CS 319 TERM PROJECT

Analysis Report Iteration 2

Section 3
Group 3B
Project Name: RISK 101

Group Members

Ramazan Melih DİKSU - 21802361
Aleyna SÜTBAŞ - 21803174
Yiğit ERKAL - 21601521
Baykam SAY - 21802030
Berk TAKIT - 21803147

Contents

1. Introduction

Risk is one of the most famous strategy conquest games. This game has been a favorite game for the strategy players since 1957. Various versions of this game have been made and played by players. Risk requires a good strategy, diplomacy, and luck to capture the world. Luck is crucially important in this game due to dice. Besides a good strategy and diplomacy cannot win the game solely if dice are not on your side. Furthermore, Risk has a turn-based game structure. Therefore, thinking and strategy development is the most important part of the game. Every turn players get the new soldiers according to his or her territories and they try to complete the mission and also they try to take over the world.

In Risk101, we decided to create the modification of the Risk game. Our game will be Bilkent University themed. Players will feel this theme in the map, in the character selection part, and also in the missions. Risk101 will be an offline desktop game that can be played multiplayer on the same computer. As a group 3B, we chose to use Java to implement our adapted Risk game. While implementing the game, we will aim to create a themed Risk game that is more fun for Bilkent students to play, and besides, we will make the Risk101 following the object-oriented programming principles.

2. Overview

2.1. Gameplay

In RISK the objective of the game is to achieve complete domination of the game map by eliminating all other players. At the start of the game the players will choose the faculty they will play as and will be given a number of troops according to the player count, and then will take turns claiming territories until all of them are occupied. All players will be given a bonus objective card, which will be kept secret from the other players. After the initial deployment phase, the game will proceed as normal. A turn consists of acquiring and placing new armies, attacking if the player chooses to do so, and finally moving armies around the game board. The amount of armies a player acquires at the start of the round is determined by the value of the territories they currently control, the value of the continents they currently control, and any objective cards they have accomplished. After deploying these armies, the player can choose to attack any enemy territory that neighbors one of their own. This action can be repeated as many times as the attacker chooses, as long as there is more than one troop in the attacking territory. The attacking and defending is done via dice rolls, with both the attacker and the defender rolling dice equal to their number of troops in either the attacking or defending territory, respectively. The attacker can roll a maximum of three dice, while the defender can only roll two. The dice rolls are then compared die by die, with the highest rolls of both sides going against each other, and the outcome of each of the rolls are determined separately. In the event of a tie, the defender wins. The attacker captures the enemy territory by eliminating all defender armies. In this case, the attacker has to move a number of troops that is at minimum equal to the number of dice they rolled. A territory must never be empty in the course of the game, so the attacker must leave at least one soldier behind. The defender wins when the attacker refuses to attack again, or if the attacker goes down to a single army in the attacking territory. After the attack phase, the player can choose to move any number of troops (leaving at least one troop behind) from a friendly territory to an adjacent friendly territory. After this last phase, the turn passes onto the next player. The winner of the game is the player who manages to capture all the territories and eliminate all players.

2.1.1. Changes from the Base Game

We have made several changes to the base RISK game, changes we believe that will make the game more fun for the players. These changes are as follows.

A) RISK Cards (Removed)

We chose to remove the RISK cards from the standard RISK game because we believed the design of them to be a hindrance to the players' ability to enjoy the game. They introduce a huge element of randomness to the game without any actual active thinking. We instead replaced these with objective cards, which incentivize the players to work towards a goal instead of mindlessly drawing cards.

B) The Game Map (Modified)

To make our game have a little more character than just a RISK clone, we decided to change the traditional world map in the RISK game to a custom map we made to represent the Bilkent University Campus.

C) Objective Cards (Added)

The objective cards contain a secret objective that when completed, rewards the player with the number of troops written on it. The objective card may ask the player to capture a certain continent, hold a territory for a number of turns, eliminate a player, capture a certain amount of territories in a single turn or control a specific group of territories. When an objective is completed, the objective card is revealed to the other players and is placed next to the player who has completed it. In the player's next turn, in the deployment phase, the card is discarded and the number of troops shown on it is added to the player's deployment pool. The player then draws another objective card.

D) Faculties (Added)

In this version of RISK, there will be a pool of faculties that the players choose from at the beginning of each game. These faculties will act as their armies for the remainder of the game. Each faculty represents a faculty from Bilkent University, with a unique ability that can be used once a turn or once a game, depending on the ability's specifications. The tentative list of faculties and their respective abilities is as follows:

1) Faculty of Art, Design, and Architecture: Everything by Design
   a) Once a turn, the player may choose to swap the result of one of their die rolls with the opponent.

2) Faculty of Applied Sciences: Eastern Affinity

   a) When attacking a territory in the Eastern Campus region, the player wins in the case of a tie.

3) Faculty of Economics, Administration, and Social Sciences: The Moderate Depression

   a) Once a turn, the player may choose to reduce 1 from the dice rolls of the enemy.

4) Faculty of Education: Reeducate

   a) When the player draws a new objective card, they may choose to discard it and draw a new one instead. This can be done only once per draw.

5) Faculty of Science: Knowledge Above All

   a) Once a game, the player may choose to reveal all of the other players' objective cards.

6) Faculty of Humanities and Letters: The Pen is Mightier

   a) The player gains an extra die if the territory they are being attacked from has more armies than the territory they are defending.

7) Faculty of Law: Lawyered

   a) Once a turn, the player may choose to redo any dice roll. Both the attacker and the defender must roll their dice again.

8) Faculty of Business Administration: Hostile Takeover

   a) Once a game, in the deployment phase of their turn, the player may choose to take over an enemy occupied territory along with the enemy troops. This action also ends the deployment phase for the player.

9) Faculty of Engineering: Over-engineered

   a) The player can't roll a 1.

10) Faculty of Music and Performing Arts: Smoke and Mirrors

   a) Once a game, the player may choose to play an extra turn right after their normal turn.

2.2. Map

Our game map will be based on the map of Bilkent University. There will be three main areas: Main Campus, East Campus, and Bilkent Island. Cities in the original Risk map will be replaced with some of the main buildings from the campus of Bilkent University and the surrounding area. The detailed list of the main areas and the buildings they will include is given below.

1. **Upper Main Campus Area**
    - 1.1. Mescid
    - 1.2. Mayfest
    - 1.3. Starbucks
    - 1.4. M Building
    - 1.5. Main Campus Student Dormitories
    - 1.6. Bilka
    - 1.7. Lecture Halls (Building V)
    - 1.8. Meteksan Bookstore
    - 1.9. A and H Buildings
    - 1.10. Main Campus Sports Center
    - 1.11. SA-SB Buildings
    - 1.12. Nanotam
    - 1.13. MSSF Building
    - 1.14. Concert Hall

2. **Lower Main Campus Area**
    - 2.1. G Building
    - 2.2. Main Campus Coffee Break
    - 2.3. Square
    - 2.4. Statue of Ihsan Dogramaci
    - 2.5. Cafe in
    - 2.6. B Building
    - 2.7. Cyberpark
    - 2.8. Main Campus Mozart Coffee
    - 2.9. Main Campus Library
    - 2.10. Main Campus Cafeteria
    - 2.11. EA Building

2.3. Rules

- Every player rolls a die, the one with the highest number starts by placing an infantry to a territory and claiming it. Whoever placed the first army opens the game.
- At the beginning of each turn, the number of new armies that a player can add is calculated based on how much control they have on the map and the value of their completed objective (if they have one).
- A player can only attack a territory that is adjacent to one of their own territories which has at least two armies on it.
- The winner of a battle is decided by die rolls. If the attacker’s die is higher, the defender loses an army. If the defender’s die is higher, the attacker loses an army. In the case of a tie, the defender wins the battle.
- If a player completes an objective card, they will discard that objective and gain the amount of armies it specifies on the following turn. Then the player will draw a new objective card.
- To win a game, a player must occupy every territory in the map.

A more detailed rulebook can be accessed here:
https://www.hasbro.com/common/instruct/risk.pdf

3. Functional requirements

1. **New Game:** Players should be able to start a new game, select the number of players, and select which faculties they want to play as.
2. **Save Game:** Players should be able to save the game and continue where they left off some other time.
3. **Adjust Sound:** Players should be able to adjust the volume of the music and the sound effects separately. They also should be able to mute them completely.
4. **Place Initial Armies:** Players should be able to place an army on an empty territory to claim that territory. The game should switch to the next player when an army is placed. It should loop through all players until no army to place is left.
5. **Game Turn:** The game should iterate through the turns of all players until one player controls all the territories and wins the game:
    a. **Place Armies:** The player should gain troops at the start of their turn according to the number of areas they own. If they have accomplished their objective the previous turn, they should gain additional troops according to their objective card and draw a new card. They should be able to send these troops to the owned areas they want by clicking on the area.
    b. **Attack Enemy:** The player should be able to wage war on another area as many times as they want. To wage war, they first need to click on an area they own to determine the attackers. Then they should click on an adjacent area to determine where to attack. The attacking and defending players should be able to select the number of troops they want to fight with (corresponding to the number of dice they will be rolling).
    c. **Fortify:** The player should be able to move their troops from one owned area to another adjacent owned area.
6. **Display Territories:** The game should display territories according to the color of the ruler of that territory.
7. **Check Objectives:** The game should check if a player has accomplished their objective at the end of their turn.
8. **Simulate Dice Throw:** The game should simulate dice throws in a war.
9. **End Game:** The game should show the winner of the game when the game ends.
10. **View Help:** The players should be able to get help on both how the original risk is played and how our game is played.
11. **View Credits:** The players should be able to view the makers of the game.
12. **Exit Game:** The players should be able to exit the game at any time they want.

4. Nonfunctional requirements

Usability:

All buttons in the game will have names or images that clearly reflect what they do. Image buttons will be larger than 50px*50px. All text will be larger than 12pt. Territories will be color-coded according to the color of their rulers.

Reliability:

The application crash probability in 1 hour of gameplay will be less than 1%. There will be no memory leaks after the application is closed. Exiting the game will automatically save the game.

Performance:

The response time of the game will have the following requirements: the input from the user will get acknowledged under 10 milliseconds and the required actions will be executed and displayed under 500 milliseconds. The game will use at most 400MB of RAM.

Supportability:

The game will be open source. It will feature guidelines for building from the source code. The source code will have comments to explain specific parts.

