Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Term Project

CS 319 3G: RISK

Analysis Report

Group Members:

Işık Özsoy 21703160

Defne Betül Çiftci 21802635

Burak Yetiştiren 21802608

Alp Üneri 21802481

Mustafa Hakan Kara 21703317

Instructor: Eray Tüzün
Teaching Assistant(s): Elgun Jabrayilzade, Emre Sülün, Barış Ardıç

Progress/Final Report

Oct 28, 2020

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object-Oriented Software Engineering course CS319.

Table of Contents

Analysis Report

CS 319 3G: RISK

1 Introduction

Risk is a strategy board game aimed for two to six players of diplomacy, conflict, and conquest. The standard version of the game consists of a board in which a map of the world is depicted. The map is divided into forty-two territories which are grouped into six continents. Turns rotate among players in control of armies with which they attempt to capture territories from other players via battles whose results are determined by dice rolls. The number of dice a player is able to attack or defend with is dependent on the number of soldiers that player controls in the applicable territories. The classic goal of the game is to occupy every territory on the board and achieve total global domination. Both formal and informal alliances may be formed during the course of the game.

Our project is based on RISK Global Domination, however some certain features will be added/removed, such as:

- We plan on allowing players to be able to leave territories in their control unguarded should they choose to.

- We plan on allowing players to set the goal of capturing a continent before the game starts and receive rewards should they be able to do so.

- We plan on implementing a rock-paper-scissors based combat system where more skill is involved as opposed to dice rolls.

- We plan on implementing an alliance mechanism where the two allied players will be allowed to go through their ally's territories to attack an opponent's territory. The allies will not be able to attack each other for the duration of their alliance. This implementation will not be available if only 2 players are playing.

- We plan on implementing an airport mechanic into the game. A player will be able to build an airport on his territories by paying a cost, which will allow them to attack territories of up to three neighbors away as opposed to only being able to attack neighboring territories.

2 Current System

In the original game, players are required to leave at least one unit in any territory that they are in control of. We plan on allowing players the ability to leave territories they currently control unguarded so as to move the soldiers on that territory to any other connected territory that they control. This would make it so that the territory previously controlled by the player is left empty, such that any other player with a connected territory can capture the empty one without any battle required.

We plan on having the players be able to declare a target continent that they will aim to capture before territories are dealt and the game begins. This would make it so should that player be able to actually conquer that continent they will receive some additional rewards in addition to the rewards they would have received for simply conquering that continent.

We are planning on implementing a combat system whose results are determined by something more skill based such as rock-paper-scissors as opposed to just dice rolls. This could serve to eliminate the frustration caused by unlucky dice rolls and add another layer of complexity/depth to the game.

In the original game, there is no official system of alliances but the player can form alliances nonetheless. One plan of ours is to implement an official system of alliances, which will not be available when only 2 players play, which will benefit the user in terms of reaching territories of their allies and performing an attack on an opponent through their ally's territory, potentially creating an environment where more strategies are supposed to be formed. The alliances will not be able to attack each other (as they are able to in the original game) nor will they be able to stay as allies for the entire duration of the game (as this would make the game end with no winners). This new implementation could add more complexity and intrigue to the game as well as make the players compose more strategies with more possible scenarios and interact with the other players more throughout this whole process.

We also plan on having the players be able to attack non-neighboring territories by building an airport on their territories by a cost of 5 soldiers (could be changed later on). By attacking through a plane, a player will be able to attack a territory which is up to 3 territories away (this detail could be changed as well).

3 Proposed System

3.1 Overview

We will be sticking to the global domination mode of the original game for the most part, the goal of any player in order to win the game will be to capture all of the individual territories found on the game map. However, we will be adding several new features to the game and will as well be revamping the combat mechanics.

In our version of the game, the result of combat will not be determined by dice rolls, but with a game of rock-paper-scissors. This is to make the combat system feel more skill based as opposed to just luck based, as rock-paper-scissors has an element of strategy to it instead of just rolling a die. In the original game, the number of dice a player is able to roll is dependent on the number of soldiers in their possession. This makes it so a player with more soldiers has a higher chance of conquering a territory. We have decided to keep this element of having a higher chance when commanding more soldiers by making it so when a player loses a round of rock-paper-scissors, a single unit of the soldiers that they are attacking/defending with will perish. Thus, the more units of troops a player has control of, the more chances they will have of winning a round of rock-paper-scissors as they will be able to play more rounds; making them more likely to conquer the territory.

We will also be adding four additional features on top of revamping the combat system. These four features are the ability to leave a territory in your possession unguarded (in the original game you must have at least one unit of soldiers on each territory you control), the ability to form alliances with other players and be able to attack territories while going over a territory belonging to your ally, the option to declare a goal continent at the start of the game which will give you bonus troops if you are able to conquer that whole continent, and the option to build an airport at any of the territories under your control; which will cost five units to build but will allow you to attack territories of up to three units away from that territory.

The game will start with the players first declaring a goal continent. This will make it so if they are successful in conquering that whole continent, they will get bonus troops in addition to the ones they will have gotten from having conquered a continent. After everyone has declared a goal continent, we will move on to territory allocation. This is the part where players will choose their starting territories. One by one, players will be choosing a single territory at a time which will be added to their starting territory list. The first player will choose a single territory, followed by the second player, and the third et cetera until no more territories remain. When a player chooses a territory in this step, they will place one unit on there to make it so they control that territory.

After all the territories are dealt (as in the starting territories have been finalized), there will be at least one unit of soldiers on all of the territories as well as additional soldiers under players' control that have been yet to be placed. Next will come the phase where players will place these soldiers under their control onto the board, one by one. After all of their starting soldiers have been placed, the set-up portion of the game will have been completed and play may begin in proper.

The game will be played in turns, with each player's turn consisting of three distinct phases. Before these three phases, however the turn player will have the ability to propose an alliance to any other non-allied player, as well the chance to accept or decline any alliances proposed to them. Accepting an alliance will prevent the two allying players from attacking each other's territories, but will give them the ability to pass over each other's territories to attack any other player neighboring their ally's territories. When a player proposes an alliance to another player, that player will have the ability to respond to the offer at the start of their turn. A player will be able to propose a single alliance to a single other player per turn. After proposing and/or accepting/declining any pending alliance offers, the player will move on to the first phase of their turn, which will be the troop allocation phase.

At the start of each of their turns, players will receive a certain number of troops depending on the number of territories and continents under their control. At the start of each turn, they will be able to place these troops in any fashion that they would like onto any territories under their control. After they have allocated the troops that they have received at the start of their turn will come the attack phase.

During the attack phase, players will be able to attack any territory controlled by any non-allying players neighboring any of the territories under their control with the soldiers present on that territory. A player may choose not to attack any territory during their attack phase, yet also has the option to attack to their heart's content. Once a player decides to attack any territory, the two players controlling the battling territories will play rock-paper-scissors to determine the victor of the battle. If the attacking side wins, the attacker will claim ownership of the territory and will choose how many soldiers to transfer to the territory they have just conquered. If the defending side wins, they will get to keep control of their territory. The attacking side wins by wiping out all of the soldiers on the defending side, and the defending side wins when the attacking side only has a single unit left on the territory they are attacking from.

After the player chooses to end their attack phase will come their fortify phase. At this phase players will be able to pick any number of troops from any of the territories under their control and move those troops to any other neighboring territories that they are also in control of. This fortification step will only be allowed to be done once, the player will not be able to completely redistribute all of the soldiers under their control at the end of each of their turns.

After the troop allocation, attack, and fortification phases are done, a player's turn will conclude and play will continue with the turn passing to the next player. If at any point a player runs out of soldiers (as in they lose all of the territories they control and thus have zero soldiers in their disposal), they will have lost the game and will continue on as spectators. If also at any point a player succeeds in controlling all of the territories on the board, they will have achieved global domination and will have won. The game ends if and only if when a player achieves global domination.

3.2 Functional Requirements

3.2.1 Starting a New Game

To start a new game, the players will have to specify the number of players that will be playing, will choose a color that will represent territories under their control, and will designate a continent as their goal continent, which will give them bonus troops per turn if they are able to conquer that specified continent. After all of this information has been decided on, the START GAME button will be pressed and the game will start with the territory allocation step. This feature will be of high priority.

3.2.2 Territory Allocation

The game will start with each of the players selecting their starting territories one by one. The first player will first choose a territory to add to their starting territory list, then the second player will choose another territory, then the third, et cetera. When a player chooses a territory to add to their starting territory list, they will place a single troop on that territory to indicate that they are in control of that region. This will also make it so there will be at least a single troop in each of the territories at the start of the game. After all of the territories have been chosen and have at least a single unit of soldiers on them, play will move on to the distribution of starting troops. On the territory allocation section the players will only have the option to designate an uncontrolled territory to add to their starting territory list, once they have chosen and confirmed their selection a single unit of soldiers will automatically be placed from their starting troops onto that territory. This feature will be of high priority.

