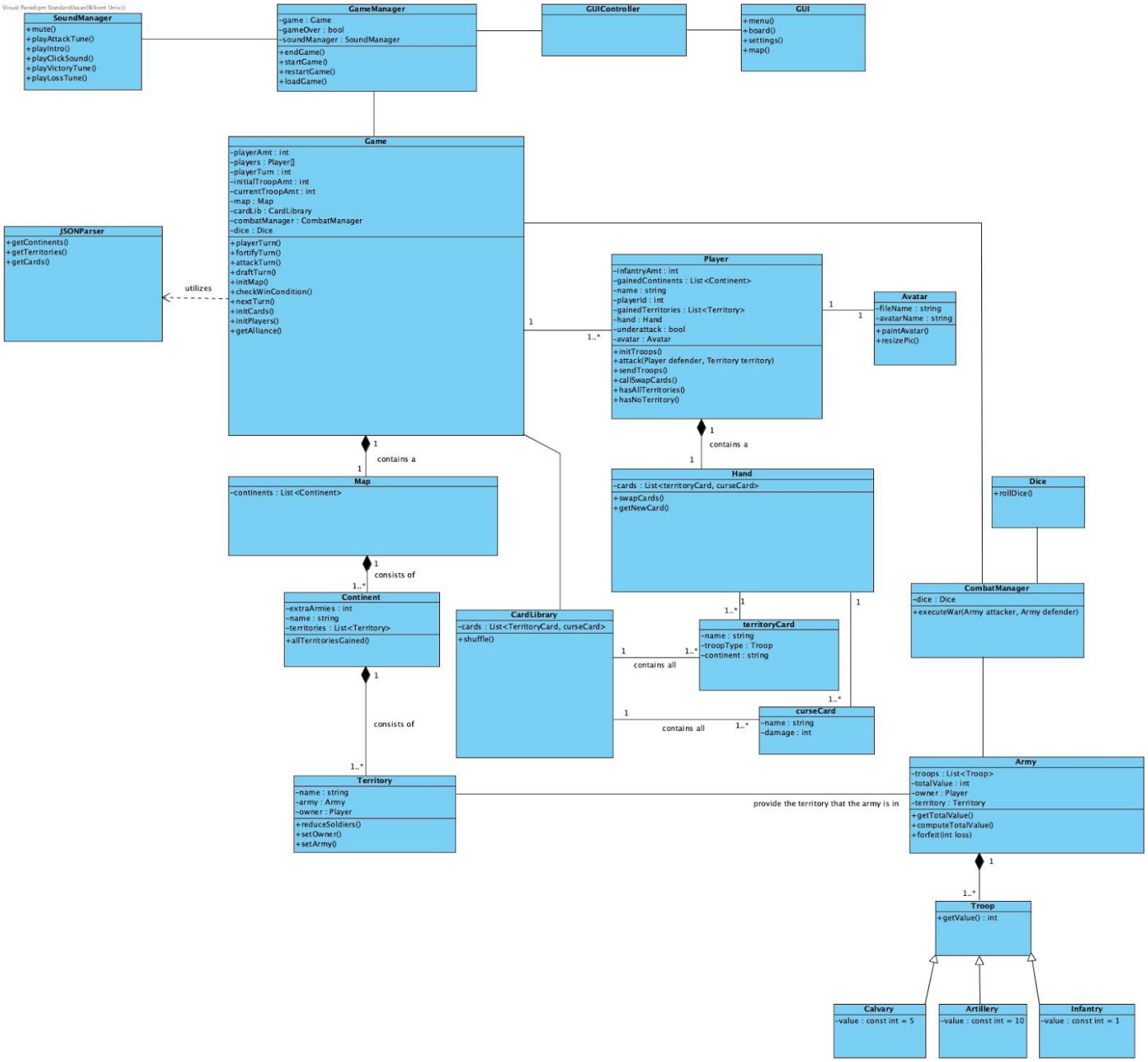

Figure 2: Object and Class diagram of the project.

The descriptions and the contents of the classes given in the diagram above (Figure 2) are as follows:

- **Dice:** Performs roll dice function.
- **Troop:** Parent class of Calvary, Artillery and Infantry. Contains value property.
- **Infantry:** Specialized troop with value of 1.
- **Artillery:** Specialized troop with value of 10 infantries.
- **Calvary:** Specialized troop with value of 5 infantries.
- **Territory:** Has properties army, name and owner.

- **Continent:** Consists of territories.
- **Map:** Consists of continents.
- **Army:** Contains ArrayList of troops of a player in a continent.
- **CombatManager:** Manages attack turns with executeWar function with parameters as attacker and defender armies.
- **Territory Card:** Has a name, troop type and territory.
- **Curse Card:** Has a name and amount of damage.
- **CardLibrary:** Contains all of the cards and is initialized at the start of the game. Has shuffle functionality which will be called at the start of the game.
- **Hand:** Consists of cards. Has swapCards functionality that will notify the player when a successful card combination has occurred.
- **Player:** Contains hand, total infantry amount, gained contents, name, id, gained territories and under attack properties. Has functions such as: troop initialization, attack, send troops, call swap cards, has all territories and has no territory.
- **Game:** Contains all of the game data including player amount, players array, player turn, initial troop amount, current troop amount, map, card library, combat manager and dice. Contains the functions such as: playerTurn which manages each player's turn by using atttackTurn, fortifyTurn and draftTurn functions.
- **JSONParser:** Converts JSON files to objects. These JSON files will contain the required game data such as: continent names, cards and territory names.
- **SoundManager:** Responsible for the in-game sounds.
- **GUIController:** Provides the connections between frontend and backend.
- **GUI:** Provides the frontend of the project such as menus and EventHandlers.

4.3 Sequence Diagrams

The sequence diagrams that refine the use case descriptions are given in subheading below. Playing of a turn is separated from starting a game use case for clarity and intelligibility.