Portability:

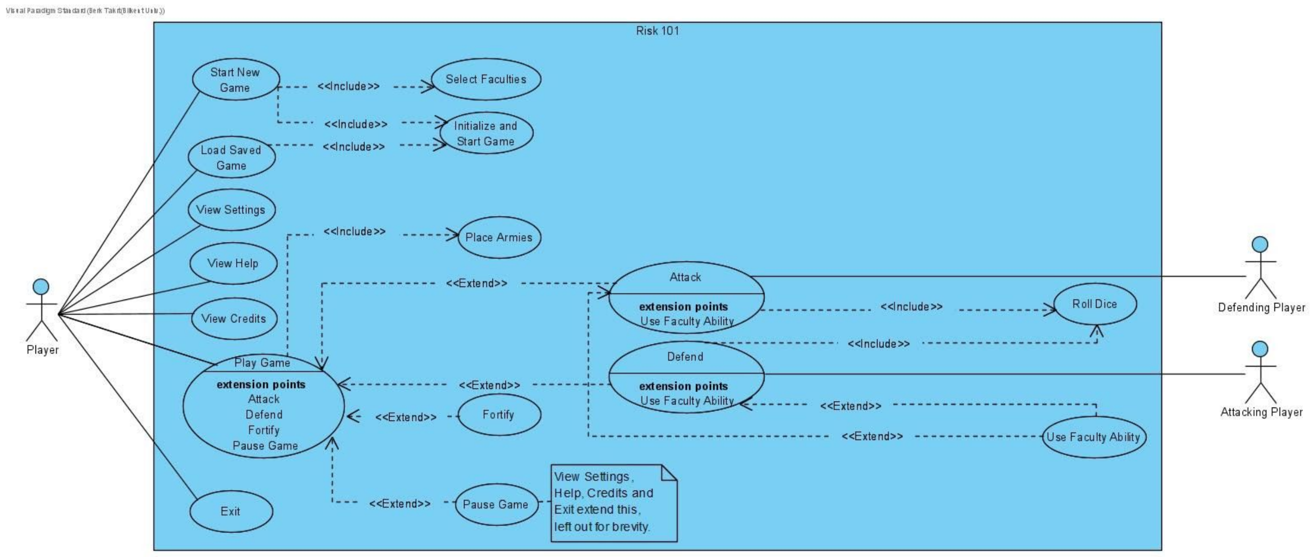
The game will run on Windows, Mac, and Linux. It will run on any x86 processor that came out in the last 10 years. It will require 256MB of available storage.

Constraints and Pseudo Requirements:

In the implementation, object-oriented programming will be used. GitHub will be used for managing the project code and sharing the documentation.

5. System models

5.1. Use case model



Higher Resolution: Risk101UseCase.jpg

Figure 1. Use Case Diagram

Use Case #1

Use Case: New Game

Primary Actor: Players (2 to 6)

Stakeholders and Interests: Players want to start a new game.

Entry Condition: Player chooses the New Game option from the main menu.

Exit Condition:

- Players exit the game from the pause menu, OR
- The game ends.

Main Flow of Events:

1) Players select New Game from the main menu.
2) System prompts the players to select a save slot.
3) The players select an empty save slot.
4) The system prompts the players to select characters.
5) The players select between 2-6 characters.

6) The players select the Start Game option.
7) System initializes the game with the selected characters.

Alternative Flow of Events:

1. Players select an occupied save slot.
    a. System asks the player to confirm if they want to overwrite the save file.

Use Case #2

Use Case: Load Game

Primary Actor: Players (2 to 6)

Stakeholders and Interests: Players want to load an existing game.

Entry Condition: Player chooses the Load Game option from the main menu.

Exit Condition:

- Players exit the game from the pause menu, OR
- The game ends.

Main Flow of Events:

1. Players select Load Game from the main menu.
2. System prompts the players to select a save slot.
3. The players select a save slot to load from.
4. System loads the saved data and continues the game from that point.

Use Case #3

Use Case: Settings

Primary Actor: Player

Stakeholders and Interests: Player wants to change the settings.

Entry Condition:

- Player chooses the Settings option from the main menu, OR
- Player chooses the Settings option from the pause menu.

Exit Condition:

- Player exits back to the main menu by choosing the back option, OR
- Player goes back to the pause menu by choosing the back option.

Main Flow of Events:

1. Player chooses the Settings option from the main menu or the pause menu .
2. System displays settings.
3. Player makes changes to the settings.
4. Player applies the settings by pressing the apply button.
5. System applies setting changes to the game.

Alternative Flow of Events:

1. Player wants to discard the setting changes.
    a. Players press the discard button .

<u>Use Case #4</u>
Use Case: Credits
Primary Actor: Player
Stakeholders and Interests: Player wants to view the credits.
Entry Condition:
- Player chooses the Credits option from the main menu.

Exit Condition:
- Player exits back to the main menu by choosing the back option.

Main Flow of Events:
1. Player chooses the Credits option from the main menu.
2. System displays credits.

<u>Use Case #5</u>
Use Case: Help
Primary Actor: Player
Stakeholders and Interests: Player wants to view the help page.
Entry Condition:
- Player chooses the Help option from the main menu, OR
- Player chooses the Help option from the pause menu.

Exit Condition:
- Player exits back to the main menu by choosing the back option, OR
- Player goes back to the pause menu by choosing the back option.

Main Flow of Events:
1. Player chooses the Help option from the main menu or the pause menu.
2. System displays the help page.

<u>Use Case #6</u>
Use Case: Exit
Primary Actor: Player
Stakeholders and Interests: Player wants to exit the game.
Entry Condition:
- Player chooses the exit option from the main menu.

Exit Condition: None.
Main Flow of Events:

1. Player chooses the Exit option from the main menu.
2. System prompts the user with an “Are you sure?” prompt.
3. Player chooses Yes.
4. System shuts down the game.

Alternate flow of events:

1. Player chooses the No option on the prompt.
    a. Player goes back to the main menu.

Use Case #7

Use Case: Play Game

Primary Actor: Player

Stakeholders and Interests: Player wants to play the game.

Entry Condition:

- The system has loaded a game, AND
- It’s the player’s turn.

Exit Condition:

- Player exits the game or the game ends.

Main Flow of Events:

1. Player places their armies on the map.
2. Player enters the Attack phase, chooses to attack another players’ territory.
3. Both players select attacking/defending army numbers and roll dice.
4. The dice roll is evaluated and the system decides whether the attacker has taken over the territory.
5. Player chooses to end their Attack phase, moving on to the fortify phase.
6. Player chooses to move troops from one of their territories to an adjacent one.
7. Player ends their turn, the next player plays their turn.

Alternate flow of events:

1. One or both of the players choose to use their faculty abilities in the Attack phase.
    a. System handles the faculty abilities in different ways for each faculty.
2. Player chooses to skip their Attack phase
    a. The player moves on to the Fortify phase without attacking any territory.
3. Player chooses to keep attacking in their Attack phase.
    a. The player repeats steps 2 to 4.
        i. The player decides to end their Attack phase, moving to step 5, OR
        ii. The player runs out of territories that can attack.
4. Player chooses to skip their Fortify phase, ending their turn.

5.2. Dynamic models

5.2.1. Sequence Diagrams

Scenario name: Execute New Game

Scenario: Player clicks the game icon to initialize the game. Then, the player clicks the "Help" button to get information about the game. When the player learns how to play Risk101, he returns to the main menu. Lastly, the player clicks the "New Game" button to start playing a new game.

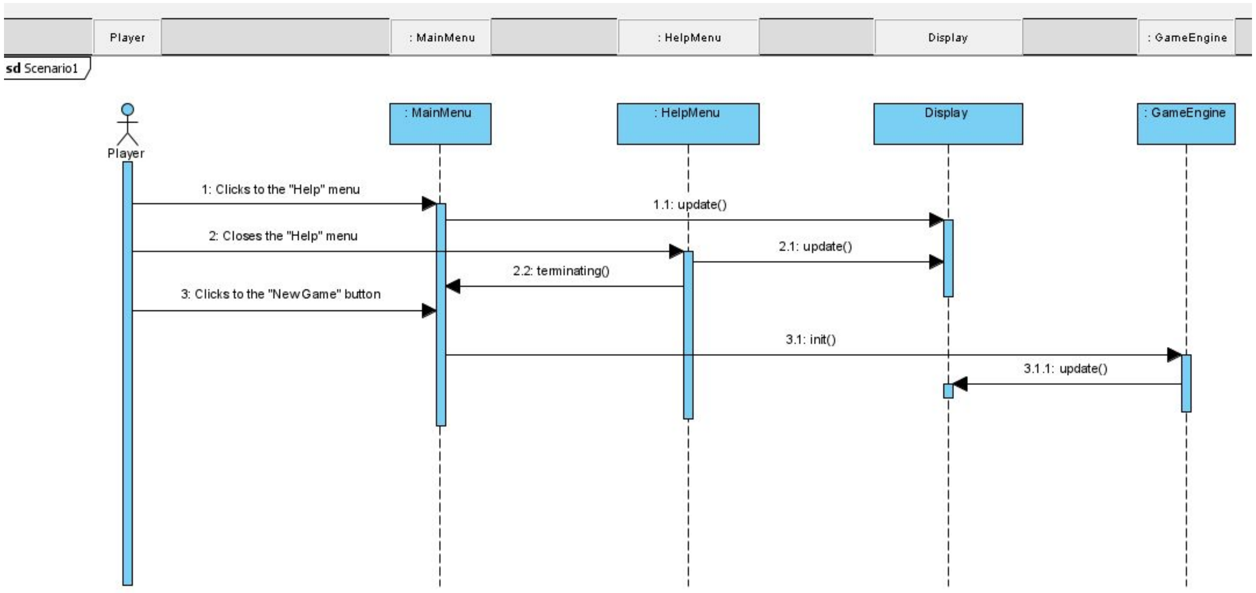


Figure 2. Sequence Diagram: Execute New Game

Description: When the player clicks the game icon, the game panel opens and it shows directly the main menu. After that game waits until the next instruction received from the player. When the player clicks the "Help" button, the game opens the new small frame that explains everything about the game. After the player closes this frame, he or she clicks the "New Game" button to start the game. This process continues until the player clicks the "Exit Game" button. After this process, the game program stops and closes the window.

Scenario name: Change Settings

Scenario: Player has already opened the game and learnt every information that he needed to know from the "Help" section. However, this player does not want to play the game with bordered screen and game music. Then, this player clicks the "Settings" button to change it. In the settings part, the player chooses the full screen option rather than the bordered screen option and closes the game music.

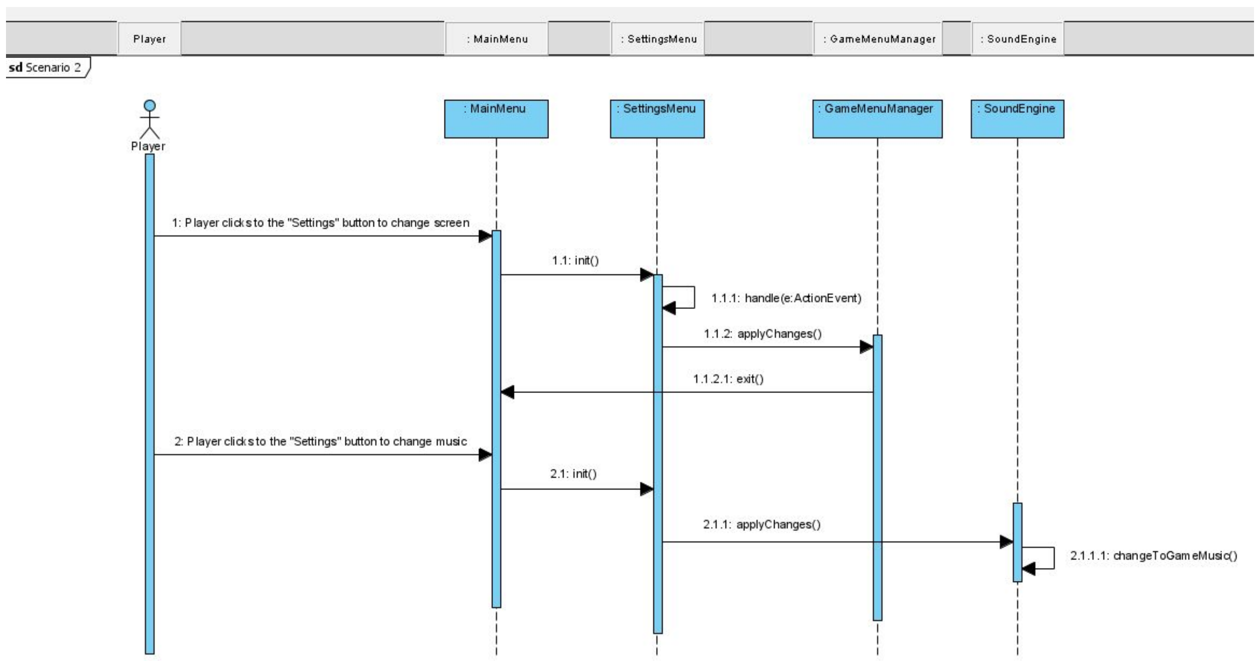

Player
: MainMenu
: SettingsMenu
: GameMenuManager
: SoundEngine
sd Scenario 2
1: Player clicks to the "Settings" button to change screen
1.1: init()
1.1.1: handle(e:ActionEvent)
1.1.2: applyChanges()
1.1.2.1: exit()
2: Player clicks to the "Settings" button to change music
2.1: init()
2.1.1: applyChanges()
2.1.1.1: changeToGameMusic()

Figure 3. Sequence Diagram: Change Settings

Description: When the player clicks the "Settings" button, MainMenu opens the SettingsMenu. In this section, the player sees the two options section. One of them is to change the screen size and the other one is to set the action sound and game music. In this scenario, the player sets and clicks the full screen and closes the game music. Then, the screen becomes a full screen and game music stops until the player wants to change it again.