3.2.3 Distribution of Starting Troops

After all of the territories have been chosen, the players will still have an equal number of starting soldiers at their possession that they will not have yet distributed. At this phase they will be distributing those remaining soldiers. This will also be done in a similar manner to the territory allocation step, as in players will one by one choose a territory under their control to add a single unit of troops to. They will not be limited to distribute any of their remaining soldiers in any way, since they will already have at least a single unit on all of the territories under their control during the territory allocation step, they will be allowed to place all of their remaining soldiers in any manner of their choosing. After all of the players have distributed all of their starting soldiers, play will begin in the format of turns. At this step players will only have the option to designate the territory under their control to which they would like to place their single unit of soldiers per turn. For example, Player 1 will choose Territory A and a single unit of their starting troops will be placed there, then Player 2 will choose Territory B and a single unit of their starting troops will be placed there et cetera. This feature will be of high priority.

3.2.4 The Turn of a Player

The game will then continue in the form of turns. The players will consecutively conduct their turns one by one until they lose the game or the game is over. If a player loses the game, which occurs when they have lost control of all of their territories and have no remaining units of troops, they will no longer have their turns and play will continue with the remaining players. The turns of players are separated into three distinct phases, the troop allocation phase, the attack phase and the fortify phase. The player will be conducting these phases in this order, and will have a button labeled NEXT PHASE at the bottom of their screen which they will press in order to move from one phase to the next. After they have completed all of their phases in their respective turns, the next player's turn will begin. This feature will be of high priority.

3.2.4.1 Alliance Requests

Alliance requests will be available to send for the duration of the Troop Allocation phase, in which the player will be able to propose an alliance by clicking on the pop-up "Send an alliance request" option that shows up when the player clicks on one of the user's username shown on the screen. If they choose to propose an alliance to a player, that player will be able to respond at the start of their next turn to that proposal. If the player accepts the alliance, the two (or more) allied players will no longer be allowed to attack each other's territories, but will be allowed to pass over allied players' territories to attack their neighboring territories with their own soldiers. If at any point there remain only two players in the game, all alliances will be disbanded. At this phase players will also be asked if they'd like to disband any of their current alliances or remain allies. A player will be allowed to propose an alliance to only a single other player that they are not currently allies with, yet will have the option to disband as many previous alliances as they'd like. They will press the next phase button on their screen to move on to the troop allocation phase. This feature will be of low priority.

3.2.4.2 Troop Allocation Phase

During the troop allocation phase, players will be able to allocate the troops that they will have received at the start of their turns. At the start of each of their turns, players will receive a number of additional troops depending on the number of territories and continents that they are in control of. Every continent will have a set number of troops that they award players controlling them, as well as a different set number of troops that they will also award if the player controlling them has designated them as their goal continent at the start of the game. After the number of troops that the turn player will receive is calculated, they will allocate these troops to territories under their control. While doing so, they will not be limited in any way to allocate these troops that they have received, they can place as many of their newly received soldiers to any territory under their control as they'd like. They can place all of the soldiers that they have received on a single territory, or may distribute them among all of the territories under their control, or do anything in between. Yet they will be forced to allocate all of the soldiers that they have received before they are able to move on with the game. At this step they will also be able to build an airport on any of the territories under their control. After they have allocated all of the soldiers that they have received during the start of their turn, and have built all of the airports that they'd like to build, the player will press the NEXT PHASE button to move on to the attack phase. They will not automatically be moved on to the attack phase when they place all of the soldiers that they have received, assuming that they might still want to build an airport. This feature will be of high priority.

3.2.4.2.1 Building an Airport

During the troop allocation phase, the players will be able to click any territory under their control with at least six units of soldiers present to build an airport there. Building an airport will cost five units of soldiers, yet will allow a player to attack territories of up to three units away from the territory on which they have built the airport. When a territory in which an airport is present is attacked, the airport will provide three units to aid with the defense of the territory. When a territory with an airport is conquered by a different player, that player will be assuming control of the airport. This feature will be of low priority.

3.2.4.3 Attack Phase

After their troop allocation phase, players will move on to their attack phase. During this phase, they will be able to attack any neighboring territories controlled by any non-allied player from their territories with at least two units present. This is because should they succeed in conquering the territory that they are attacking, they will have to transfer at least a single unit there while leaving at least another different single unit at the territory from which they are attacking. Although the original game of risk handles battles via dice rolls, we have decided to implement a rock-paper-scissors based combat system in order to add some more skill-based combat elements into the game. Once a player declares combat to another player, they will start playing rock-paper-scissors to commence their battle. A battle will be considered over if the attacking side has at most a single soldier left, the defending side has at most zero soldiers left, or the attacking party decides to stop their attack. After every single round of rock-paper-scissors, if the attacking side decides that they no longer wish to attack the selected territory, they may quit attacking there. This does not stop the turn player from attacking any other territory during the rest of their attack phase, however. After the turn player decides that they no longer wish to attack any other territories, they will press the NEXT PHASE button to move on to the fortify phase. They may also skip this attack phase completely if they do not wish to attack any territory that turn. This feature will be of high priority.

3.2.4.3.1 Rock-Paper-Scissors

Although the original game uses a dice-based combat system, we have decided to use a rock-paper-scissors based combat system in our design. This decision was made so that we could add another skill-based layer of complexity to our game. However, a challenge we had to face was the issue of how we were going to allow two players to play rock-paper-scissors on the same screen without seeing what the other player had chosen. We decided to implement the following system to solve this issue, the attacking player will use the keys A-S-D to choose between rock-paper-scissors and the defending player will use NUMPAD 4-5-6 to do so. This will make it so neither of the players will be able to see beforehand what the other player has chosen. The result of a battle will be determined solely by rounds of rock-paper-scissors. After every round, if there is a draw either no soldiers will be killed or a single unit of troops will be killed from both sides (we have not yet decided and will experiment with both ideas to see which will produce smoother gameplay), and if any side loses they will lose a single unit of troops. We are also considering a system in which when there is a draw, the attacking side will lose a troop. This will be more in line with the original version of the game, where the defending side has a home field advantage in which they are considered to have won the battle when there is a draw with the dice. We are concerned that such an implementation may be too punishing to the attacking side as it is more probable to have a draw during rock-paper-scissors as opposed to the dice-based system. As stated in section 3.2.4.3, combat ends once the attacking player has at most a single unit of troops or the defending side has at most zero. This feature will be of medium priority.

3.2.4.4 Fortify Phase

During the fortify phase, players will have the option to move any number of soldiers from one territory under their control to any other neighboring territory (also under their control). This is to allow players a chance at redistributing their soldiers in anticipation of the turns of the following players. In this phase, players will also have the option to leave the territory from which they are transferring soldiers to any other neighboring territory under their control. This will result in that territory being under the control of no particular player, such that the next player wishing to conquer that land during their own attack phase can do so without any battle (rounds of rock-paper-scissors) required. After the fortify phase, the player will press the NEXT PLAYER button to end their turn. This feature will be of high priority.

3.2.5 Winning/Losing the Game

The game will be over once a single player has conquered all of the territories on the map, achieving global domination. This also includes any territories that have been emptied out during other players' fortify phases, such that there are no empty territories remaining at the end of the game (all territories must be under the control of a single player as in all territories must have at least a single unit of troops belonging to the same player). This is the winning condition of the game, as well as its finish condition. Once this happens, a screen displaying the winning player will be shown, and players will have the option to quit the game or start a new one.

A player loses once they run out of soldiers and thus have at most zero territories under their control. Once this happens, that player will have been eliminated and any and all of their subsequent turns will be treated as having been skipped, they will essentially now be spectating the game. This feature will be of high priority.

3.2.6 Changing Settings

At any time via pausing the game, or in the main menu, players will have the option of choosing the resolution of the game, whether they'd like background music or not, or how loud they'd like the game to sound. This will be done through a separate OPTIONS menu. This feature will be of low priority.

3.2.7 Pausing the Game

At any time during the turn of any player, players will have the option to press the ESC key to pause the game. This will pause the turn timer that a player has. Pausing will also allow players to quit the game, go back to the main menu, change any of the settings, or view credits. This feature will be of low priority.

3.2.8 Viewing Credits

At any time via pausing the game, or in the main many, players will have the option to view the credits of the game; where the names of the creators of the game will be written. This feature will be of low priority.

3.2.9 Cards

If during their turn a player is able to conquer at least one territory, they will receive a random card from a deck of cards. The deck will contain the following four types of cards; Infantry, Cavalry, Cannon, and Joker. There will be fewer Joker cards than the other three types of cards. Cards that a player earns will be kept in their possession until they decide to exchange them for soldiers (details of which will be explained below), or are eliminated. Once a player is eliminated, all of their cards will be returned to the deck and the deck will be shuffled. If a player is able to successfully conquer at least one territory during their turn yet the deck has run out of cards, they will not be able to get a card that turn. This feature will be of medium priority.