4.3.1 Start New Game

The diagram for this scenario is given in the figure below (Figure 3). The user is in the main menu and clicks the "Start New Game" button which prompts the user to choose the

number of players, which can be between 2 and 4, enter the names, pick the colors and chose the avatars of each player. The player can choose to enter these values or use the preset values to start the game, or click on the “Back” button to go back to the main menu. If the user clicks on the “Continue” button, the GameManager class calls the startGame() function and sets the gameOver Boolean to false. The initialization of the map, cards and players are done by Game class via initMap(), initCards() and initPlayers() functions. The interface including the map, the players and cards are loaded by the GUI class. The order of players are chosen by dice roll and kept in the Player array in Game class. After all initializations are over, each player start to play their turns which is kept out of this diagram for ease of reading and is presented in Figure 4 in the next subheading. After each turn, the Game class checks whether the game is over by the Boolean gameOver returned from playing a turn diagram. If the game is finished, the GameManager class ends the game, otherwise the nextTurn() function is called by the Game class which goes on to playing another turn.

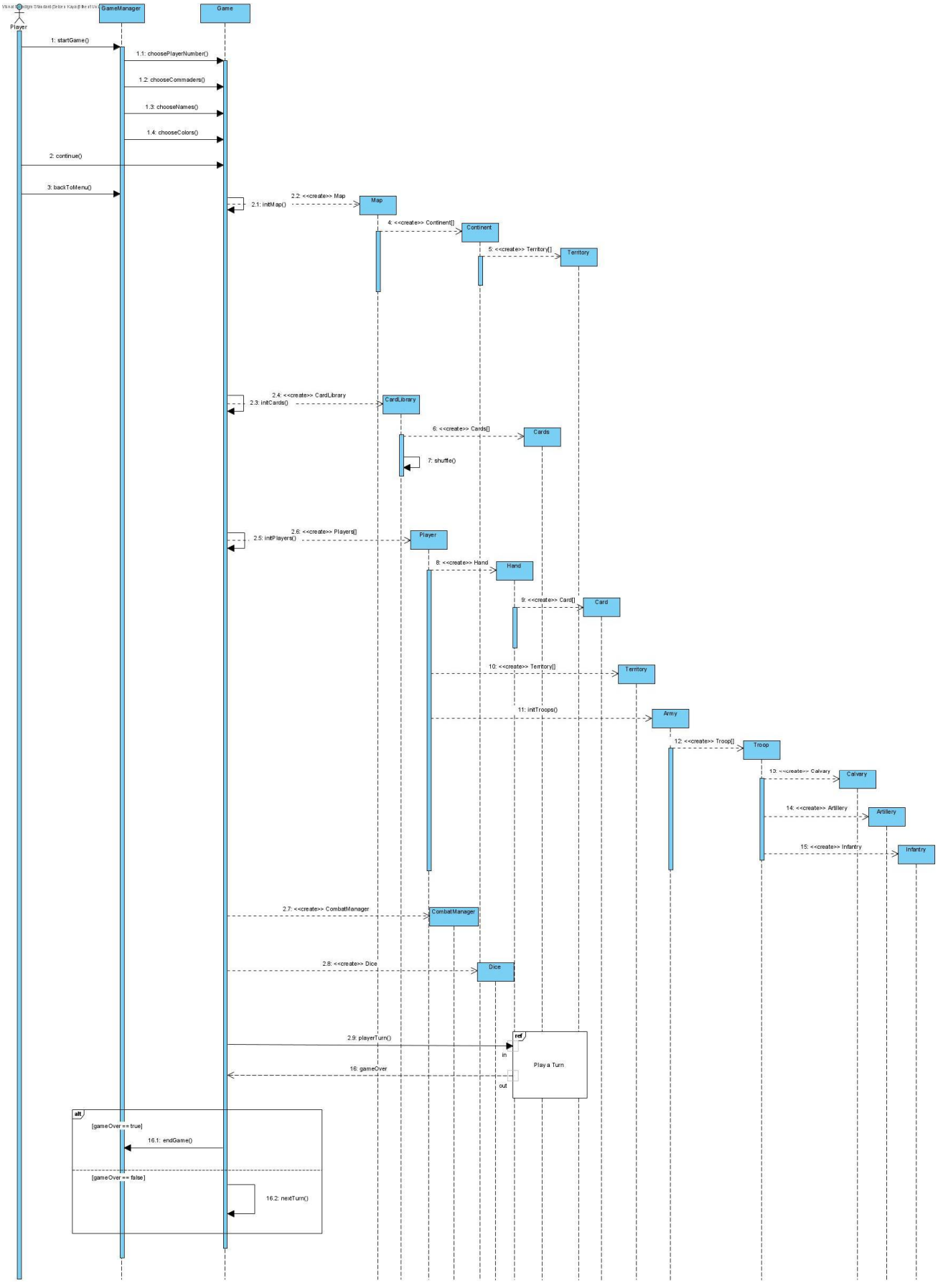

Figure 3: Sequence diagram for Start New Game scenario.

4.3.2 Play a Turn

The diagram is given in Figure 4. The user is given a turn by the playerTurn() function in the Game class. At the beginning of each turn, the player receives new troops depending on their current territories via getNewTroops() functions. The player then has the option to draft their turn, attack an enemy territory, swap their cards for troops or fortify by changing the locations of their troops. After the player has played their turn, checkWinCondition() is called to see whether the game has ended by a player conquering all territories on the map. If the game is over, gameOver Boolean in the GameManager class is set to true and is returned to the reference point in Start New Game scenario given in Figure 3. If the game is not over, the gameOver Boolean is set to false and returned as described. The playerTurn() function is called until the game ends.