Scenario name: Initial Deployment

Scenario: Player has already started the new game. Initially, the player picks the territories according to his or her tactics. Then, the player passes to deployment phase. In this phase, the player sees a screen that shows his or her territories brighter and the player starts to place the soldiers one by one.

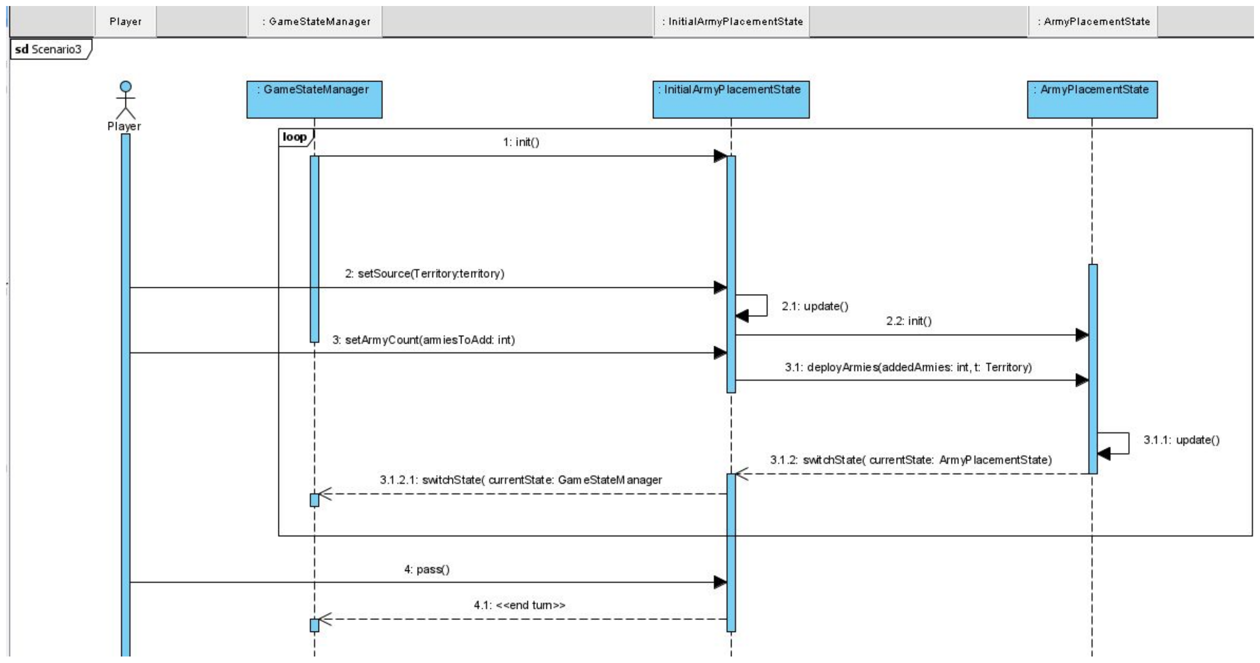
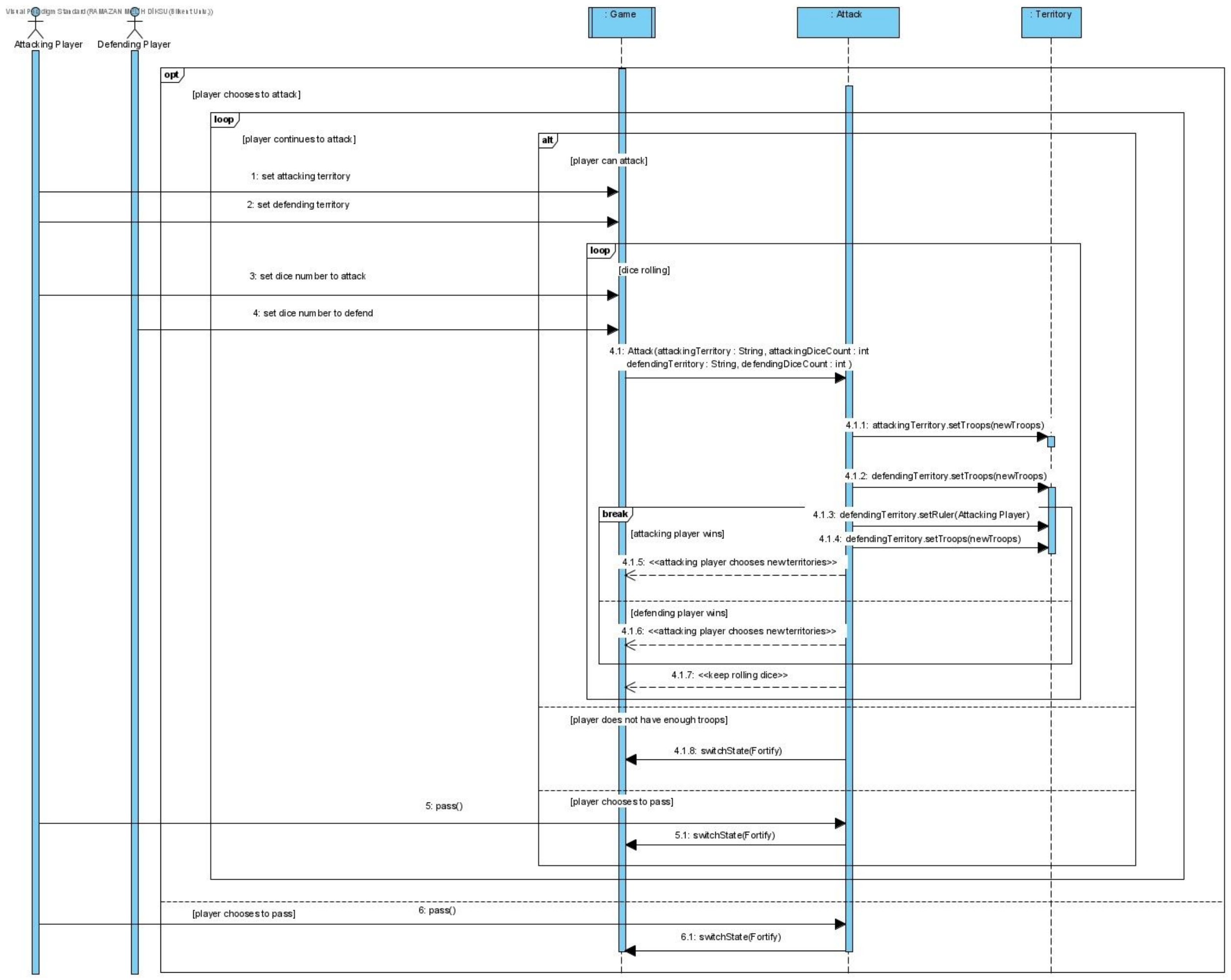

Player
: GameStateManager
: InitialArmyPlacementState
: ArmyPlacementState
sd Scenario3
Player
: GameStateManager
: InitialArmyPlacementState
: ArmyPlacementState
loop
1: init()
2: setSource(Territory:territory)
2.1: update()
2.2: init()
3: setArmyCount(armiesToAdd: int)
3.1: deployArmies(addedArmies: int, t: Territory)
3.1.1: update()
3.1.2: switchState(currentState: ArmyPlacementState)
3.1.2.1: switchState(currentState: GameStateManager
4: pass()
4.1: <<end turn>>

Figure 4. Sequence Diagram: Initial Deployment

Description: GameStateManager initializes InitialArmyPlacementState. Then, the player chooses the territory and InitialArmyPlacementState gets updated. ArmyPlacementState gets initialized and the player chooses the army count to add. Then, placeArmies() method places the soldiers to the desired territory and ArmyPlacementState gets updated. After this update, the game returns to GameStateManager and does the same sequence again if there are unplaced soldiers left. When the placeable soldiers are finished, the loop ends and the player ends the turn.

Scenario Name: Attack

Scenario: After the army placement state, the player will have a chance to attack or pass on to the fortifying state. Attacking player will choose a territory to attack from and an adjacent territory to attack to. Both attacking and defending players will choose the amount of dice they want to roll. Players will roll dice until the one side wins. Then, the attacking player may choose to attack again or pass in order to fortify.


Visual Paradigm Standard (RAMAZAN MELIH DİKSU (Bilkent Univ.))
Attacking Player
Defending Player
: Game
: Attack
: Territory
opt
[player chooses to attack]
loop
[player continues to attack]
alt
[player can attack]
1: set attacking territory
2: set defending territory
loop
[dice rolling]
3: set dice number to attack
4: set dice number to defend
4.1: Attack(attackingTerritory : String, attackingDiceCount : int defendingTerritory : String, defendingDiceCount : int)
4.1.1: attackingTerritory.setTroops(newTroops)
4.1.2: defendingTerritory.setTroops(newTroops)
break
4.1.3: defendingTerritory.setRuler(Attacking Player)
[attacking player wins]
4.1.4: defendingTerritory.setTroops(newTroops)
4.1.5: <<attacking player chooses newterritories>>
[defending player wins]
4.1.6: <<attacking player chooses newterritories>>
4.1.7: <<keep rolling dice>>
[player does not have enough troops]
4.1.8: switchState(Fortify)
[player chooses to pass]
5: pass()
5.1: switchState(Fortify)
[player chooses to pass]
6: pass()
6.1: switchState(Fortify)