3.2.9.1 Exchanging Cards for Troops

If a player is in possession of three cards of the same type (with the Joker card being able to be substituted for any of the other three types), they will be able to exchange those cards for a number of extra troops. The number of troops that they will receive by doing so will be dependent on the type of cards they will be exchanging, with Infantry granting the least number of troops, Cannon granting the most, and Cavalry granting an amount between the two. The actual amounts will be decided later on and will be optimized to maximize the enjoyment of the game. This feature will be of medium priority.

3.2.10 Table of Prioritization

Feature	Priority
Starting a New Game	High
Territory Allocation	High
Distribution of Starting Troops	High
The Turn of a Player	High
Alliance Requests	Low
Troop Allocation Phase	High
Building an Airport	Low
Attack Phase	High
Rock-Paper-Scissors	Medium
Fortify Phase	High
Winning/Losing the Game	High
Changing Settings	Low
Pausing the Game	Low
Viewing Credits	Low
Cards	Medium
Exchanging Cards for Soldiers	Medium

3.3 Non-Functional Requirements

3.3.1 User Interface and Human Factors

The game will support resolution values ranging from 800*600 to 1920*1080. This will be done to accommodate monitors and screens of different resolutions. The map will display who controls what territory by painting different territories the color of the player controlling them, making it easy to understand which territories any player controls at any time. The players will be choosing a color at the start of the game which will then be used as the color of any territories under their control. Each territory will also have a number in its middle, indicating the number of troops present on that territory. Empty territories will be painted grey with a 0 on them, making them easily identifiable.

3.3.2 Documentation

There will be a number of reports written throughout the development of the game detailing topics such as requirements elicitation and analysis, the structure of the game such as the classes and their interactions, et cetera. We also plan on having a help screen in our game, and may also place a link where players will be able to access the official guide of the original version of the game, published by Hasbro.

3.3.3 Hardware Considerations

The game will be simple enough that any computer able to run an operating system and Java should be able to run our game. Play will however be controlled for the most part using a mouse, except for rock-paper-scissors parts which will be played using a keyboard. Therefore, a mouse and keyboard system is required in order to play our game.

3.3.4 Performance Characteristics

The game will be played on a 2D board with the display of a world map on top. From this display players will be able to see which player controls which territories via the color of that territory, as well as how many units of troops any player has on any territory via numbers on the territories. The game will run in 60 frames per second by default, which is considered the minimum standard for smoothness for gaming, yet may occasionally drop down to 30 FPS as well, when a particularly cumbersome operation is being performed in the background. This should occur quite rarely (about a maximum of five times per game) in order to make sure the immersion of the gaming experience is not interrupted constantly. The control inputs will be executed without any noticeable delay, which means a response time of approximately 0.20 to 0.25 milliseconds. Since the game will be played on a single screen, no networking performance requirements are needed to be specified.

3.3.5 Error Handling and Extreme Conditions

As we will only be allowing players to control the game using a mouse, except for the rock-paper-scissors battles; the player should not be able to enter difficult to deal with inputs to our system. During the writing of the actual code of the game, we will be utilizing try-catch blocks when necessary to be able to deal with possible errors that may occur. During rock-paper-scissors battles, we will only be listening to the six relevant keys (two each for either player's rock/paper/scissors) and perhaps the ESC key. There should be no extreme conditions with regards to memory leak as we will be writing our game using Java, which features an automatic garbage collection system.

3.3.6 System Interfacing

The player will be using the mouse for the most part while playing the game, and the menus will not be able to be navigated via arrow keys, the player will have to click the button that they wish to be clicked. The only time where players will be using the keyboard is when they are conducting battle, and they will be using only six keys A-S-D and NUMPAD 4-5-6 to do so. This will be done during rounds of rock-paper-scissors. No matter how novice, players will be expected to have no trouble playing the game assuming that they are familiar with the use of a mouse and keyboard. As there will be no real time events, there will be no timed execution required during gameplay, no player should have a hard time getting to grips with the controls of the game.

3.3.7 Quality Issues

The game will be expected to include all of the features deemed of high priority in the previous section, and will also be expected to contain all of the other features as well. However, if we find that we will not be able to implement all of the features we have specified above, we may choose not to implement all of the lower priority ones.

3.3.8 System Modifications

Our game will not need any particular system modifications in order to be played as it will only require a mouse and a keyboard. We reckon that our game will take a memory space of less than one gigabyte, so any machine has at least a gigabyte of empty space should be able to run our game.

3.3.9 Physical Environment

Our game will be played on a single computer, where the turn player will get to assume control of the computer during their turn. This will only create a potential problem during the rock-paper-scissors battles, and we have come up with the following solution: The attacker will use the A-S-D keys to choose either rock/paper/scissors and the defending player will use NUMPAD 4-5-6 to do the same. This choice will be done at the same time, allowing neither player to see what the other player has chosen before making their own decision.

3.3.10 Security Issues

There will be no input or storage on sensitive information whatsoever onto our game, therefore there will be no security considerations with regards to our game. Moreover, as our game will not have any networking functionalities, the attack surface will be incredibly small, negligible even.

3.3.11 Resources and Management Issues

As our game will be developed using Java, which features an automatic garbage collection system, resource management issues will be handled by Java and will therefore not be a problem.

3.4 Pseudo Requirements

3.4.1 Language

The game will be developed using solely object-oriented programming languages.

3.4.2 Version Control System

Git and GitHub will be used as the version control system and the code repository of the project.

3.5 System Models

For a higher resolution view of any of the diagrams given, please see the vector files provided with our submission.

3.5.1 Scenarios

Use case name	SendAllianceRequest
Participating actor	Initiated by Player
Flow of events	1. The player clicks on the alliance menu 2. S/he presses the send alliance request button 3. S/he chooses the player to send the request to
Entry condition	This use case is initiated in the beginning of the soldier placement phase if the player wishes to create an alliance.
Exit condition	It is exited whenever the request is sent.
Quality requirements	· Interaction does not stop the game · This use case can only be done in the soldier placement phase

Use case name	Attack
Participating actor	Initiated by player
Flow of events	1. Attacking player chooses the territory that s/he wants to attack out of the territories s/he is able to attack 2. Attacker chooses the number of troops s/he wants to attack with 3. A rock papers scissors game is initiated with the enemy 4. If the attack is successful attacker chooses the number of troops s/he wants to transfer to the new territory
Entry condition	This use case is extended by SelectNumberOfSoldiers use case and includes ChooseTerritory and RockPaperScissors use cases. It is initated by the user whenever the player is in the attack phase.
Exit condition	· Player runs out of attack time · Player skips to the next phase
Quality requirements	· There is a time limit for the players to attack · Users can attack several times · Users can skip this phase

Use case name	ChooseTerritory
Participating actor	Communicates with player
Flow of events	Player chooses from the available territories to attack
Entry condition	Player is in the attack phase
Exit condition	Player clicks on a territory
Quality requirements	· The available territories to attack are shown clearly to the user

Use case name	MoveSoldiers
Participating actor	Initiated by player
Flow of events	1. Player chooses a region to move his/her soldiers 2. Player chooses how many soldiers s/he wants to move
Entry condition	This use case is initiated in the last phase of a player’s turn. It is extended by SelectNumberOfSoldiers and ChooseRegion use cases.
Exit condition	· Player moves his/her soldiers · Player ends the phase · Player runs out of time
Quality requirements	· There is a time limit for the players to move soldiers · Players can move soldiers only one time · Players can skip this phase

Use case name	InteractWithAllianceRequest
Participating actor	Communicates with player
Flow of events	1. Player sees the alliance request 2. S/he chooses how to interact with the request
Entry condition	This use case is initiated automatically when the player is in the beginning of the soldier placement phase, if s/he has been sent an alliance request.
Exit condition	· Player chooses an option
Quality requirements	· Request is a window easy to interact with. · Interaction does not stop the game · This use case can only be done in the attack phase

Use case name	AddSoldiers
Participating actor	Initiated by player
Flow of events	1. Player is given some number of soldiers according to regions s/he owns 2. If the player has cards s/he can choose to use them if s/he has the correct set of cards 3. Player places the soldiers to his/her regions until s/he runs out of troops
Entry condition	This use case is initiated in the first phase of a players turn. It is extended by SelectNumberOfSoldiers and ChooseRegion use cases.
Exit condition	· Player places all of his/her troops · Player runs out of time
Quality requirements	· There is a time limit for the players to move soldiers · If the time limit is reached troops are automatically placed · Players cannot skip this phase; they have to place all troops

Use case name	UseAlliesTerritory
Participating actor	Inherited from ChooseTerritory use case
Flow of events	Player presses attack button
Entry condition	A territory is selected such that · it is not the neighbor territory of the player · the territories between the selected territory and the territory from where the attack is initiated belongs to one of the allies
Exit condition	The attack is initiated
Quality requirements	· This use case is initiated only when the selected territory satisfies the requirements

Use case name	FlyToDistantTerritory
Participating actor	Inherited from ChooseTerritory use case
Flow of events	Player presses attack button
Entry condition	A territory is selected such that · it has at most three territories between the selected territory and origin territory
Exit condition	The attack is initiated
Quality requirements	· This use case is initiated only when the selected territory satisfies the requirements