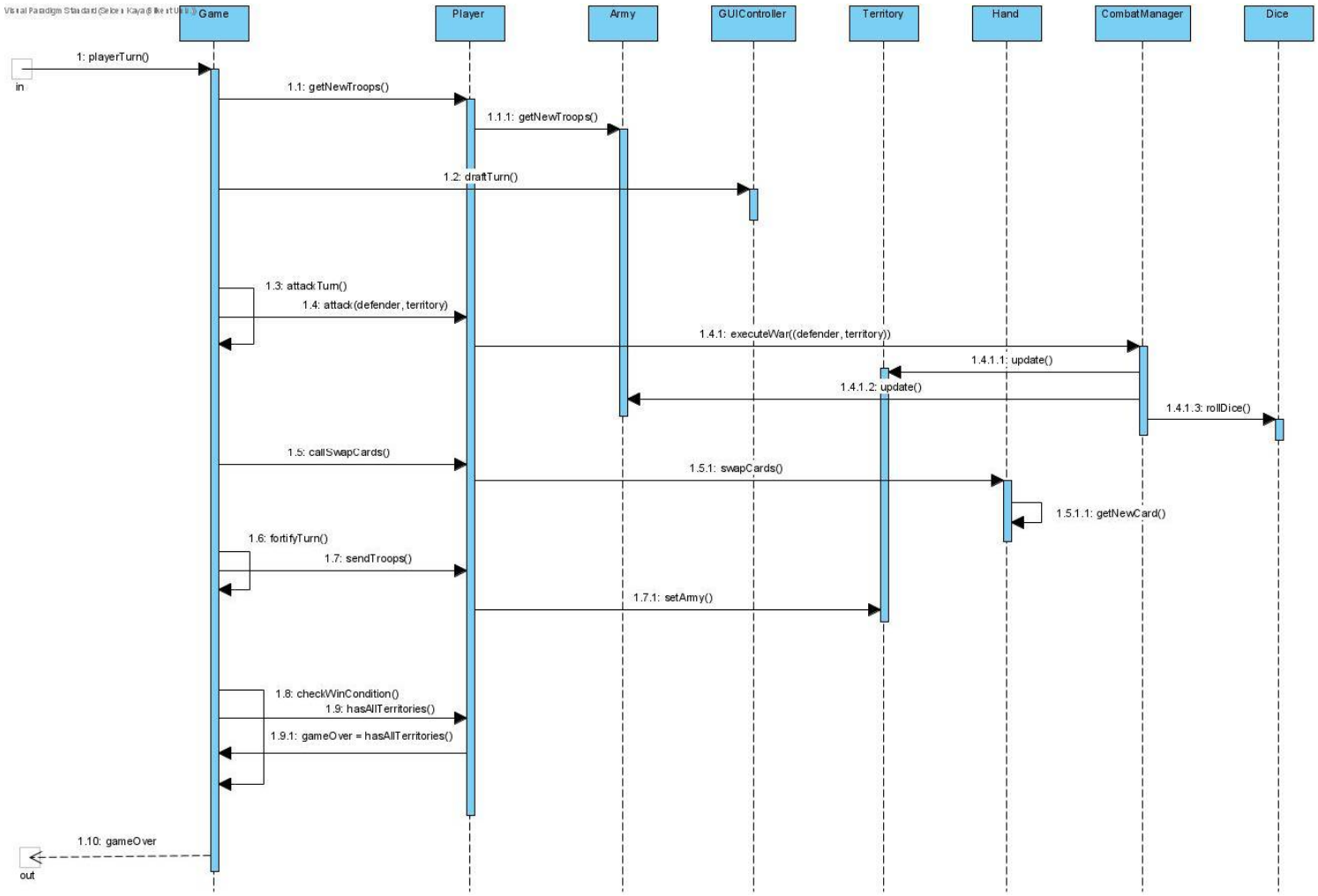

Figure 4: Sequence diagram for Play a Turn scenario.

4.3.3 How to Play

The diagram of this scenario is given in the figure below (Figure 5). The user is in the main menu or in the pause menu during a game and clicks the how to play button. A page showing the description and the mechanics of the game is shown when the “How to Play” button is clicked through the GUIController class. The page is displayed by the GUI class. The user can come back to the menu by pressing the “Back” button.

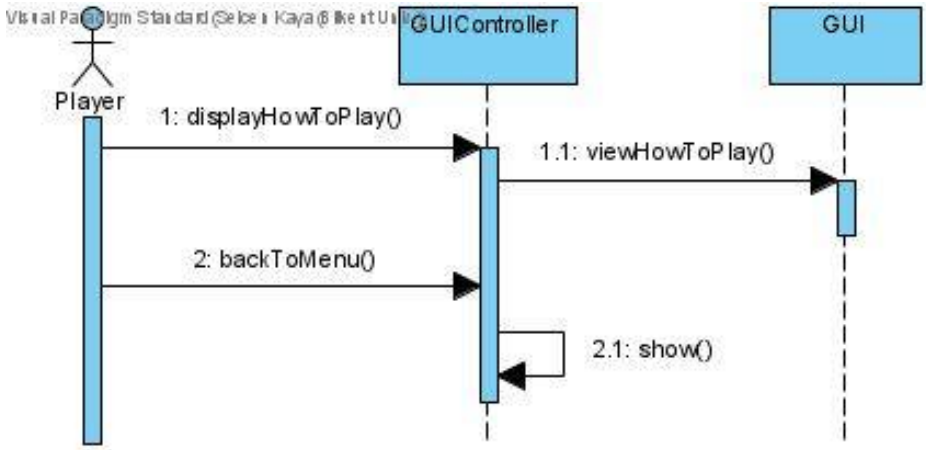

Figure 5: Sequence diagram for How to Play scenario.

4.3.4 Settings

The diagram for this scenario is given in Figure 6 below. The user is in the main menu or the pause menu during a game and clicks on the “Settings” button. The settings for the game which include sound and music are displayed by the settings() function in the GUI class through the GUIController class. The user can choose to adjust the volume of sound and music or mute them and click the “Save” button to save their settings, which are handled by the SoundManager class. The user can come back to the menu by pressing the “Back” button.