Higher resolution: AttackSequenceDiagram.jpg

Figure 5. Sequence Diagram: Attack

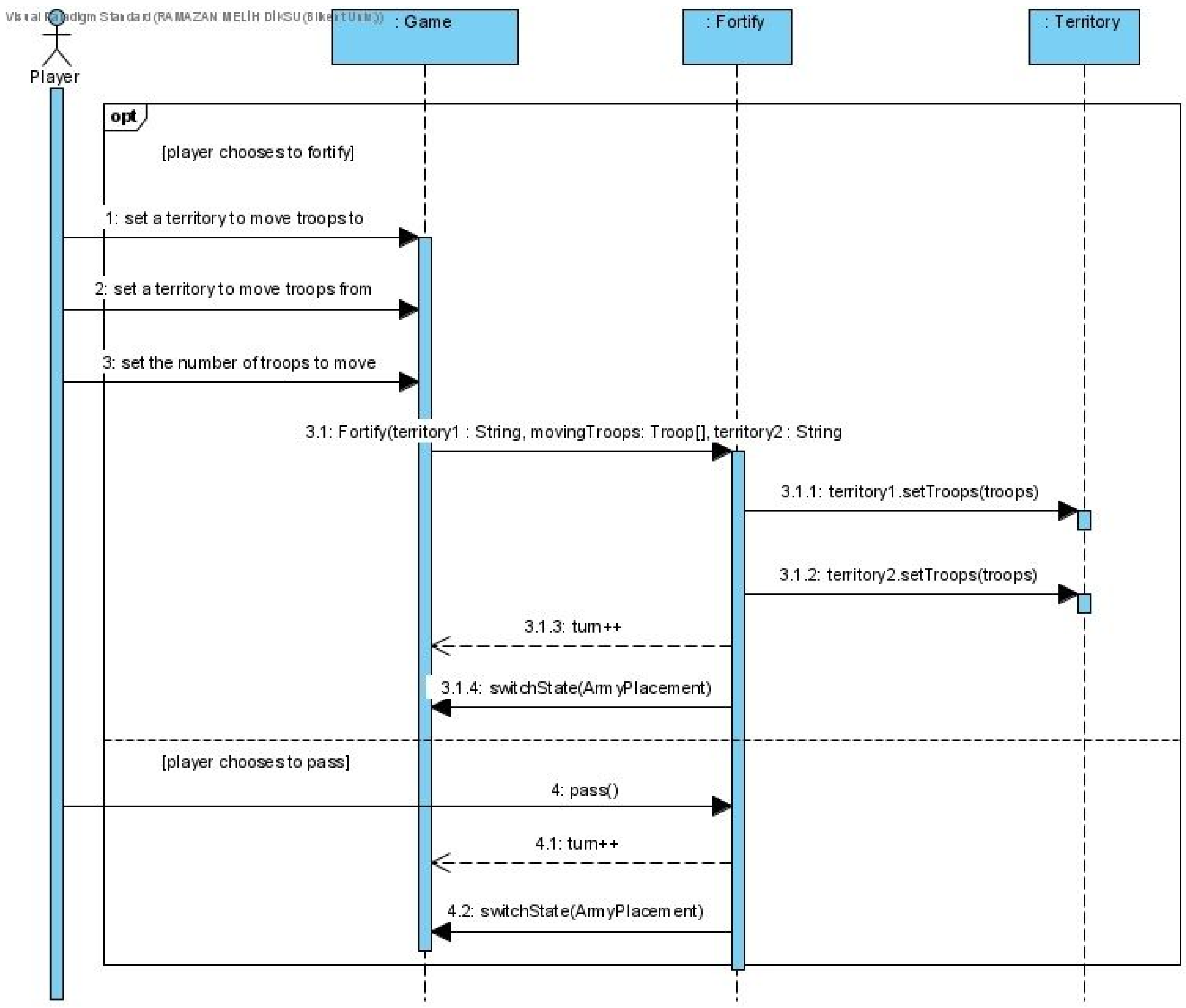
Description: At the start of the Attack Sequence, Game class will receive:

- Two territory names from the attacking player where one will be the attacking and the other one will be the defending territory.
- Amount of dice from both attacking and defending players.

The attacking player may optionally choose to pass the state where the input will be received by the Attack class and will call the switchState method on the Game to move to the Fortify state. After the inputs are received the Attack class is called and a loop for dice rolling will be initiated. In each loop state, the arrays of troops of the attacking and defending territories will be updated using set methods, according to the result of the war. If one of the sides wins, the dice rolling loop will break. If the winning player is the attacker, ruler attribute and the troop array will be updated on the defending territory. The attacking player may wish to choose two other new territories to perform the attack sequence again as shown in the player continues to attack loop. This outer loop will terminate if the attacking player has no more territories left with enough troops on or the player has chosen to pass. In that case switchState method will be called in the same way as mentioned.

Scenario Name: Fortify

Scenario: After the attack state, the player can choose to fortify or to end their turn immediately. If fortified, the player will choose a territory to move troops from, a territory to move troops to and the number of troops to move. After a single movement operation the player's turn will end.


Visual Paradigm Standard (RAMAZAN MELİH DİKSU (Bilkent Univ))
Player
: Game
: Fortify
: Territory
opt
[player chooses to fortify]
1: set a territory to move troops to
2: set a territory to move troops from
3: set the number of troops to move
3.1: Fortify(territory1 : String, movingTroops: Troop[], territory2 : String
3.1.1: territory1.setTroops(troops)
3.1.2: territory2.setTroops(troops)
3.1.3: turn++
3.1.4: switchState(ArmyPlacement)
[player chooses to pass]
4: pass()
4.1: turn++
4.2: switchState(ArmyPlacement)

Higher Resolution: FortifySequenceDiagram.jpg

Figure 6. Sequence Diagram: Fortify

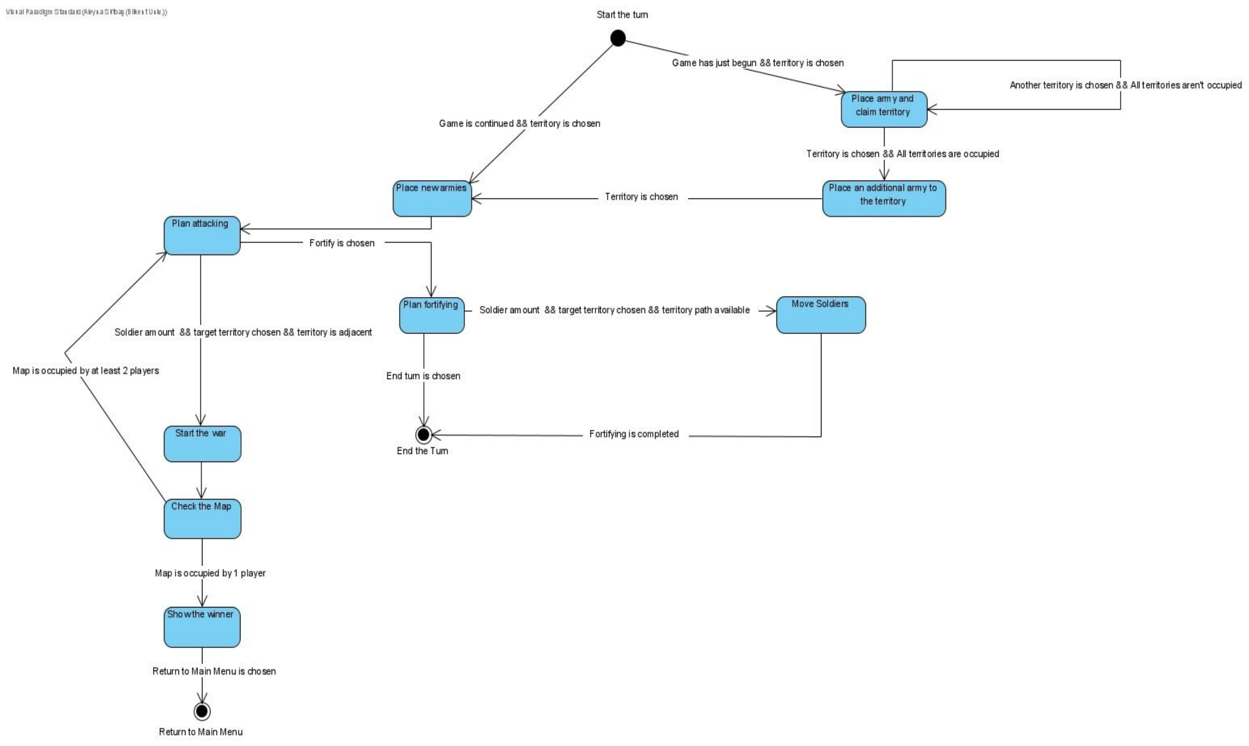
Description: At the start of the Attack Sequence, Game class will receive:

- Two territory names from the player to indicate where the transportation will be executed between.
- Amount of troops to transport from the player.

The player may optionally choose to pass the state where the input will be received by the Fortify class. Fortify class will increment the turn integer by one and will call the switchState method on the Game. After the inputs are received the Fortify class is called and the troop arrays on both territories are updated by set methods according to the moved troop amount. Turn number is incremented by one to switch the turn to the next player and the state is switched to the ArmyPlacement state with switchState method.

5.2.2. State Diagrams

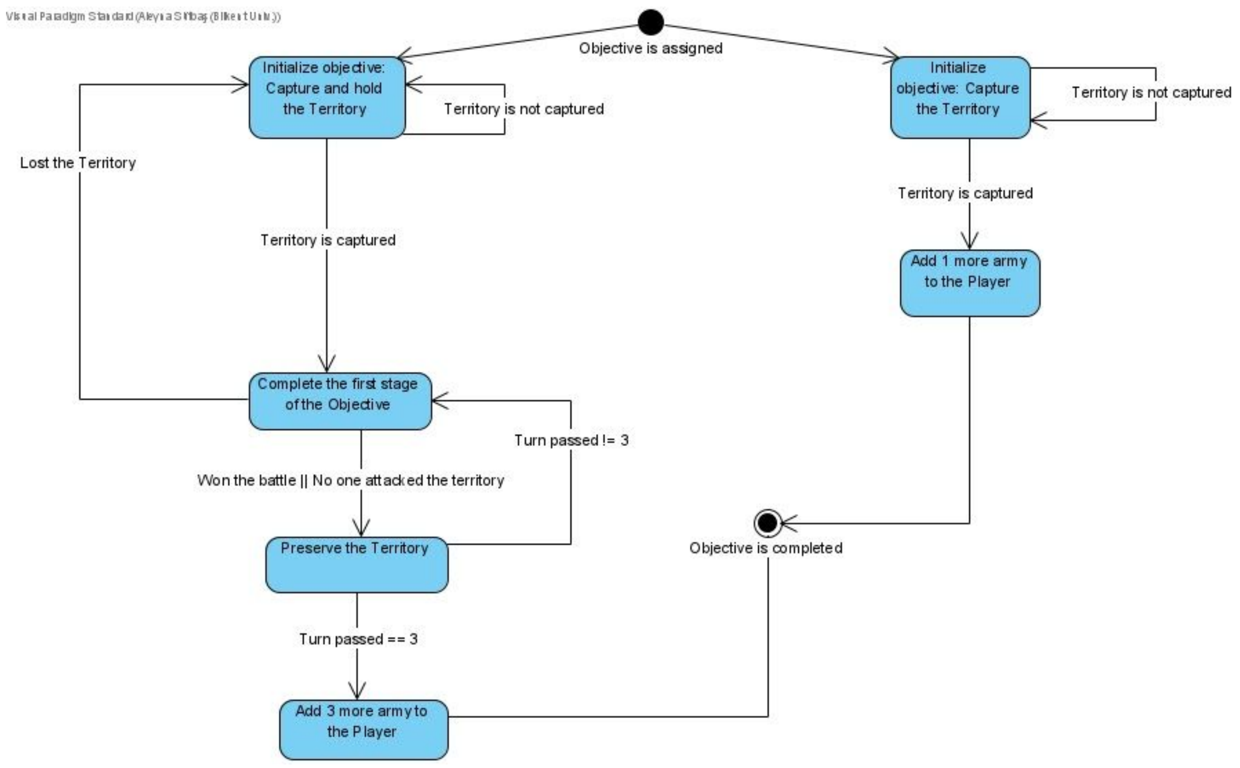
State Diagram of the Attack Stage


Visual Paradigm Standard (Aleyna Sirtbag (Bilkent Univ.))
Start the turn
Game has just begun && territory is chosen
Place army and claim territory
Another territory is chosen && All territories aren't occupied
Game is continued && territory is chosen
Territory is chosen && All territories are occupied
Place new armies
Territory is chosen
Place an additional army to the territory
Plan attacking
Fortify is chosen
Plan fortifying
Soldier amount && target territory chosen && territory path available
Move Soldiers
Soldier amount && target territory chosen && territory is adjacent
Map is occupied by at least 2 players
End turn is chosen
Fortifying is completed
End the Turn
Start the war
Check the Map
Map is occupied by 1 player
Show the winner
Return to Main Menu is chosen
Return to Main Menu