Use case name	ChooseNeighborTerritory
Participating actor	Inherited from ChooseTerritory use case
Flow of events	Player presses attack button
Entry condition	A territory is selected such that it is the neighbor of the origin territory of attack
Exit condition	The attack is initiated
Quality requirements	· This use case is initiated only when the selected territory satisfies the requirements

Use case name	SelectNumberOfSoldiers
Participating actor	Communicates with player
Flow of events	1. An AddSoldiers, Attack, or MoveSoldiers use case is initiated 2. Player selects the number of soldiers s/he wants to interact with
Entry condition	This use case extends AddSoldiers, Attack, and MoveSoldiers use cases. It is initiated when the user initiates an attack, wants to move soldiers, or adds soldiers at the beginning of his/her turn
Exit condition	Player selects the number of soldiers
Quality requirements	· Player cannot select all of his/her soldiers · Player is presented with a dropdown menu to select the number of soldiers

Use case name	ChooseRegion
Participating actor	Communictes with player
Flow of events	1. Player is either in the first or third phase 2. S/he is presented with the region(s) 3. Player chooses a region
Entry condition	Player is either in the first or third phase
Exit condition	Player chooses a region
Quality requirements	· Player is presented clearly the regions that s/he can choose · This use case is different from the ChooseTerritory use case, as in that use case player has three options

Use case name	RockPaperScissors
Participating actor	Player and enemy
Flow of events	1. Player initiates an attack 2. A rock paper scissors game is initiated 3. After each round one of the troops of the loser is killed 4. Another rock paper scissors game is initiated until one of the sides are left with no troops
Entry condition	An attack is initiated
Exit condition	A round is over
Quality requirements	· The game has three possible outcomes for player (win, lose, draw) · The game is initiated between the attacker (player) and the defender (enemy) · Player uses the keyboard keys (A, S, D) for rock, papers, scissors respectively, while the enemy uses (4, 5, 6)

Use case name	SelectMove
Participating actor	Communicates with player
Flow of events	1. A rock, paper, scissors game is initiated 2. Player chooses one of the three options to choose (rock, paper, scissors)
Entry condition	A rock, paper, scissors game is initiated
Exit condition	Player chooses a move
Quality requirements	· The players can use the buttons of the keyboards on two different sides of the keyboard, because the game is played on the same computer synchronously

Use case name	KillSoldier
Participating actor	Communicates with the game
Flow of events	1. A round of a rock papers scissors game is over 2. There is not a tie 3. A troop of the loser is killed
Entry condition	A round of a rock papers scissors game is over and the round is not a tie
Exit condition	One of the troops of the loser is killed
Quality requirements	· Only one troop is killed · If there are no troops left the game is over · In a case of tie no troop is killed · This use case does not communicate with player

Use case name	PauseGame
Participating actor	Initated by player
Flow of events	1. The game is being played 2. Player presses the “pause game” button
Entry condition	Player presses the “pause game” button
Exit condition	Player presses the “continue playing” button
Quality requirements	· Time is frozen when player pauses the game · A button is provided to pause the game

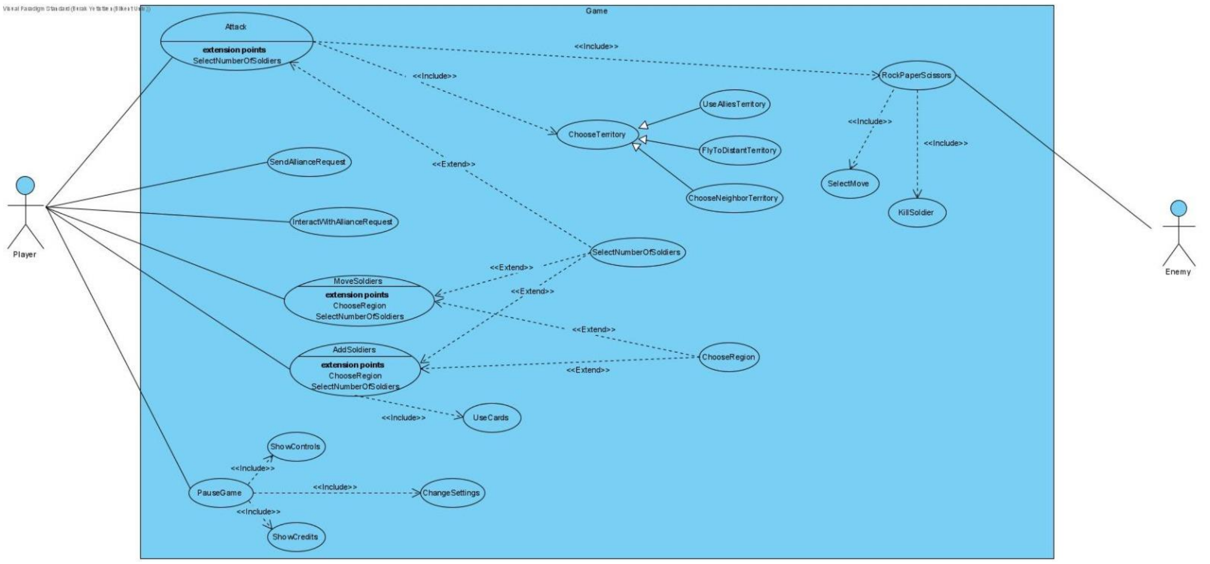
Use case name	ShowControls
Participating actor	Communicates with player
Flow of events	1. Game is paused 2. Player presses the “show controls” button
Entry condition	In the pause menu player presses the “show controls” button
Exit condition	Player presses “back” button in the controls menu
Quality requirements	· The controls of the game are presented to the player in the controls menu

Use case name	ShowCredits
Participating actor	Communicates with player
Flow of events	1. Game is paused 2. Player presses the “show credits” button
Entry condition	In the pause menu player presses the “show credits” button
Exit condition	Player presses “back” button in the credits menu
Quality requirements	· The credits of the game are presented to the player in the credits menu

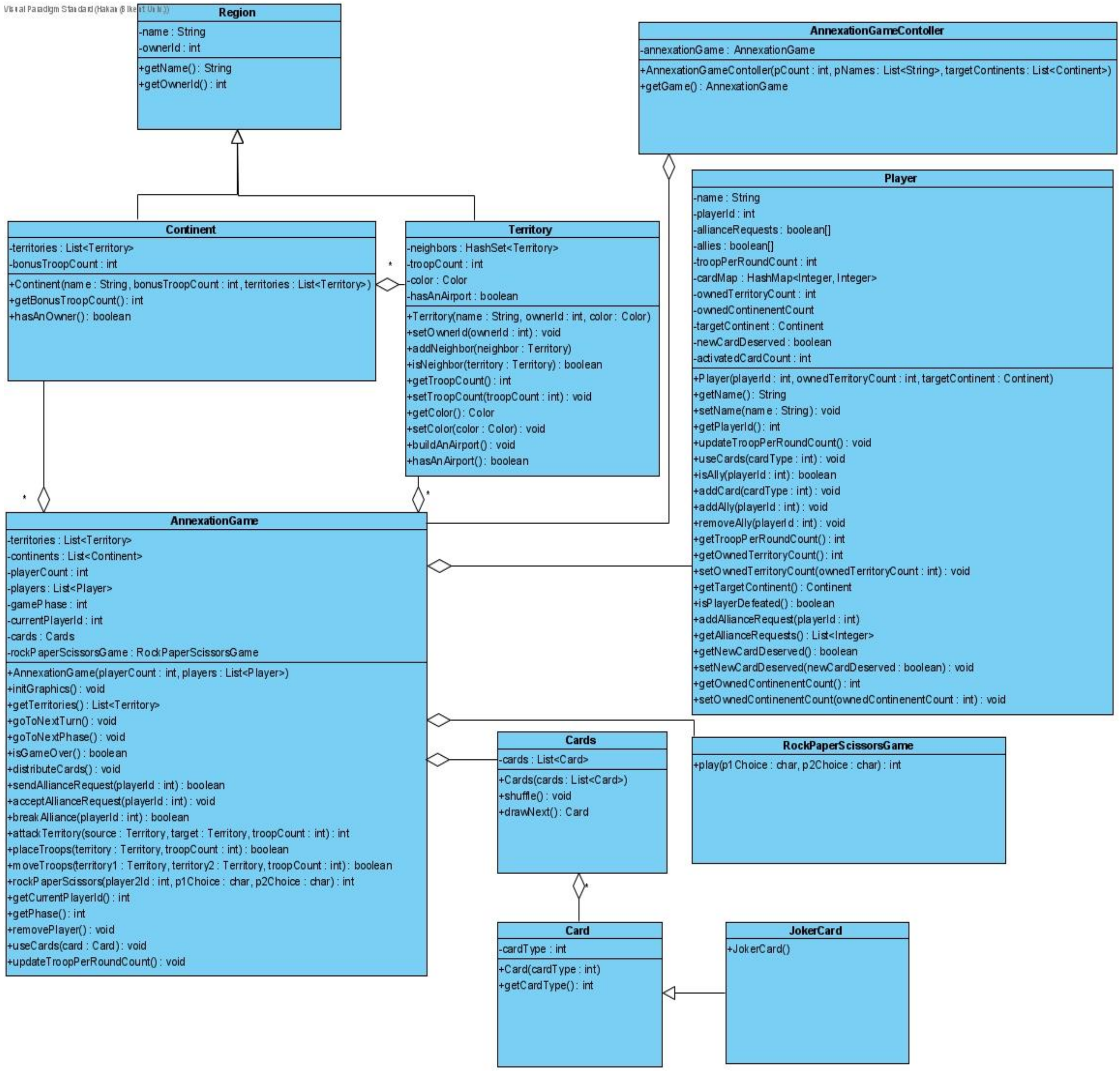
Use case name	ChangeSettings
Participating actor	Communicates with player
Flow of events	1. Game is paused 2. Player presses the “change settings” button
Entry condition	In the pause menu player presses the “change settings” button
Exit condition	Player presses “back” button in the credits menu
Quality requirements	Player is presented with the options s/he can change in the “change settings” menu