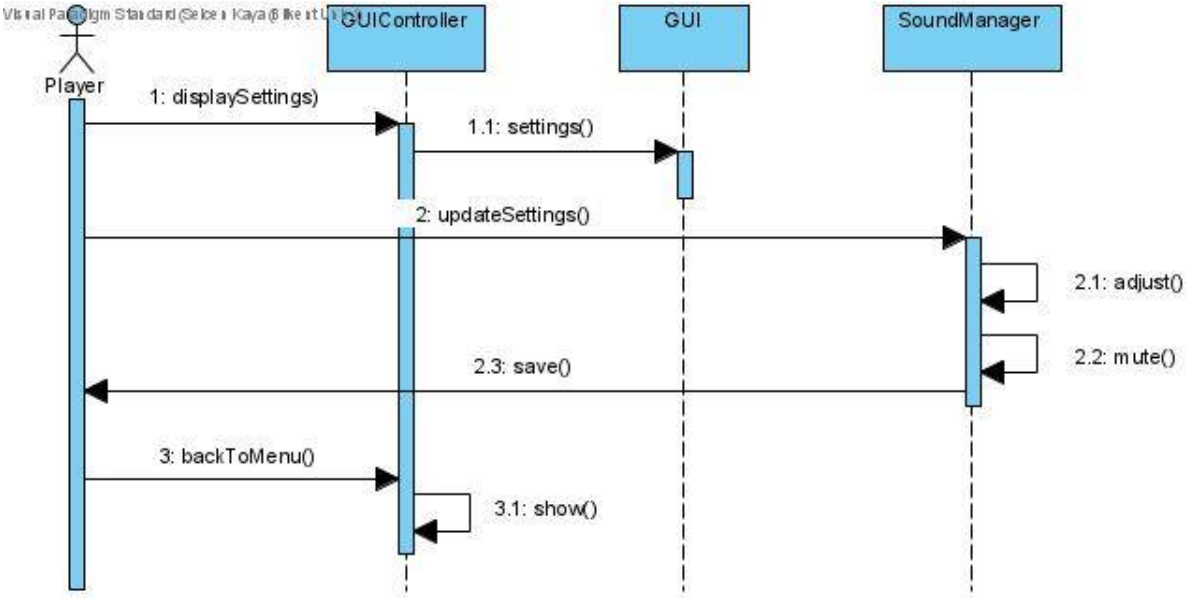

Figure 6: Sequence diagram for Settings scenario.

4.3.5 Credits

The diagram for this scenario is given in the figure below (Figure 7). The user is in the main menu and clicks the “Credits” button. A page showing all contributor’s names are displayed by the GUI class through the GUIController class. The user can come back to the main menu by clicking the “Back” button.

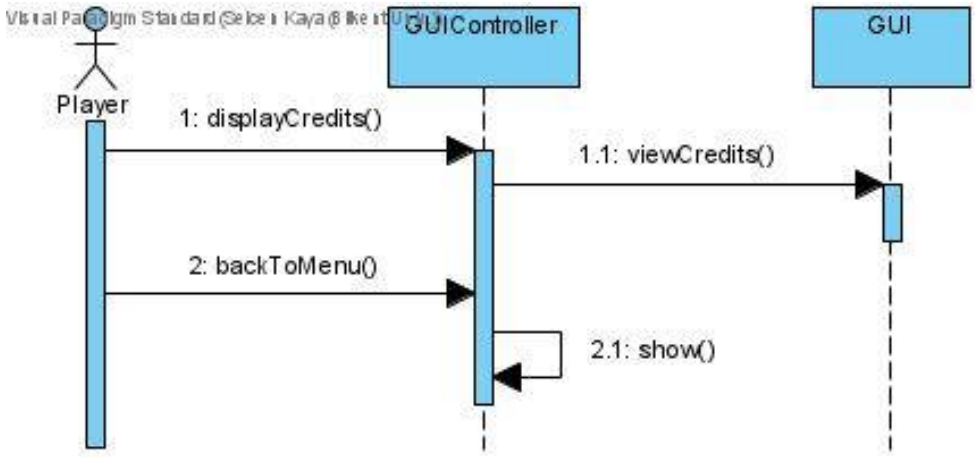

Figure 7: Sequence diagram for Credits scenario.

4.3.6 Pause

The diagram for this scenario is given in Figure 8. The user is playing a game and clicks the pause button. The pause menu is displayed which includes buttons saying “Continue”, “How to Play”, “Settings” and “Quit”. The user may choose to press any of these buttons or go back to the game with the back button. The diagram shows references to How to Play and Setting scenarios instead of showing them for clarity which are given in Figure 5 and Figure 6 respectively. The user can also quit the program which calls for exitGame() function in the GameManager class.

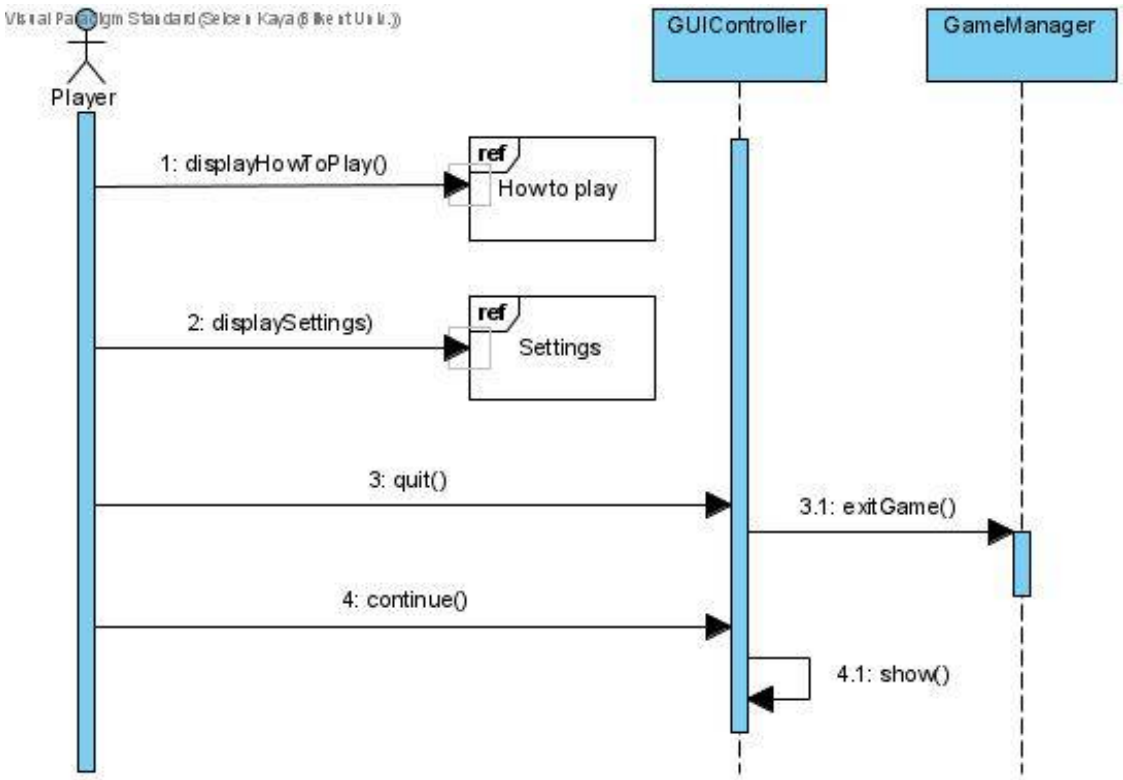

Figure 8: Sequence diagram for Pause scenario.

4.4 Activity Diagram

The activity diagram representing the actions of a player in a turn is given in the figure below (Figure 9).