Higher Resolution: AttackStageStateDiagram.jpg

Figure 7. State Diagram: Attack

State Diagram of Objectives



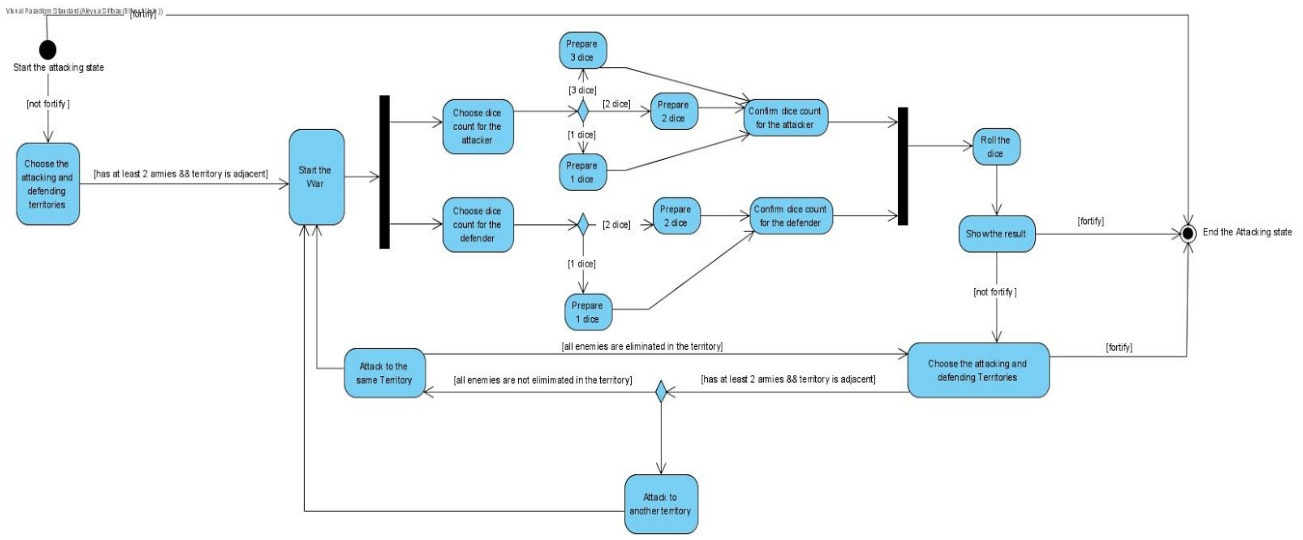
Visual Paradigm Standard (Aleyna Sirbaş (Bilkent Univ.))
Objective is assigned
Initialize objective: Capture and hold the Territory
Territory is not captured
Lost the Territory
Territory is captured
Complete the first stage of the Objective
Turn passed != 3
Won the battle || No one attacked the territory
Preserve the Territory
Turn passed == 3
Add 3 more army to the Player
Initialize objective: Capture the Territory
Territory is not captured
Territory is captured
Add 1 more army to the Player
Objective is completed

Higher Resolution: [ObjectivesStateDiagram.jpg]()

Figure 8. State Diagram: Objective

5.2.3. Activity Diagrams

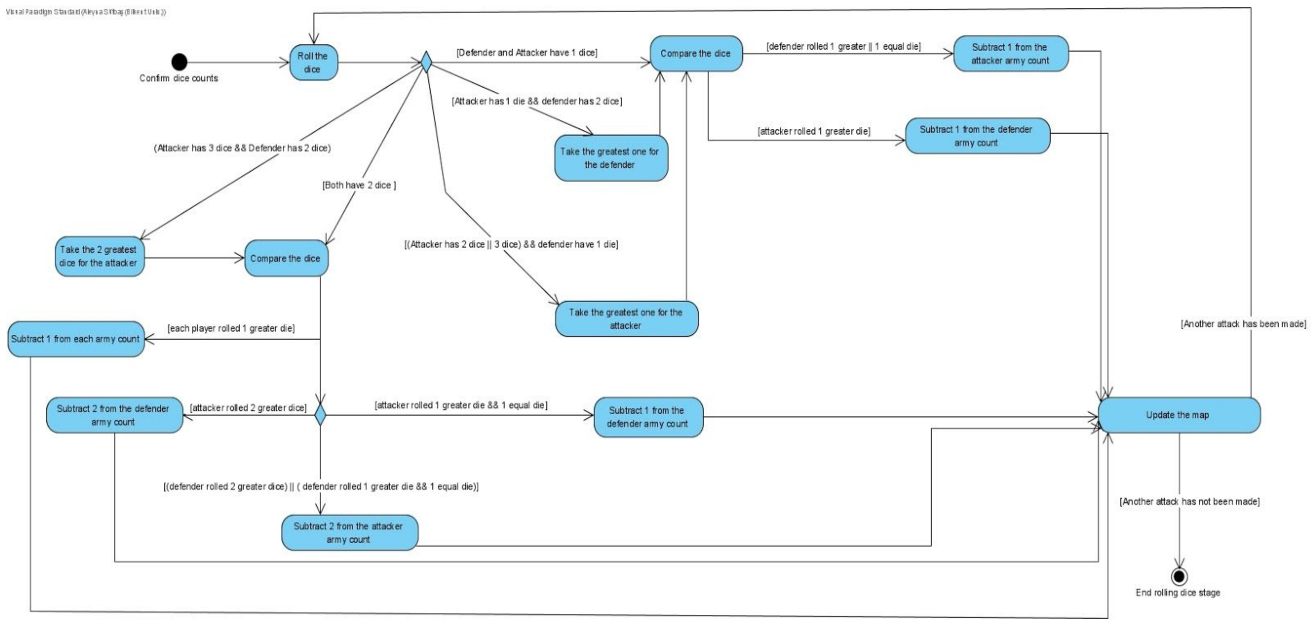
Activity Diagram for the War Stage


[fortify]
Start the attacking state
[not fortify]
Choose the attacking and defending territories
[has at least 2 armies && territory is adjacent]
Start the War
Choose dice count for the attacker
Choose dice count for the defender
Prepare 3 dice
[3 dice]
[2 dice]
Prepare 2 dice
[1 dice]
Prepare 1 dice
Confirm dice count for the attacker
Confirm dice count for the defender
Roll the dice
Show the result
[fortify]
End the Attacking state
[not fortify]
[all enemies are eliminated in the territory]
[fortify]
Attack to the same Territory
[all enemies are not elimimated in the territory]
[has at least 2 armies && territory is adjacent]
Choose the attacking and defending Territories
Attack to another territory

Higher Resolution: WarStageActivityDiagram.jpg

Figure 9. State Diagram: War Stage

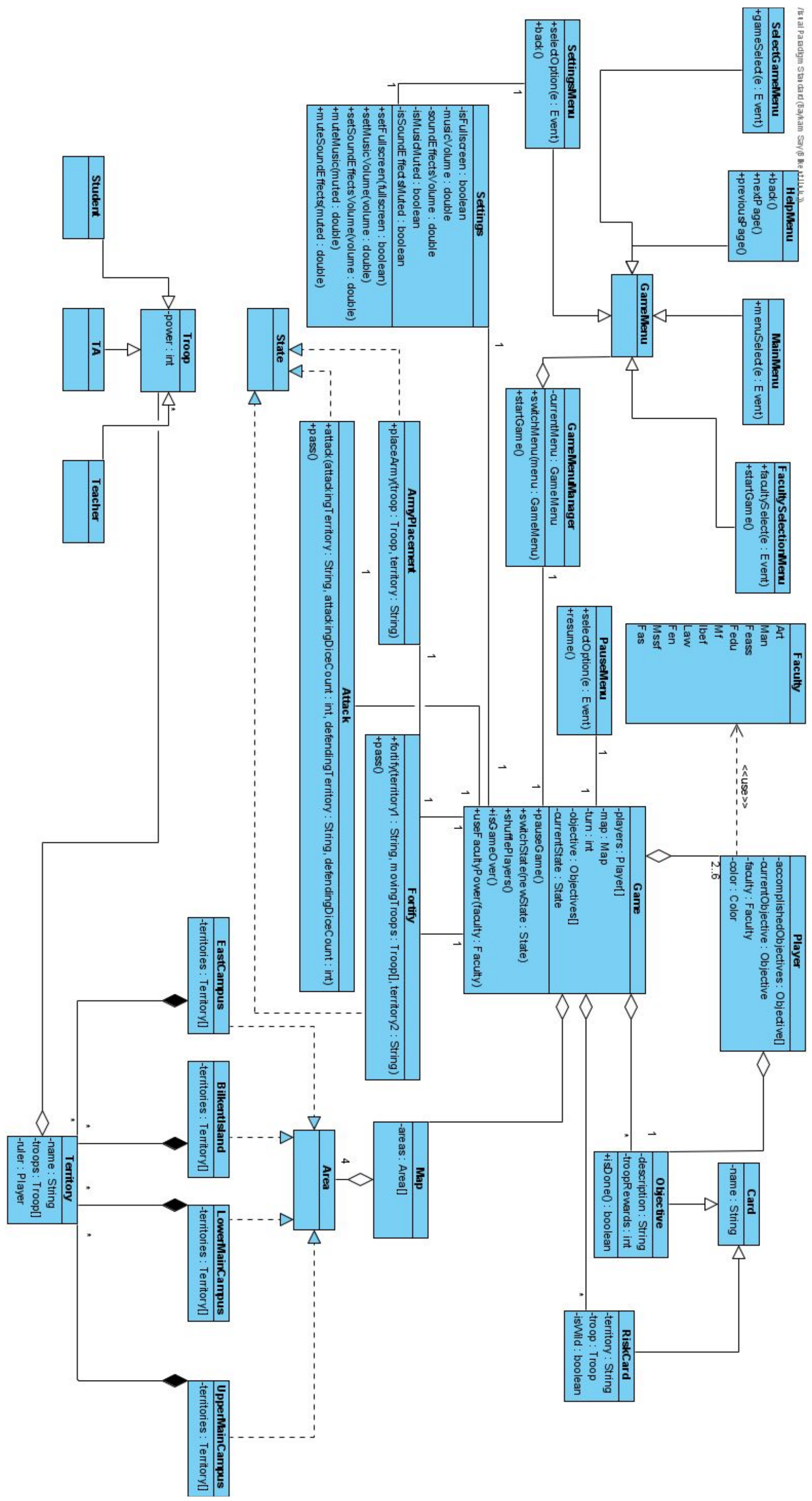
Activity Diagram for Determining the Winner of a War


Confirm dice counts
Roll the dice
[Defender and Attacker have 1 dice]
Compare the dice
[defender rolled 1 greater || 1 equal die]
Subtract 1 from the attacker army count
[Attacker has 1 die && defender has 2 dice]
Take the greatest one for the defender
[attacker rolled 1 greater die]
Subtract 1 from the defender army count
(Attacker has 3 dice && Defender has 2 dice)
[Both have 2 dice]
Take the 2 greatest dice for the attacker
Compare the dice
[(Attacker has 2 dice || 3 dice) && defender have 1 die]
Take the greatest one for the attacker
[Another attack has been made]
Subtract 1 from each army count
[each player rolled 1 greater die]
Subtract 2 from the defender army count
[attacker rolled 2 greater dice]
[attacker rolled 1 greater die && 1 equal die]
Subtract 1 from the defender army count
Update the map
[(defender rolled 2 greater dice) || (defender rolled 1 greater die && 1 equal die)]
Subtract 2 from the attacker army count
[Another attack has not been made]
End rolling dice stage