Use case name	UseCards
Participating actor	Initiated by the player
Flow of events	1. Player is in the first phase 2. Player chooses the option to use his/her cards to gain more troops
Entry condition	Player presses the button, which enables him/her to gain more soldiers if s/he has the proper set of cards
Exit condition	After pressing the button player is provided with the soldiers if s/he has the right set
Quality requirements	· Player can only use cards in the first phase of his/her round · Only the right set of cards provides additional troops

3.5.2 Use-Case Model



3.5.3 Object and Class Model


Visual Paradigm Standard (Hakan (Bilkent Univ.))
Region
-name : String
-ownerId : int
+getName() : String
+getOwnerId() : int
AnnexationGameContoller
-annexationGame : AnnexationGame
+AnnexationGameContoller(pCount : int, pNames : List<String>, targetContinents : List<Continent>)
+getGame() : AnnexationGame
Player
-name : String
-playerId : int
-allianceRequests : boolean[]
-allies : boolean[]
-troopPerRoundCount : int
-cardMap : HashMap<Integer, Integer>
-ownedTerritoryCount : int
-ownedContinenentCount
-targetContinent : Continent
-newCardDeserved : boolean
-activatedCardCount : int
+Player(playerId : int, ownedTerritoryCount : int, targetContinent : Continent)
+getName() : String
+setName(name : String) : void
+getPlayerId() : int
+updateTroopPerRoundCount() : void
+useCards(cardType : int) : void
+isAlly(playerId : int) : boolean
+addCard(cardType : int) : void
+addAlly(playerId : int) : void
+removeAlly(playerId : int) : void
+getTroopPerRoundCount() : int
+getOwnedTerritoryCount() : int
+setOwnedTerritoryCount(ownedTerritoryCount : int) : void
+getTargetContinent() : Continent
+isPlayerDefeated() : boolean
+addAllianceRequest(playerId : int)
+getAllianceRequests() : List<Integer>
+getNewCardDeserved() : boolean
+setNewCardDeserved(newCardDeserved : boolean) : void
+getOwnedContinenentCount() : int
+setOwnedContinenentCount(ownedContinenentCount : int) : void
Continent
-territories : List<Territory>
-bonusTroopCount : int
+Continent(name : String, bonusTroopCount : int, territories : List<Territory>)
+getBonusTroopCount() : int
+hasAnOwner() : boolean
*
Territory
-neighbors : HashSet<Territory>
-troopCount : int
-color : Color
-hasAnAirport : boolean
+Territory(name : String, ownerId : int, color : Color)
+setOwnerId(ownerId : int) : void
+addNeighbor(neighbor : Territory)
+isNeighbor(territory : Territory) : boolean
+getTroopCount() : int
+setTroopCount(troopCount : int) : void
+getColor() : Color
+setColor(color : Color) : void
+buildAnAirport() : void
+hasAnAirport() : boolean
*
*
AnnexationGame
-territories : List<Territory>
-continents : List<Continent>
-playerCount : int
-players : List<Player>
-gamePhase : int
-currentPlayerId : int
-cards : Cards
-rockPaperScissorsGame : RockPaperScissorsGame
+AnnexationGame(playerCount : int, players : List<Player>)
+initGraphics() : void
+getTerritories() : List<Territory>
+goToNextTurn() : void
+goToNextPhase() : void
+isGameOver() : boolean
+distributeCards() : void
+sendAllianceRequest(playerId : int) : boolean
+acceptAllianceRequest(playerId : int) : void
+breakAlliance(playerId : int) : boolean
+attackTerritory(source : Territory, target : Territory, troopCount : int) : int
+placeTroops(territory : Territory, troopCount : int) : boolean
+moveTroops(territory1 : Territory, territory2 : Territory, troopCount : int) : boolean
+rockPaperScissors(player2Id : int, p1Choice : char, p2Choice : char) : int
+getCurrentPlayerId() : int
+getPhase() : int
+removePlayer() : void
+useCards(card : Card) : void
+updateTroopPerRoundCount() : void
Cards
-cards : List<Card>
+Cards(cards : List<Card>)
+shuffle() : void
+drawNext() : Card
RockPaperScissorsGame
+play(p1Choice : char, p2Choice : char) : int
*
Card
-cardType : int
+Card(cardType : int)
+getCardType() : int
JokerCard
+JokerCard()

Class Name	Region
Properties	• String name: holds the name of the region. • int ownerId: holds the id of the region’s owner.
Methods	• String getName(): returns the name of the region. • int getOwnerId(): returns the id of the region’s owner.

Class Name	Continent (This class inherits Region class.)
Properties	• List<Territory> territories: holds the territories of which the continent is composed. Note that territories are only references, since every variable of non-primitive type in Java is a reference. • int bonusTroopCount: holds the number of troops that the continent gives to its owner as a bonus in each turn.
Constructor	• Continent(String name, int bonusTroopCount, List<Territory> territories): sets name, territories and bonusTroopCount to its parameters and assigns -1 to ownerId.
Methods	• int getBonusTroopCount(): returns the bonusTroopCount of the region. • boolean hasAnOwner(): If ownerId is different than -1, this method immediately returns true. Otherwise, it checks whether all the ownerIds of its territories are the same. If they are the same, it sets its ownerId to that and returns true. If not, it returns false.

Class Name	Territory (This class inherits Region class.)
Properties	• HashSet<Territory> neighbors: holds the neighbors of the territory. The reason why they are stored in a hash set because this way, checking whether a given territory is its neighbor or not takes O(1) time. • int troopCount: holds the number of troops in the territory. • Color color: holds the color of the territory • hasAnAirport: determines whether the territory has an airport.
Constructor	• Territory(String name, int ownerId, Color color): sets name, ownerId and color to its parameters.
Methods	• void setOwnerId(int ownerId): sets the region’s ownerId to its parameter. • void addNeighbor(Territory neighbor): adds the parameter to neighbors. • boolean isNeighbor(Territory neighbor): checks whether a given neighbor is a neighbor the territory. • int getTroopCount(): returns the number of troops in the territory. • void setTroopCount(int troopCount): sets the troopCount to its parameter. • Color getColor(): returns the color of the territory. • setColor(Color color): sets the color of the territory to its parameter. • buildAnAirport(): sets hasAnAirport property to true. • boolean hasAnAirport: returns the value of hasAnAirport property.

Class Name	AnnexationGameContoller
Property	• AnnexationGame annexationGame: holds an instance of annexationGame.
Constructor	• AnnexationGameContoller(int pCount, List<String> pNames, List<Continent> targetContinents): Creates a new according to Annexation game according to its parameters, the continents and the territories of the game.
Method	• AnnexationGame getGame(): returns annexationGame.