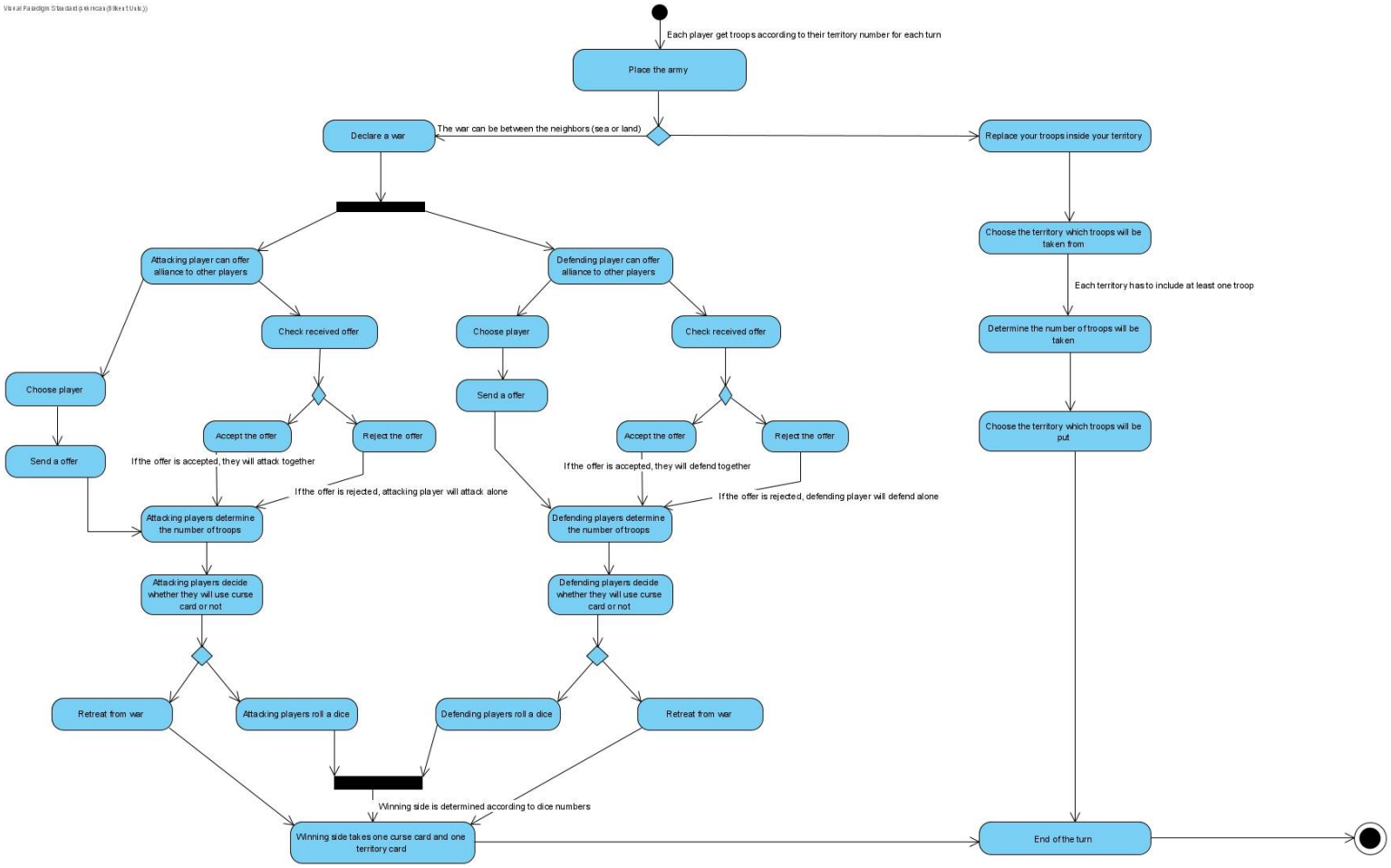

Figure 9: Activity diagram of the project.

The turn starts with placing the new army which is given to the player by the system according to the number of territories. The player can put their troops in any territory which they owns. Then, the player has two alternatives to continue his turn.

The first action is that a player can declare a war on a territory owned by another player, but to declare a war, the opponent's territory must have a border to the player's territory, and the border can be either land or sea. In a turn, a player can declare only one war. After declaring a war, both sides can request help from other players by asking for an alliance. The attacker and defender can send the request to other players by choosing a player to offer alliance to. After sending the offer the receiving player can either accept or reject the request. This alliance operation can be done by attacking side and defending side. For war preparation, all the attackers, the original attacker and the ally, will decide the number of troops which are sent to war. The number of troops can be at most 3 for each attacker, but they cannot leave any territory which they own empty. This operation is also done by the defending side, but they can assign

maximum two troops per defender. Then, each side decides whether they will use a curse card or not. The curse cards affect progress of the war since these cards provide advantages to war, but these cards are limited for every player. When every factor is released, attacking, and defending sides have a chance to retreat. The preparations have been made for the war and to determine which side will win, each war participant rolls dice, and the number of dice is related to the number of troops. After this operation, the dice results are compared, and the winning side is determined. Then, the winning side gains spoils of war and the defender or attacker will get one territory card or curse card. After these actions, the turn is finished for this player and the next player continues the game.

The second action is replacing your troops inside your territories. This action is important for developing strategies. This operation starts with choosing a territory which troops will be taken from. Then, the player should decide the number of troops to be sent. However, the player cannot leave any territory which they own empty, so at least one troop has to stay. After determining the number of troops, the destination territory should be chosen. After these actions, the turn is finished for this player and the next player continues the game.

4.5 User Interface Screen Mock-ups

The mock-ups of the user interface screens are shown below. These mock-ups demonstrate the general look of the game, which will most probably change as the game is developed however the basics will be the same. The components used in the making of these mock-ups were found on the internet which include the map [1], the background of the menus [2], the avatar icons [3] [4], the pop-up menus and card icons [5].

4.5.1 Main Menu

Main menu, presented in Figure 10, gives the player the options to start a new game, learn how to player the game, quit the game, go to settings of the game and learn the developers of the game.