Higher Resolution: DiceWinnerActivityDiagram.jpg

Figure 10. State Diagram: War Winner

5.3. Object and class model



Higher Resolution: Object and Class Model.png

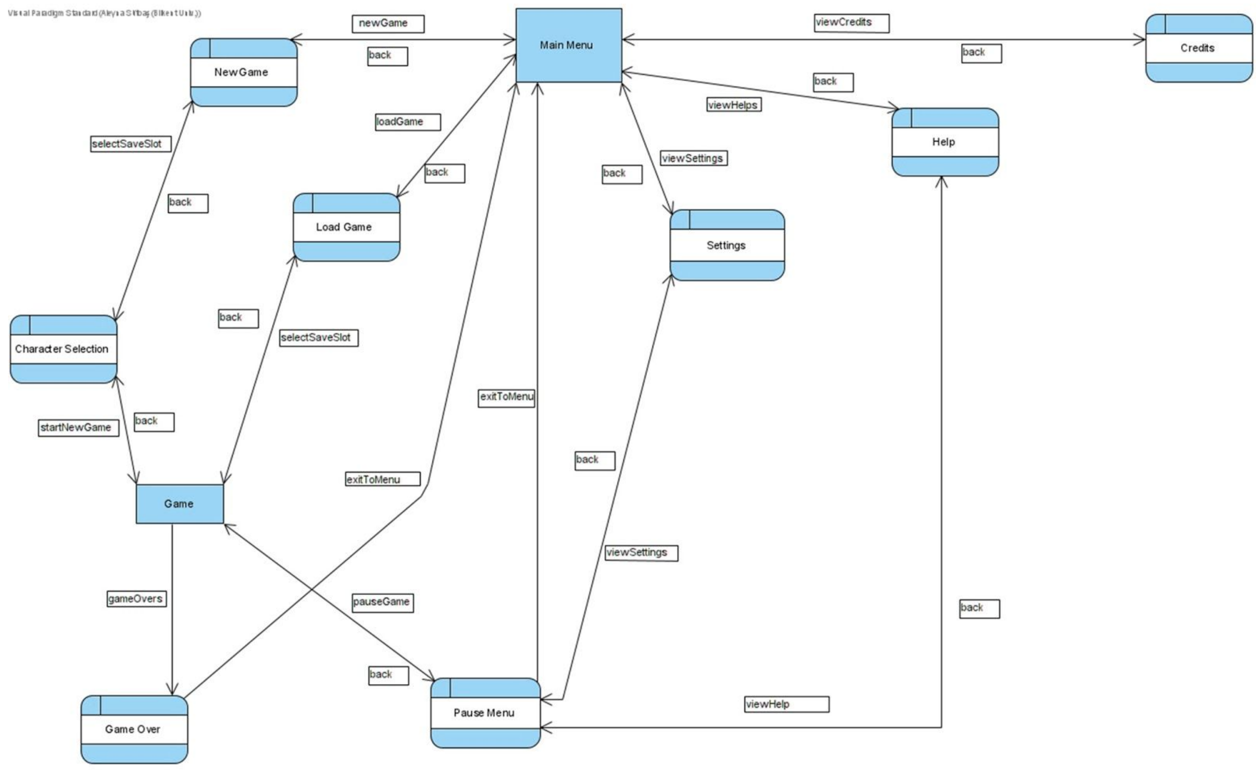
Figure 11. Object and Class Model

5.3.1. Explanation

There are two main classes in the game: the **GameMenuManager** class and the **Game** class. The GameMenuManager switches between **GameMenus** and starts the game. The Game class manages the game's entities and the different states of the game. It keeps two to six **Player** objects each having a **Faculty**, a color, and an **Objective** card. Objective cards are given to players from the Game class, each player can hold one Objective card and the rest are kept in the Game class. Objective cards give the player something to do, like capturing a territory or holding a territory for “x” number of turns and gives them a number of troops in return. Game class also keeps the **Map** of the game. Map has four areas: **EastCampus**, **BilkentIsland**, **LowerMainCampus**, and **UpperMainCampus**. Each area has a list of **Territory** objects. Each territory keeps its name, the **Troop**s on it, and its ruler. There are three different kinds of troops: **Student**, **TA**, and **Teacher**. These have different powers (student represents one dice, TA two, and teacher three). The Game also keeps the **Settings** class which stores the music and sound effects volume along with whether the game window is fullscreen. Finally, the game also has three states. These states are represented as three **State** classes: **ArmyPlacement**, **Attack**, and **Fortify**.

5.4. User interface

5.4.1. Navigational Path



Higher resolution:
https://drive.google.com/file/d/1PmuGWHjeXxh394GPWw-fbgXZNX3QzDtr/view?usp=sharing
Figure 12. User Interface: Navigational Path