Class Name	AnnexationGame
Properties	• AnnexationGame(int playerCount, List<Player> players): sets playerCount and players to its parameters, creates all the territories, continents, players and cards, instantiates a RockPaperScissorsGame and finally assigns 1 to gamePhase
Constructor	• Territory(String name, int ownerId, Color color): sets name, ownerId and color to its parameters.
Methods	• List<Territory> getTerritories: returns all the territories in the game. • void initGraphics(): initializes graphics. • void goToNextTurn(): sets currentPlayerId to the next player’s ID. • void goToNextPhase(): sets gamePhase to the next phase. • boolean isGameOver(): checks whether all the continents have the same owner. If so, returns true. Otherwise, returns false. • void distributeCards(): if the current player deserved a new card, calls cards.drawNext()

	method and passes it to players.get(currentPlayerId).addCard(card) method as a parameter. • boolean sendAllianceRequest(int playerId): sends an alliance request from the current player to the player whose ID is given by calling players.get(playerId).addAllianceRequest(currentPlayerId). • void acceptAllianceRequest(int playerId): forms an alliance between the current player and the player whose ID is given by calling players.get(currentPlayerId).addAlly(playerId) and players.get(playerId).addAlly(currentPlayerId) functions. Of course, for an alliance to be formed, both players must not be eliminated. • boolean breakAlliance(int playerId): breaks an alliance as per the request of one of the allies or when one of the allies is eliminated. • int attackTerritory(Territory source, Territory target, int troopCount): attacks the target territory with troopCount troops in the source territory. The success of the attack is determined according to the Rock, Paper, Scissors game that will be played between the current player and the owner of the target territory. Returns 0, if the attack is impossible (e.g. the source territory does not belong to the current player), Returns 1, if the attack is successful. Otherwise, returns 2. • boolean placeTroops(Territory territory, int troopCount): if the current player tries to place their new troops to one of their own territory, calls territory.setTroops(territory.getTroops() + players.get(currentPlayerId).getTroopPerRoundCount()) and returns true. Otherwise, returns false. • boolean moveTroops(Territory territory1, Territory territory2, int troopCount): if the territory1 and territory2 belongs to the current player and territory1 has more than one troop, moves troopCount troops from territory1 to territory2 and returns true. Otherwise, returns false. • int rockPaperScissors(int player2Id, char p1Choice, char p2Choice): calls rockPaperScissorsGame.play(p1Choice, p2Choice) method and returns its value. • int getCurrentPlayerId(): returns currentPlayerId.

	• int getPhase(): returns gamePhase. • void removePlayer(): checks if there is a player who owns no territory and if such a player exists, removes him/her from the players list. • void useCards(Card card): If the current player uses his/her cards, calls players.get(currentPlayerId).useCards(card.getCardType()) method. • void updateTroopPerRoundCount(): calls players.get(currentPlayerId). updateTroopPerRoundCount() method.

Class Name	Player
Properties	• String name: holds the name of the player. • int playerId: holds the ID of the player. • boolean[] allianceRequests: holds the alliance requests sent to the player. • boolean[] allies: holds the allies of the player. • int troopPerRoundCount: holds the number of troops that the player receives for each turn. • HashMap<Integer, Integer> cardMap: holds the count of each type of card the player owns. • int ownedTerritoryCount: holds the number of territories that player owns. • int ownedContinenentCount: holds the number of continents that player owns. • Continent targetContinent: holds the target continent of the player, which is determined at the beginning of the game. • boolean newCardDeserved: determines whether the player deserved to receive a new card. if the player occupies at least one territory in a turn, this property is set to true. Otherwise, it is set to false. Note that this property is reset in each turn.

	• int activatedCardCount: holds the number of cards that the player used.
Constructor	• Player(int playerId, int ownedTerritoryCount, int targetContinentId): sets playerId, ownedTerritoryCount and targetContinent to its parameters.
Methods	• String getName(): returns the name of the player. • void setName(String name): sets the name of the player. • int getPlayerId(): returns the ID of the player. • void updateTroopPerRoundCount(): updates the number of troops that the player receives in each turn according to ownedTerritoryCount, ownedContinenentCount, activatedCardCount and whether the player captured his/her targetContinent. • void useCards(int cardType): increments activatedCards and updates cardMap according to the card types used. • boolean isAlly(int playerId): returns true if the player whose ID is given is an ally of the player that is the implicit parameter of this function. Otherwise, returns false. • void addCard(int cardType): increments the count of the given cardType in the cardMap. • void addAlly(int playerId): adds an ally to the player by setting allies[playerId] to true. • void removeAlly(int playerId): adds an from the player by setting allies[playerId] to false. • int getTroopPerRoundCount(): returns troopPerRoundCount. • int getOwnedTerritoryCount(): returns ownedTerritoryCount.

	• void setOwnedTerritoryCount(int ownedTerritoryCount): sets ownedTerritoryCount to its parameter. • Continent getTargetContinent(): returns targetContinent. • boolean isPlayerDefeated(): returns true if ownedTerritory count is zero. Otherwise, returns false. • void addAllianceRequest(int playerId): sends an alliance request to the player by setting allianceRequests[playerId] to true. • List<Integer> getAllianceRequests(): returns alliance requests as a list. • boolean getNewCardDeserved(): returns newCardDeserved. • void setNewCardDeserved(boolean newCardDeserved): sets newCardDeserved to its parameter. • int getOwnedContinenentCount(): returns ownedContinentCount. • void setOwnedContinenentCount(int ownedContinenentCount): sets ownedContinentCount to its parameter.

Class Name	Cards
Property	• List<Card> cards: holds a list of cards.
Constructor	• Cards(List<Card> cards): sets cards to its parameters.
Methods	• void shuffle(): shuffles the cards list. • Card drawNext(): removes and returns the first element of the cards list.

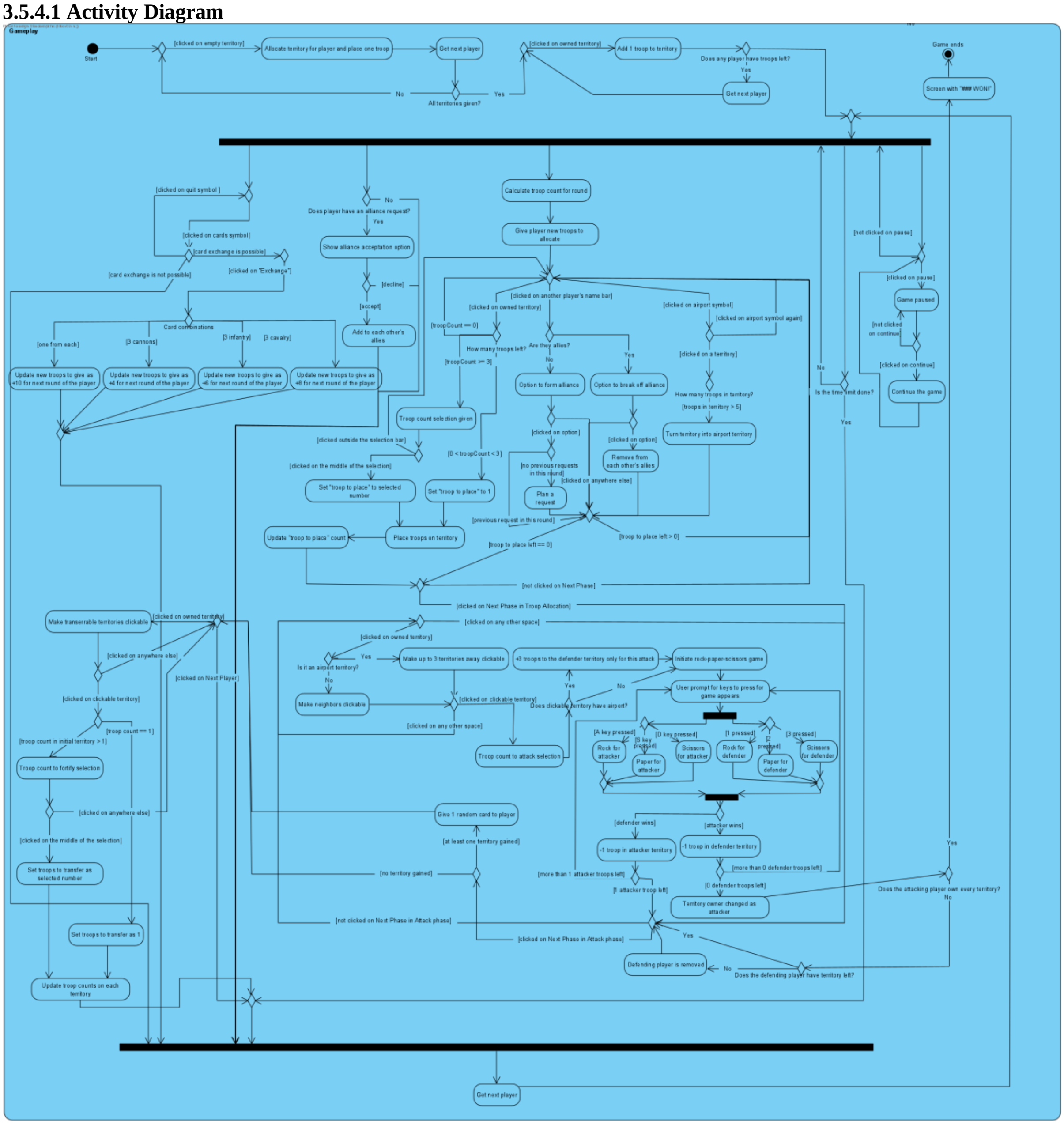
Class Name	Card
Property	• int cardType: holds the type of the card.
Constructor	• Card(int cardType): sets cardType to its parameters.
Method	• int getCardType(): returns cardType.

Class Name	JokerCard
Constructor	• JokerCard(): assigns -1 to cardType.

Class Name	RockPaperScissorsGame
Method	• int play(char p1Choice, char p2Choice): simulates a Rock, Paper, Scissors game.