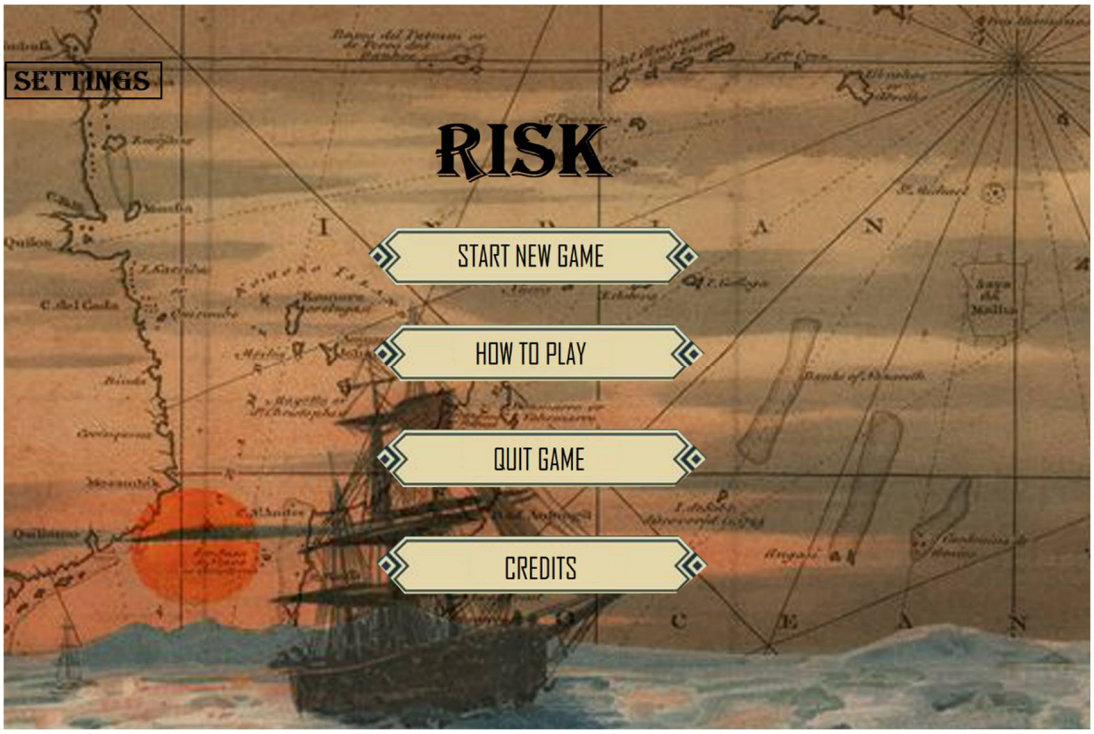

Figure 10: Main menu interface.

4.5.2 How to Play

How to play page, presented in Figure 11, gives the player information about the rules of the game. Player can go back to the main menu or the pause menu with the “BACK” button.

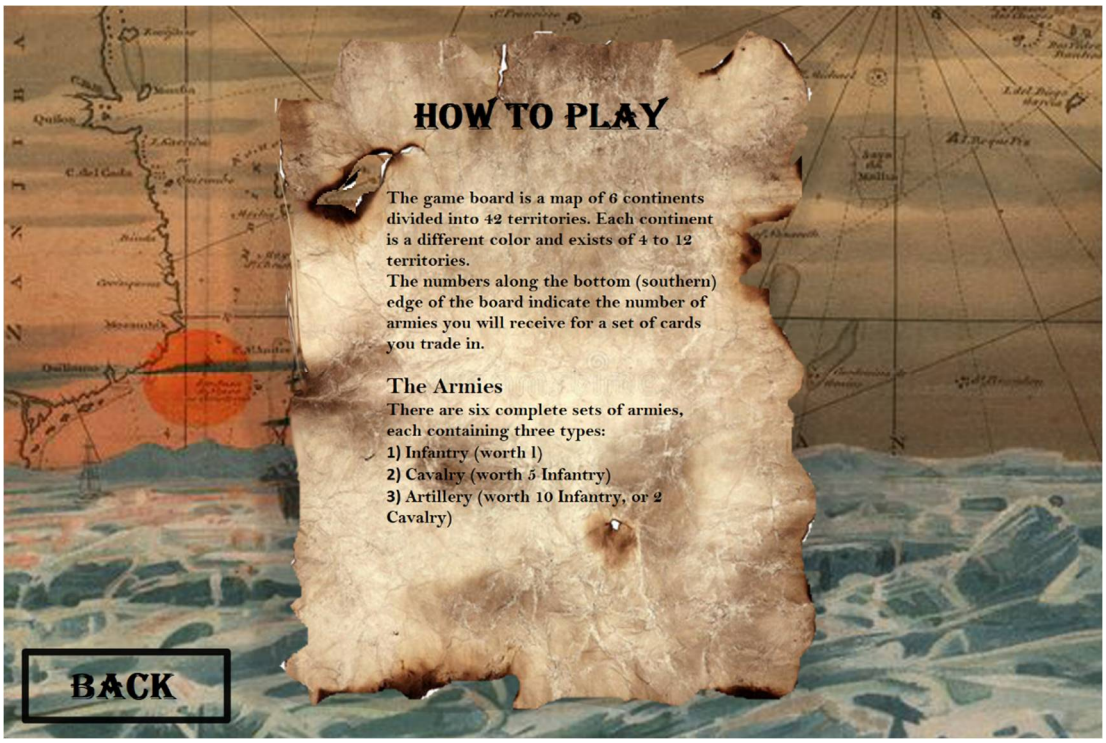

Figure 11: How to Play interface.

4.5.3 Settings

In settings page given in the figure below (Figure 12) the user will be able to change the volume or completely turn of the sounds and music of the game. Sound are the sound effects for the gameplay while music is the general background music that plays throughout the game. Player can save the changes with “SAVE” button or go back to main menu or pause menu with “BACK” button without saving the changes.