5.4.2. Screen Mock-ups

a.Main Menu Screen

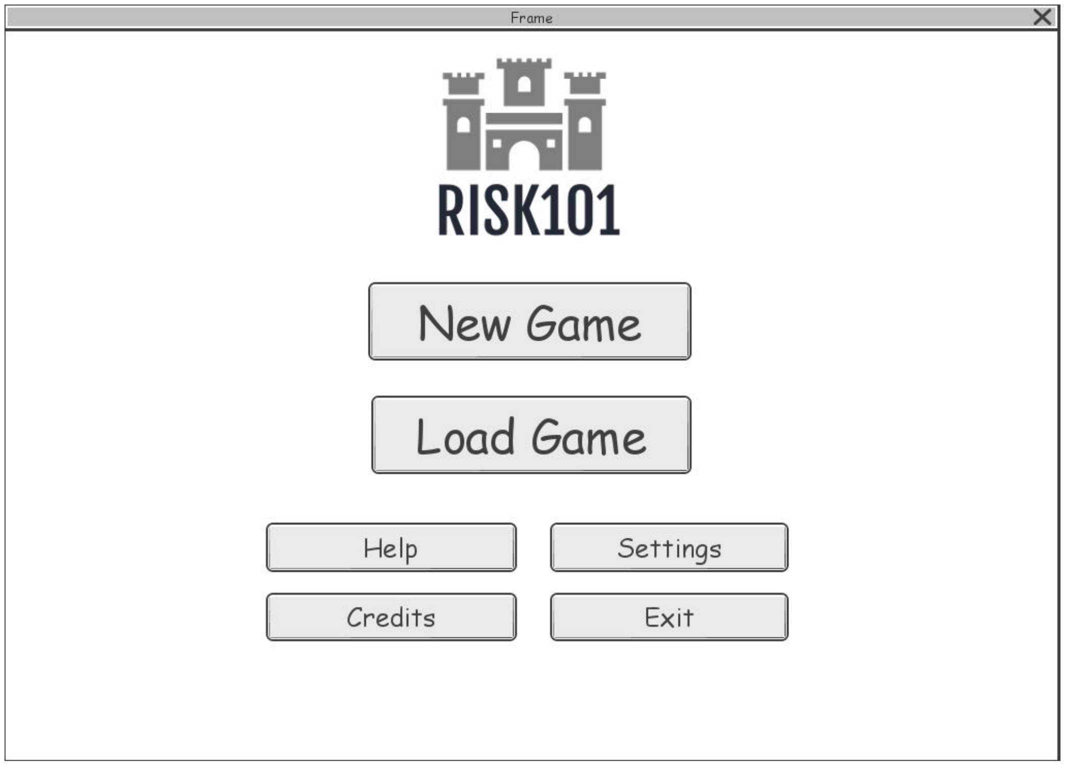


Figure 13. Main Menu GUI Mock-up

b.New Game Screen

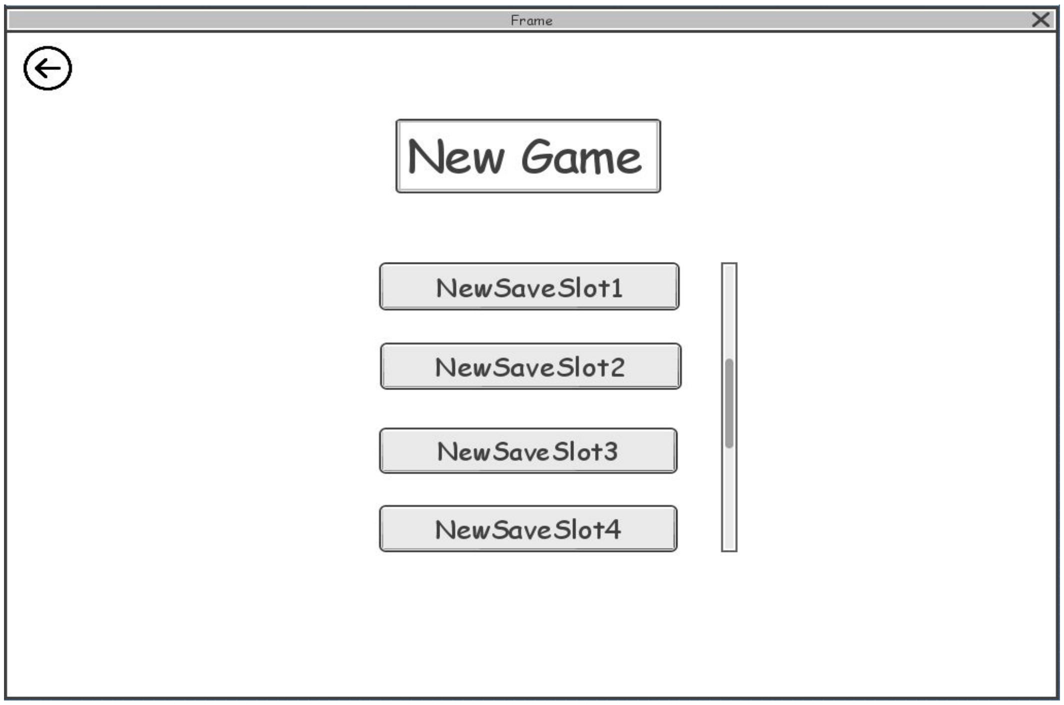


Figure 14. New Game Menu GUI Mock-up

c.Load Game Screen

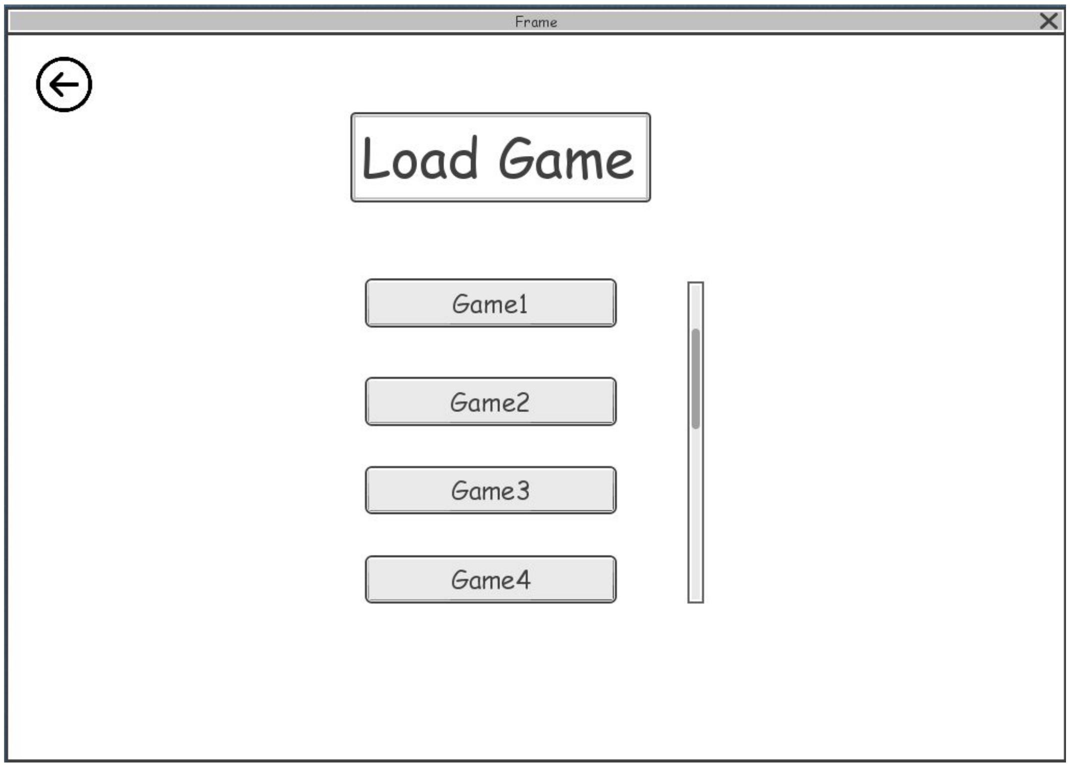


Figure 15. Load Game Menu GUI Mock-up

d.Settings Screen

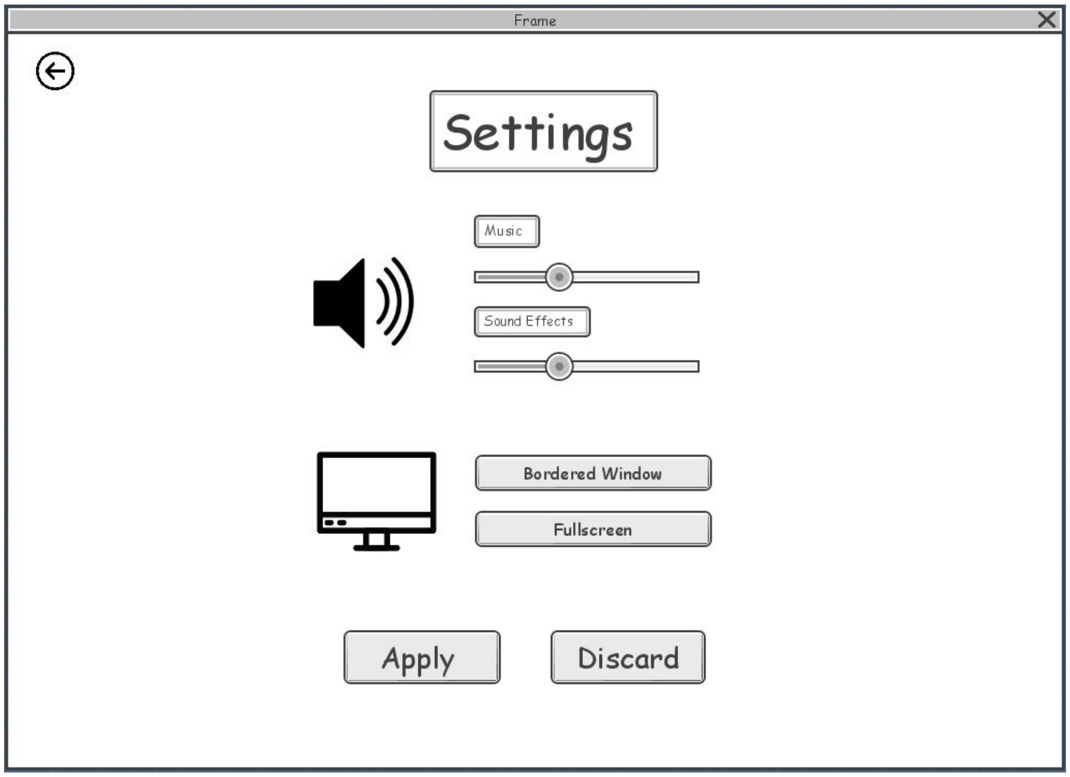
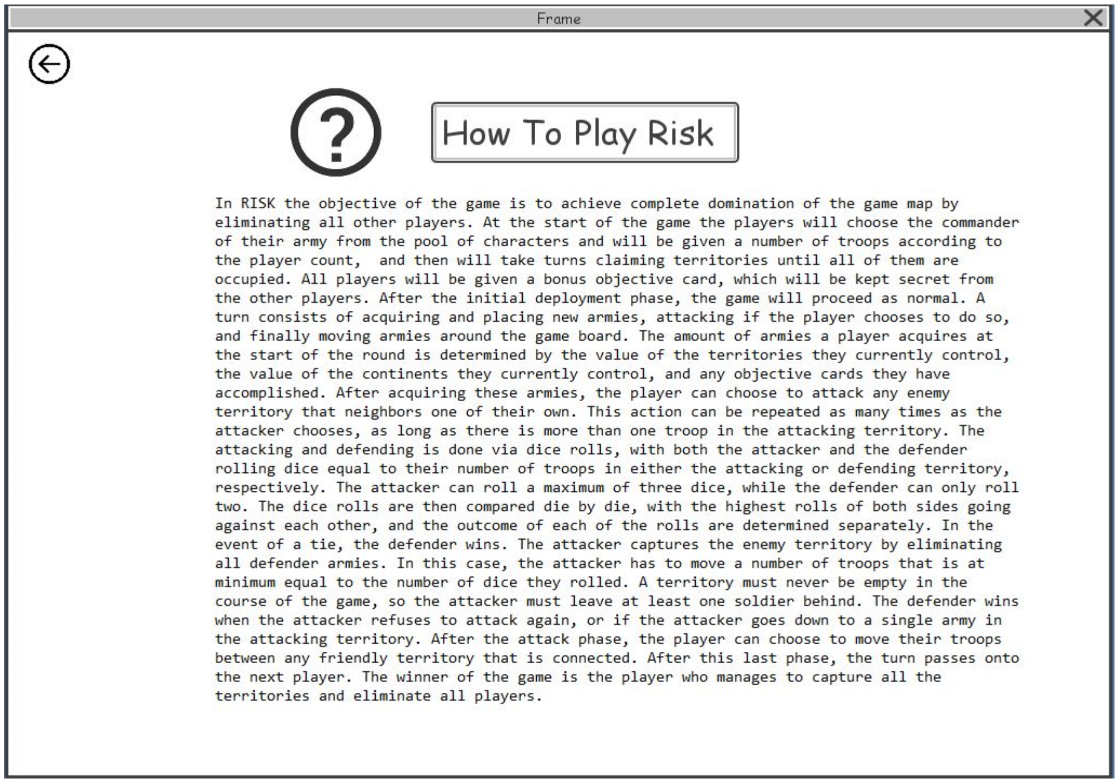


Figure 16. Settings Menu GUI Mock-up

e.Help Screen



In RISK the objective of the game is to achieve complete domination of the game map by eliminating all other players. At the start of the game the players will choose the commander of their army from the pool of characters and will be given a number of troops according to the player count, and then will take turns claiming territories until all of them are occupied. All players will be given a bonus objective card, which will be kept secret from the other players. After the initial deployment phase, the game will proceed as normal. A turn consists of acquiring and placing new armies, attacking if the player chooses to do so, and finally moving armies around the game board. The amount of armies a player acquires at the start of the round is determined by the value of the territories they currently control, the value of the continents they currently control, and any objective cards they have accomplished. After acquiring these armies, the player can choose to attack any enemy territory that neighbors one of their own. This action can be repeated as many times as the attacker chooses, as long as there is more than one troop in the attacking territory. The attacking and defending is done via dice rolls, with both the attacker and the defender rolling dice equal to their number of troops in either the attacking or defending territory, respectively. The attacker can roll a maximum of three dice, while the defender can only roll two. The dice rolls are then compared die by die, with the highest rolls of both sides going against each other, and the outcome of each of the rolls are determined separately. In the event of a tie, the defender wins. The attacker captures the enemy territory by eliminating all defender armies. In this case, the attacker has to move a number of troops that is at minimum equal to the number of dice they rolled. A territory must never be empty in the course of the game, so the attacker must leave at least one soldier behind. The defender wins when the attacker refuses to attack again, or if the attacker goes down to a single army in the attacking territory. After the attack phase, the player can choose to move their troops between any friendly territory that is connected. After this last phase, the turn passes onto the next player. The winner of the game is the player who manages to capture all the territories and eliminate all players.