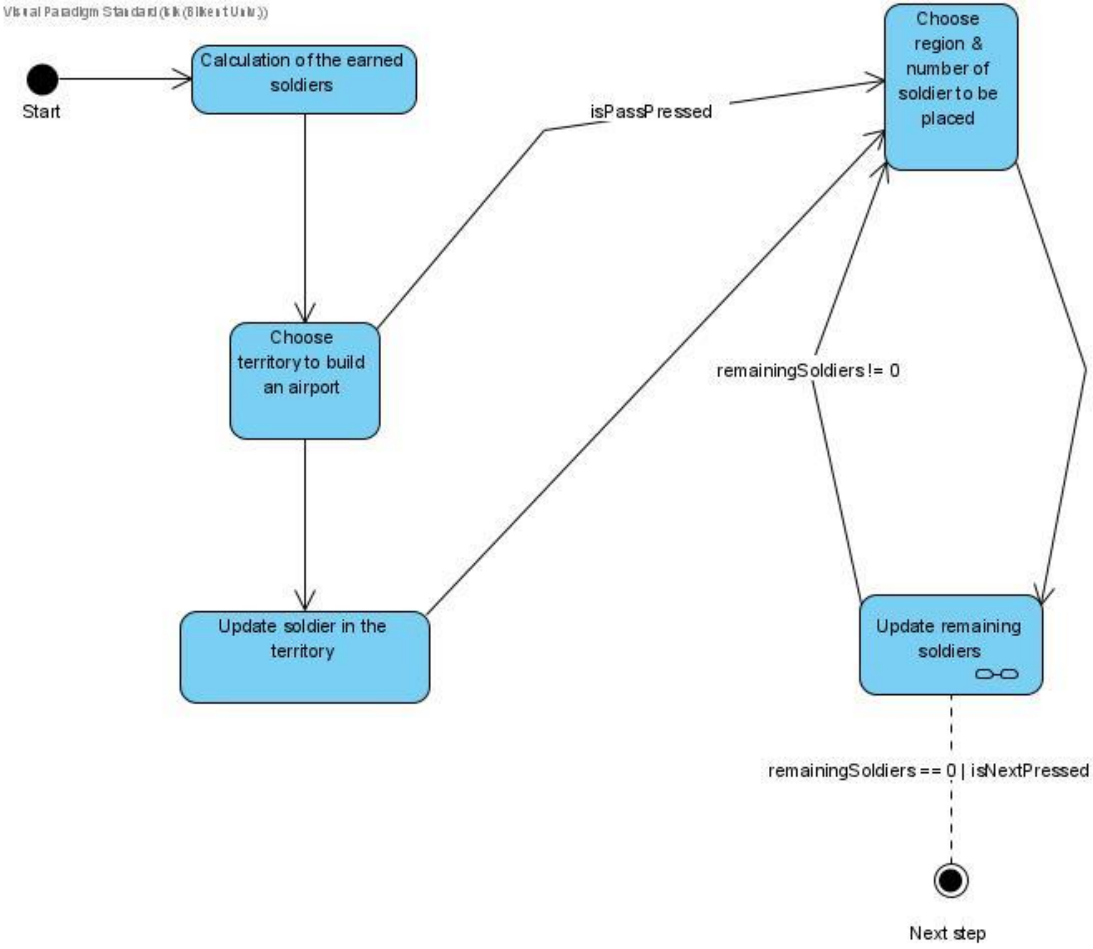
3.5.4 Dynamic Models

3.5.4.1 Activity Diagram


Gameplay
Start
[clicked on empty territory]
Allocate territory for player and place one troop
Get next player
No
Yes
All territories given?
[clicked on owned territory]
Add 1 troop to territory
Does any player have troops left?
Yes
Get next player
Game ends
Screen with "### WON!"
[clicked on quit symbol]
[clicked on cards symbol]
[card exchange is possible]
[card exchange is not possible]
[clicked on "Exchange"]
Card combinations
[one from each]
[3 cannons]
[3 infantry]
[3 cavalry]
Update new troops to give as +10 for next round of the player
Update new troops to give as +4 for next round of the player
Update new troops to give as +6 for next round of the player
Update new troops to give as +8 for next round of the player
No
Does player have an alliance request?
Yes
Show alliance acceptation option
[decline]
[accept]
Add to each other's allies
Calculate troop count for round
Give player new troops to allocate
[clicked on another player's name bar]
[clicked on owned territory]
[troopCount == 0]
How many troops left?
[troopCount >= 3]
Troop count selection given
[clicked outside the selection bar]
[clicked on the middle of the selection]
[0 < troopCount < 3]
Set "troop to place" to selected number
Set "troop to place" to 1
Place troops on territory
Update "troop to place" count
Are they allies?
No
Yes
Option to form alliance
Option to break off alliance
[clicked on option]
[clicked on option]
[no previous requests in this round]
Plan a request
Remove from each other's allies
[clicked on anywhere else]
[previous request in this round]
[troop to place left == 0]
[troop to place left > 0]
[clicked on airport symbol]
[clicked on airport symbol again]
[clicked on a territory]
How many troops in territory?
[troops in territory > 5]
Turn territory into airport territory
No
Is the time limit done?
Yes
[not clicked on pause]
[clicked on pause]
Game paused
[not clicked on continue]
[clicked on continue]
Continue the game
[not clicked on Next Phase]
[clicked on Next Phase in Troop Allocation]
[clicked on any other space]
[clicked on owned territory]
Is it an airport territory?
Yes
No
Make up to 3 territories away clickable
Make neighbors clickable
[clicked on clickable territory]
[clicked on any other space]
Troop count to attack selection
Does clickable territory have airport?
Yes
No
+3 troops to the defender territory only for this attack
Initiate rock-paper-scissors game
User prompt for keys to press for game appears
[A key pressed]
[S key pressed]
[D key pressed]
[1 pressed]
[2 pressed]
[3 pressed]
Rock for attacker
Paper for attacker
Scissors for attacker
Rock for defender
Paper for defender
Scissors for defender
[defender wins]
[attacker wins]
-1 troop in attacker territory
-1 troop in defender territory
[more than 1 attacker troops left]
[1 attacker troop left]
[more than 0 defender troops left]
[0 defender troops left]
Territory owner changed as attacker
Does the attacking player own every territory?
Yes
No
Does the defending player have territory left?
No
Yes
Defending player is removed
Give 1 random card to player
[at least one territory gained]
[no territory gained]
[not clicked on Next Phase in Attack phase]
[clicked on Next Phase in Attack phase]
Make transerrable territories clickable
[clicked on owned territory]
[clicked on anywhere else]
[clicked on Next Player]
[clicked on clickable territory]
[troop count == 1]
[troop count in initial territory > 1]
Troop count to fortify selection
[clicked on anywhere else]
[clicked on the middle of the selection]
Set troops to transfer as selected number
Set troops to transfer as 1
Update troop counts on each territory
Get next player