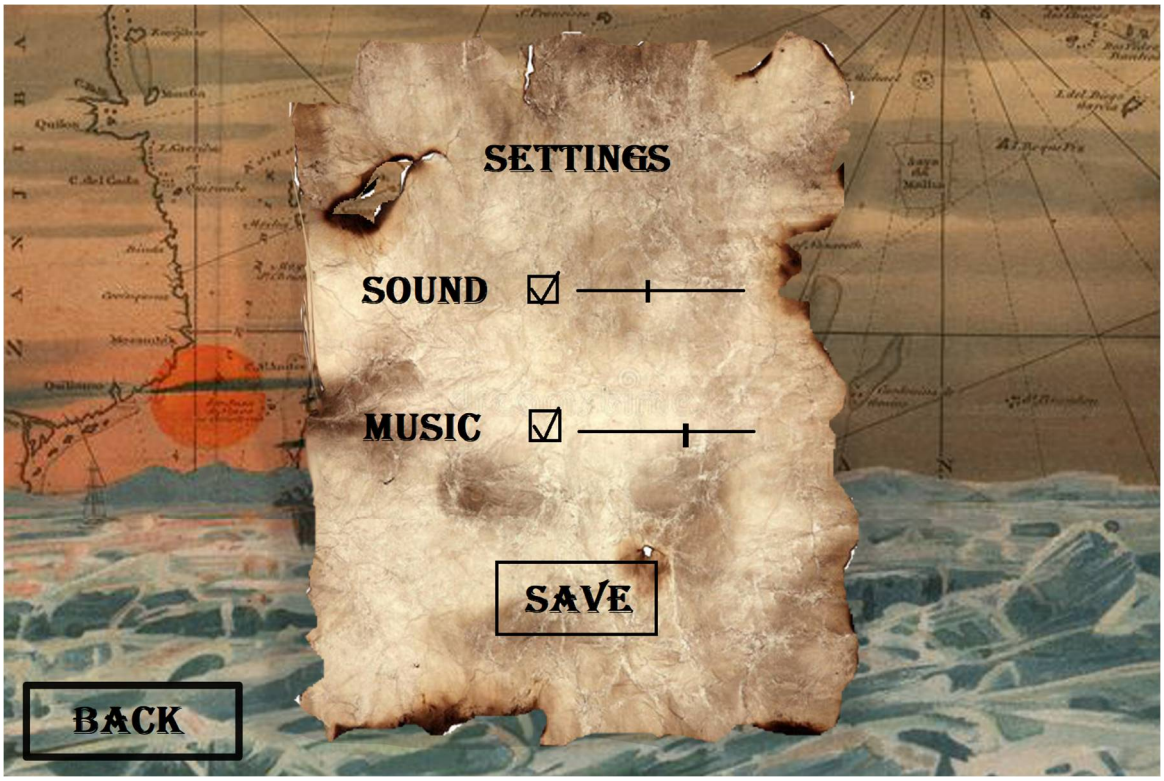

Figure 12: Settings interface.

4.5.4 Credits

Credits page, as shown in Figure 13, contains the names of the developers of the game. Player can go back to main menu with “BACK” button.

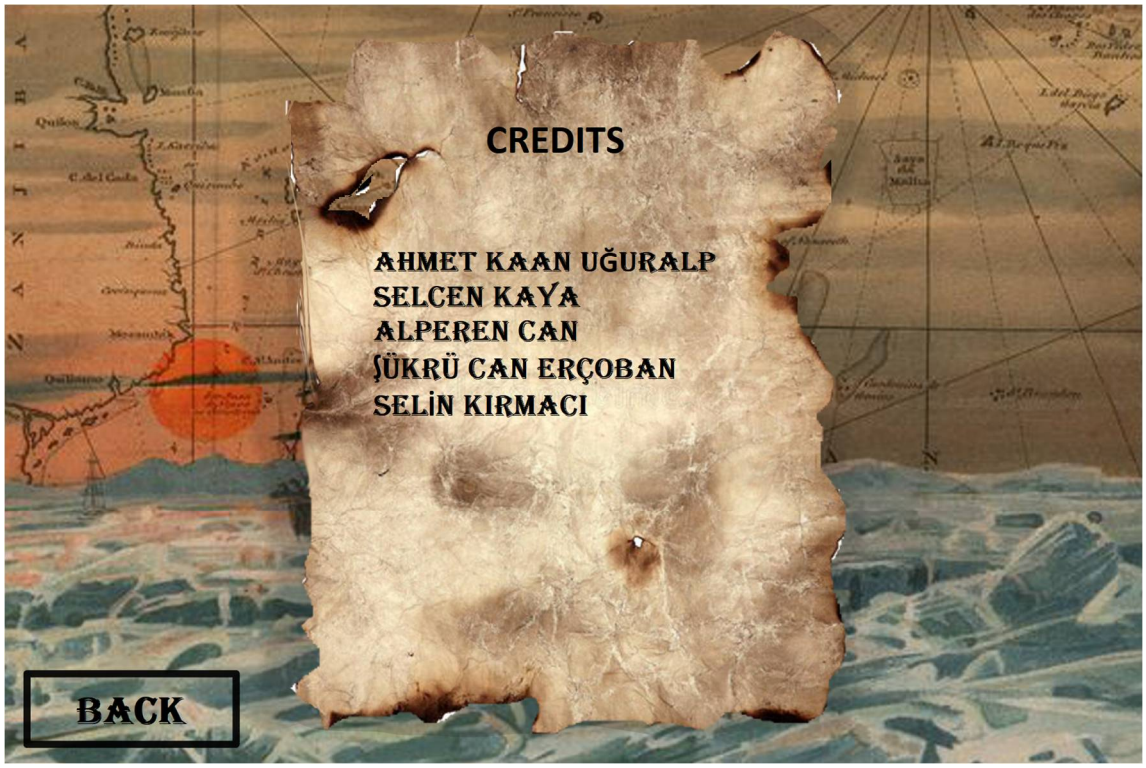

Figure 13: Credits interface.

4.5.5 Setting the Game

The player will be directed to this page given in the figure below (Figure 14) when they start a new game. They will be able to choose the number of players. Every player will be able to write their name, choose their territory color and their avatar as well. With “CONTINUE” button game will start with the chosen options. With “BACK” button changes will be lost and player will go back to the main menu.