Figure 17. Help Menu GUI Mock-up

f.Credits Screen

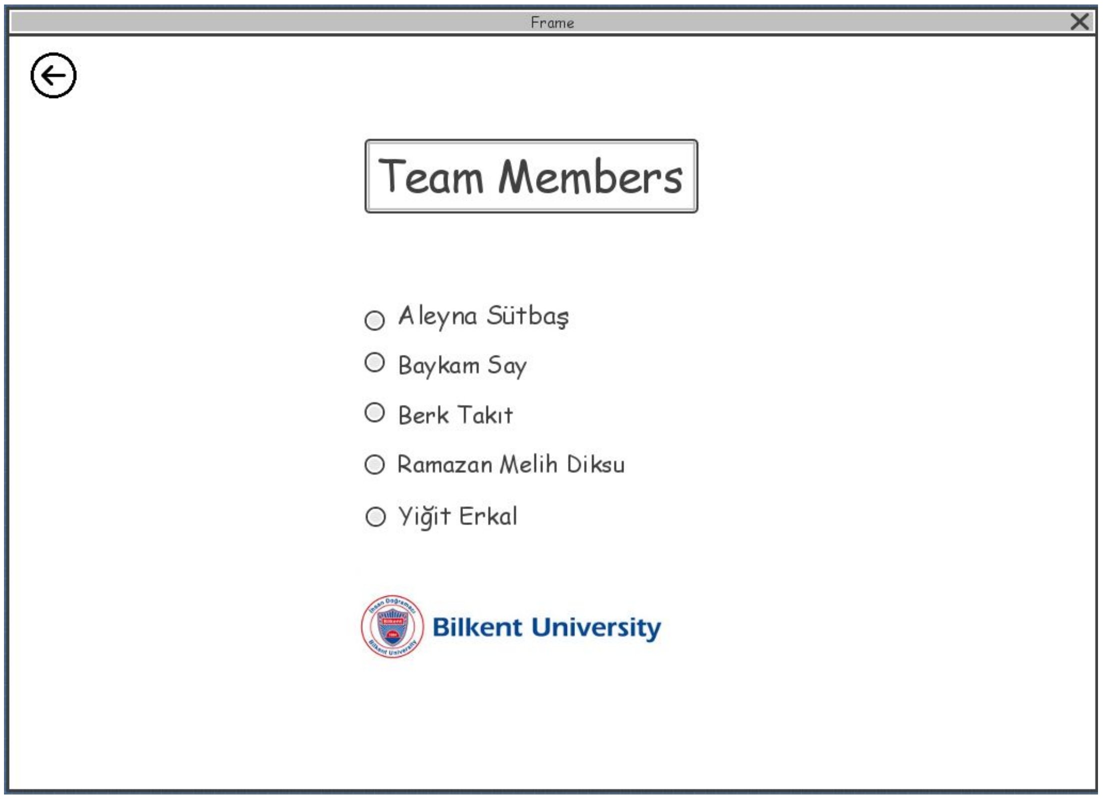
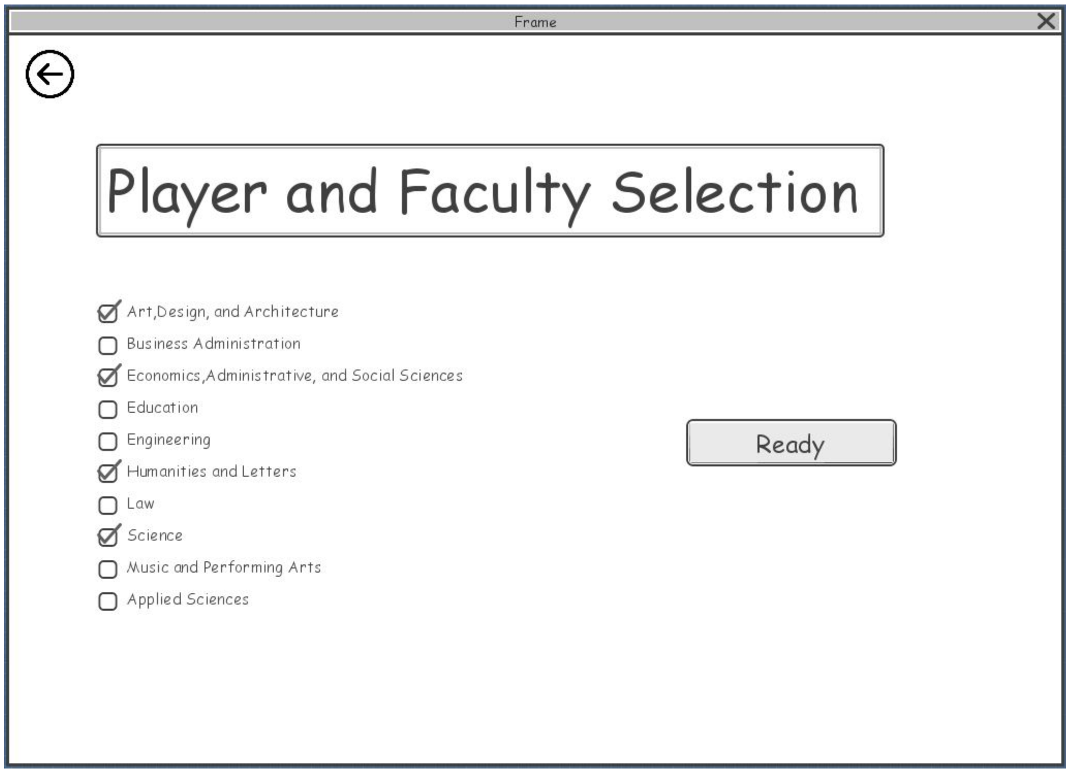


Figure 18. Credits Screen GUI Mock-up

g.Character Selection Screen



- [✓] Art,Design, and Architecture
- [ ] Business Administration
- [✓] Economics,Administrative, and Social Sciences
- [ ] Education
- [ ] Engineering
- [✓] Humanities and Letters
- [ ] Law
- [✓] Science
- [ ] Music and Performing Arts
- [ ] Applied Sciences

Figure 19. Faculty Selection Screen GUI Mock-up

h.Main Game Screen

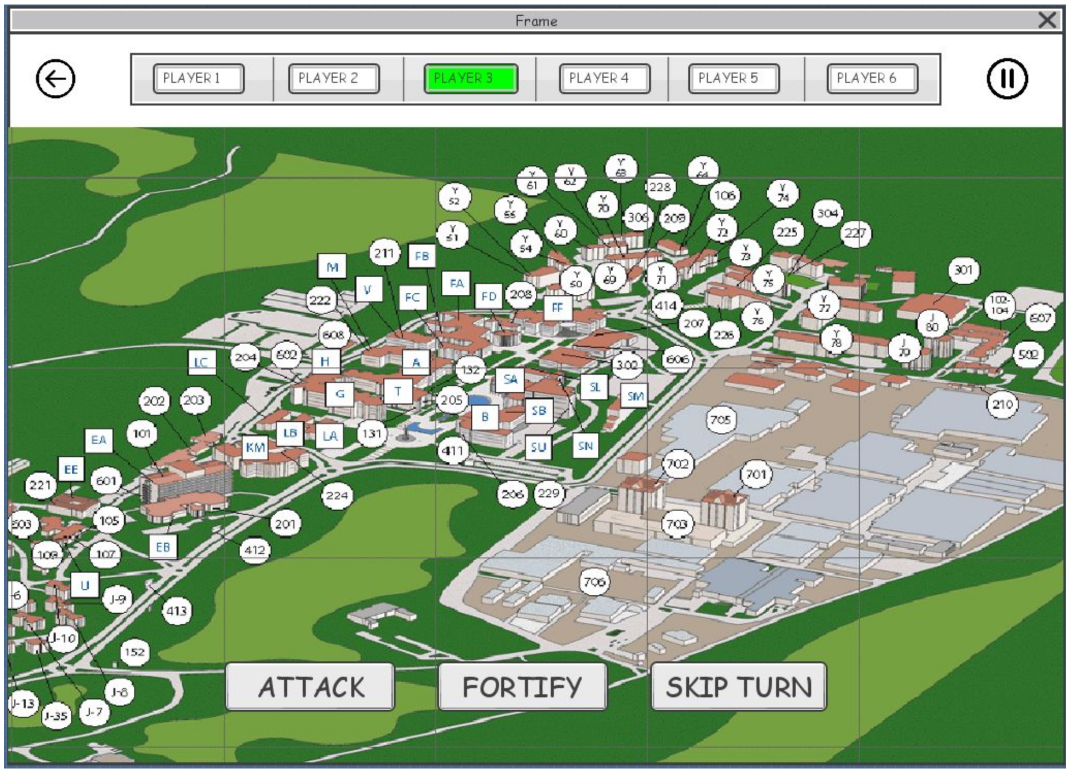


Figure 20. Main Game GUI Mock-up

i.Pause Game Screen

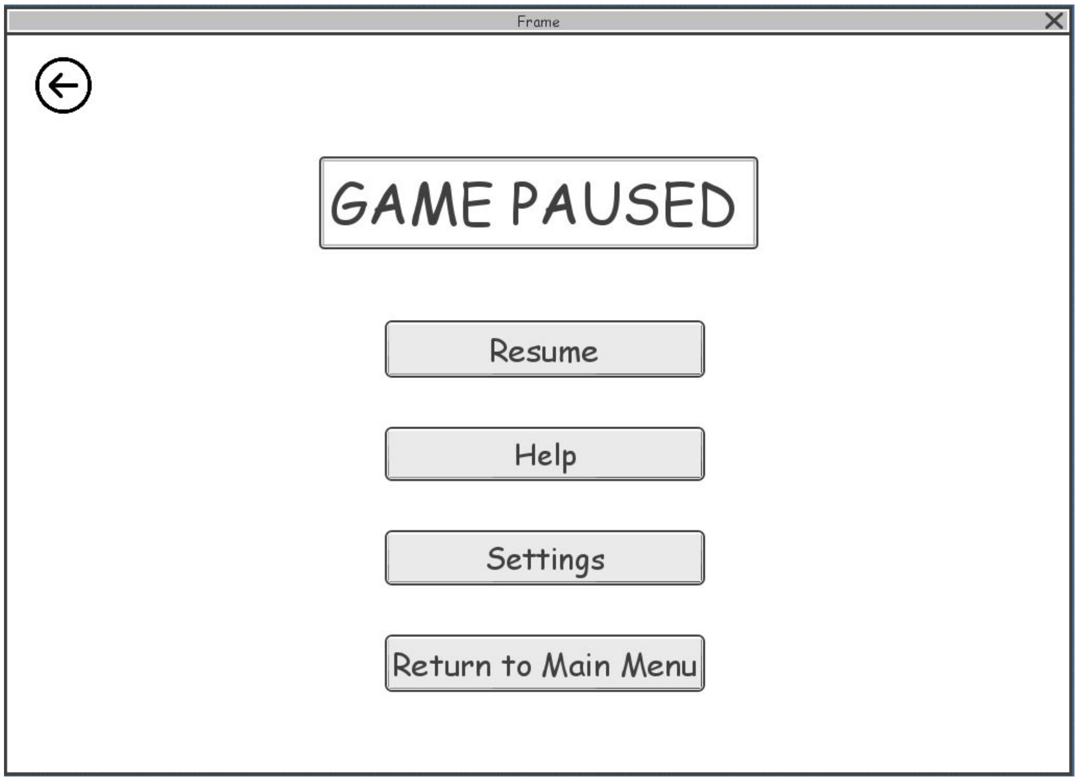


Figure 21. Pause Menu GUI Mock-up

j.End Game Screen

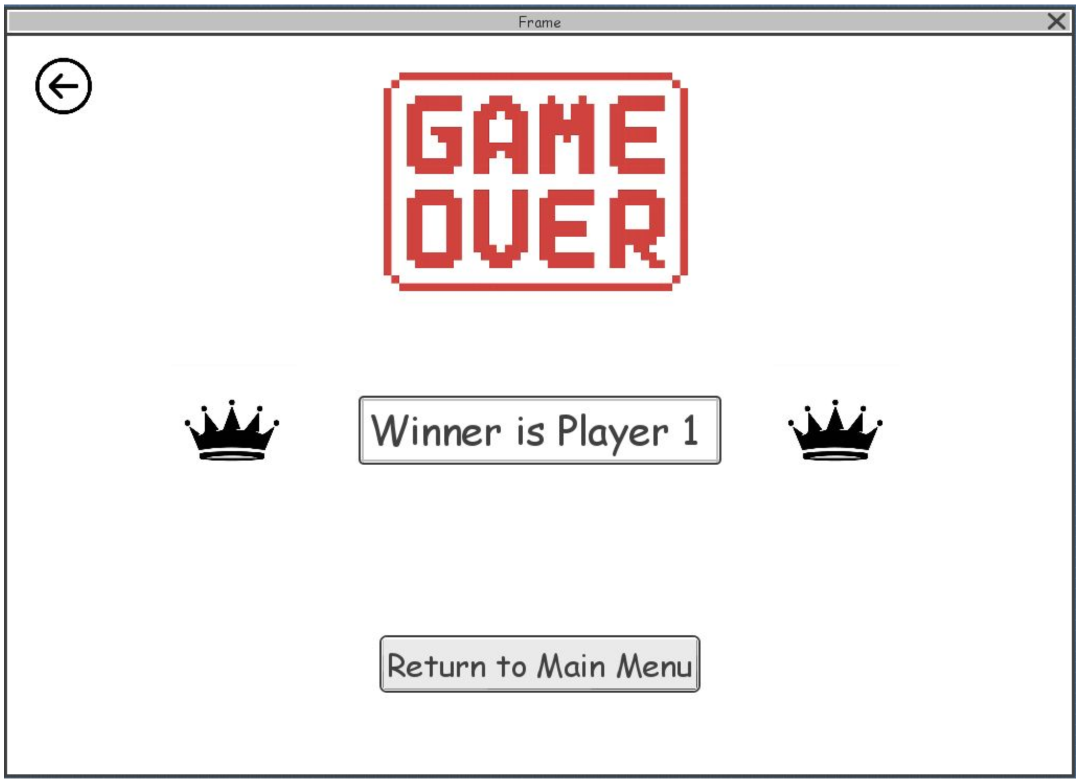


Figure 22. End Game Screen GUI Mock-up