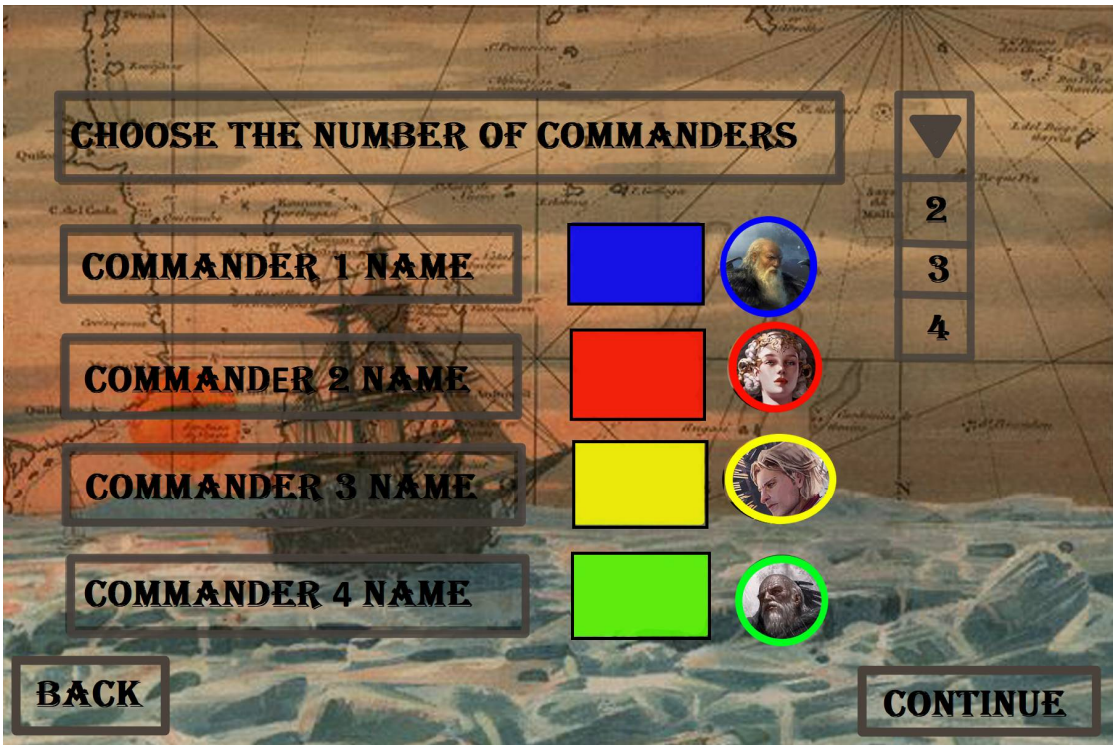

Figure 14: Setting the Game interface.

4.5.6 Gameplay

This page shown in Figure 15 is the main game view. Player's avatars will be displayed in the corners and when its player's turn, their avatar will be lit up. In the bottom, the cards of the player will be displayed and will be used when it's their turn if they want. Player will be able to attack the territories of their enemies if it is in the ruleset. Player will also be able to retreat from their territory if they find it necessary. Player is also able to pause the game with the button in the top left corner.

Figure 15: Gameplay interface.

4.5.7 Attack Pop-up

When a player decides to attack this page, given in Figure 16, will pop up. Players can roll the dice with “ROLL DICE” button and decrease the number of dice that they would like to attack with, with down arrow button. Player under attack can also call for help from other players with “ALLIANCE” button. With “BACK” button attack will be declined.

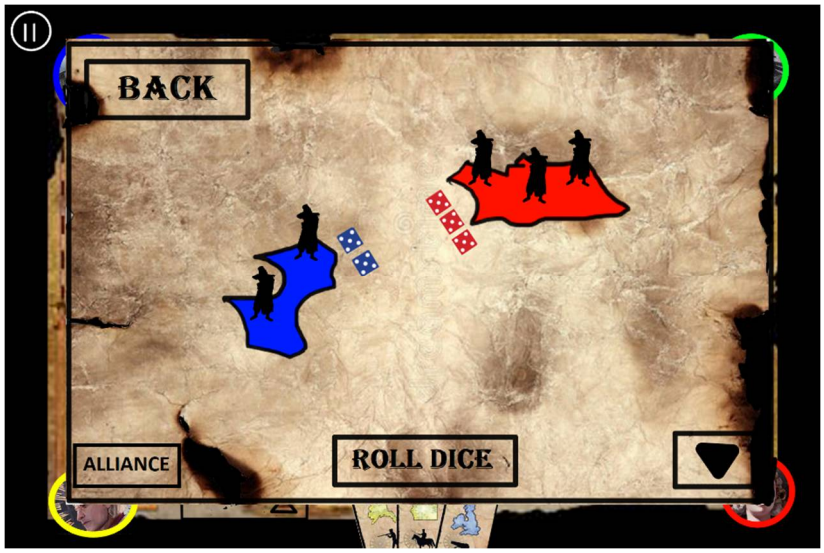

Figure 16: Attack pop-up interface.

4.5.8 Pause Game

In this menu given below (Figure 17), player is able to remind themselves of the rules of the game with “HOW TO PLAY” button that will take the player to how to play page or change the settings of the game with “SETTING” button that will take the player to the settings page. Player will be able to continue the game with “CONTINUE” button and quit the game with “QUIT” button and go back to the main menu.

Figure 17: Pause Menu interface.

4.5.9 Information Pop-up

This pop-up, shown with Figure 18, will be shown to the player when it’s their turn when new soldiers are added to their army or when they exchange their cards with soldiers. The player will be able to close this pop-up with the “X” symbol located at the top right of the pop-up.

Figure 18: Information Pop-up interface.

5. Conclusion

The analysis of our implementation of the classic board game Risk has been described with the content above. The overview of the implementation clarified the components, mechanics and the rules of the game as well as new features added by us. The functional requirements explained all the actions that the users could perform and the non-functional requirements presented the functionalities that needed to be considered in order to reach a wide and happy user base. The system models were planed out and carefully produced by all of us, with giving as much attention to detail as possible. The diagrams and descriptions will help us in the implementation of this project.

6. References and Glossary

[1] “Risk: An Unexpected Journey”. Zorcon’s World. http://zorconsword.blogspot.com/2012/12/risk-unexpected-journey-part-4.html (accessed October 24, 2020).

[2] Design You Trust. https://designyoutrust.com/ (accessed October 24, 2020).

[3] Reddit. https://www.reddit.com/r/ImaginaryWesteros/comments/elgb0m/lord_commander_mormont_by_ryan_valle/ (accessed October 24, 2020).

[4] Reddit. https://www.reddit.com/r/gameofthrones/comments/bj2tey/no_spoilers_jaime_lannister_painting_by_jewell/?utm_source=ifttt (accessed October 24, 2020).

[5] “Risk Oyun Kılavuzu”. Hasbro. https://www.hasbro.com/common/documents/dad2886d1c4311ddbd0b0800200c9a66/D3A98A4650569047F5C718754E1236CF.pdf (accessed October 24, 2020).