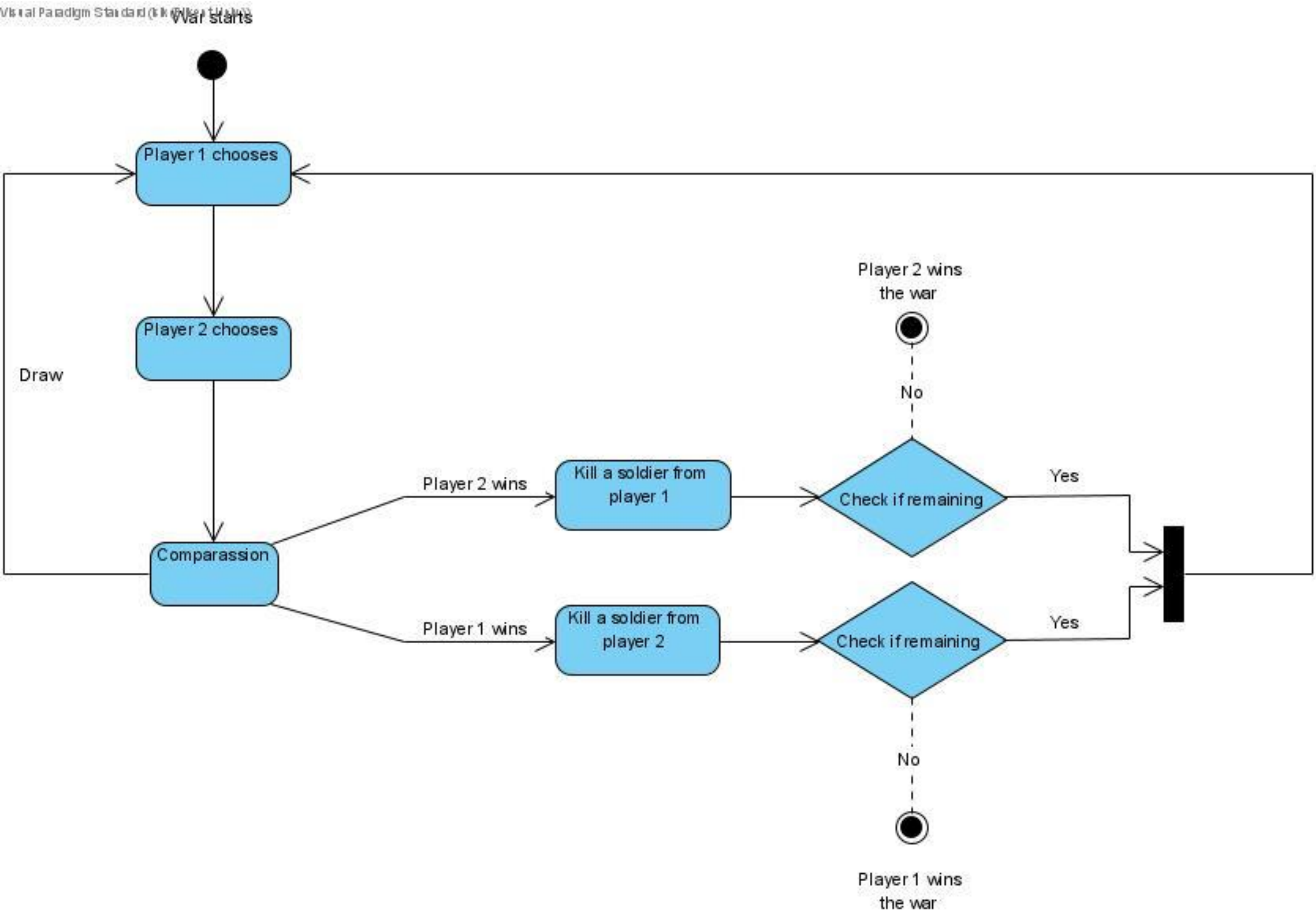
The Gameplay of Annexation will first start with actions for territory allocation and the placement of their correspondent troops. Each action is done with clicks and the inputs are given to the system as such. Each user can choose one territory for themselves and after the allocation ends with all the territories being given, the player who is on the next turn then places one troop to one of their territories each turn. This goes on until there is no extra troop to place. After this phase is done, the game continues with multiple concurrent actions: check for clicks on the card symbol for the card exchange system, the time limit check, and the game with multiple phases itself. The time limit will go on for the duration of each turn and a turn ending will either be by the player clicking on “Next Player” at the last phase, or the time limit being reached, or by the player winning and ending the game. The first phase of the game, Troop Allocation which is referred to as “Allocation” in our interfaces, will start by new troop calculation, and although the card system will occur for the entire duration of each turn, the new troops awarded will only be given in the Troop Allocation phase – the player exchanging cards in the Attack phase will result in them getting awarded for it in their next turn, for example. Here, another click option given is for the “Airport Territory” system, in which the player will click on the symbol for this system, and then click on one of their territories. A click on another space in the window will not make this process terminate, and they will terminate this process by clicking on airport symbol again. One other important thing is that going to the next phase before allocating troops will not be allowed, although ending the phase without doing anything is allowed in other phases. Alliance option is also only available in this phase in which the user will click on the bar for which the user name will be written on and then click on the option seen, in which the player can either form alliance (allowed once for each round) or break it. Click detected on “Next Phase” will make the system go to the next phase, which has been called “Attack Phase” throughout this report. In this phase, the player can either terminate the whole phase by clicking on “Next Phase” upon entering or click on a territory of theirs from which they have to click on a neighboring territory or a territory that is up to 3 territories away in the case of selected territory being an Airport territory. The player clicking on any other space in the window will result in the process returning to the beginning of the phase, in which the user can click on any other territory. If the player clicked on both of these territories without deciding to go back to the beginning, a troop count selection will appear, and the user selecting the number of troops of theirs to play will take the players to the rock-paper-scissors game. The system then will prompt them to keyboard positions to press for certain actions of rock-paper-scissors game, and the results of the game will be compared as there are three possible outcomes for one game: no one wins, in which case both players chose the same rock-paper-scissors action to play and the game will be repeated until there is a different result, and either the defender or the attacker wins the game. The defender winning one game will result in 1 troop to be redacted from the attacker territory, and vice versa for the case of attacker winning. The rock-paper-scissors game rounds will end either when the troop count the attacker has chosen to attack with has reached 1, or when the defender has no troops to defend. For the former case, the game will proceed to the Attack phase as usual while the latter case will result in the owner of the territory changing. After each successful attack, the system will first check whether every territory belongs to the player that currently won, and the “### won!” will appear in the game if so. Another checking will happen to make sure that the attacked player has any territory left, and if not, the player will be removed. The phase will go on as long as either the user selects to go to the next phase or the time limit is reached. The system will also check, after the attack phase, if the user has gained any territory during the attack phase and will give the player a random troop card. The fortify phase will occur in the same way as the game with the player choosing one territory to displace troops and a second other territory out of territories that have land connection to the first one, giving the troop count to transfer in the process as an input to the game. In our game, one addition to this process is the option to leave no soldiers in the territory, but this is an addition that does not involve a separate action and was not added to the activity diagram as such. This phase will reach to an end either when the player clicks on “Next Player” or when there is no time for this round. Thus, the round of one player will end and the system will get the next player for which the same phases will occur for as long as one player has ownership of all of the territories.

3.5.4.2 State Machine Diagrams


Visual Paradigm Standard(lk (Bilkent Univ.))
Start
Calculation of the earned soldiers
isPassPressed
Choose region & number of soldier to be placed
Choose territory to build an airport
remainingSoldiers != 0
Update soldier in the territory
Update remaining soldiers
remainingSoldiers == 0 | isNextPressed
Next step

3.5.4.3 Troop Allocation

In this diagram, the soldier allocation phase is presented. The beginning of each round is shown. Firsty the troop count is calculated. Firstly, the game asks layer whether s/he wants to build an airport, if the player selects a territory, the number of soldiers is updated and the airport is built, otherwise the current state becomes soldier placement state, otherwise . In this state, the player selects a territory and soldier count. After specifying those, the remaining soldier count is updated in update remaining soldiers state. If remainingSoldiers is equal to zero, the game continues.

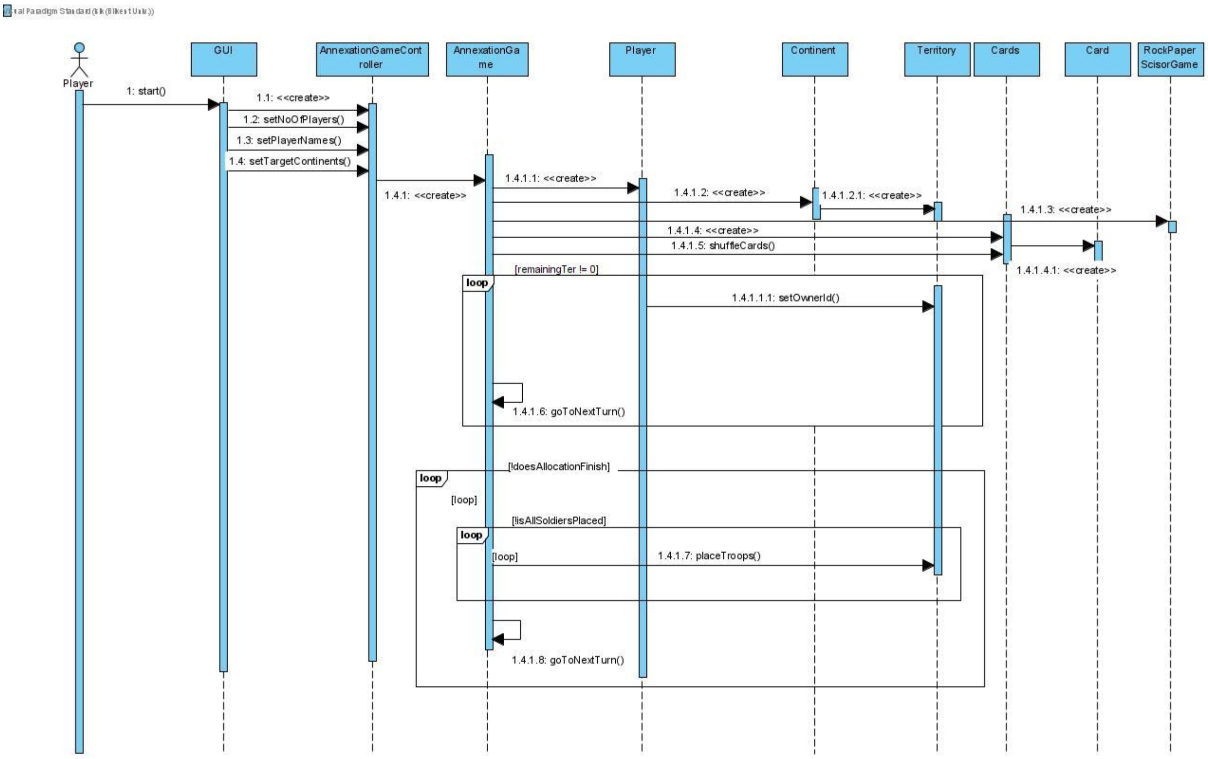


3.5.4.4 Rock-Paper-Scissors

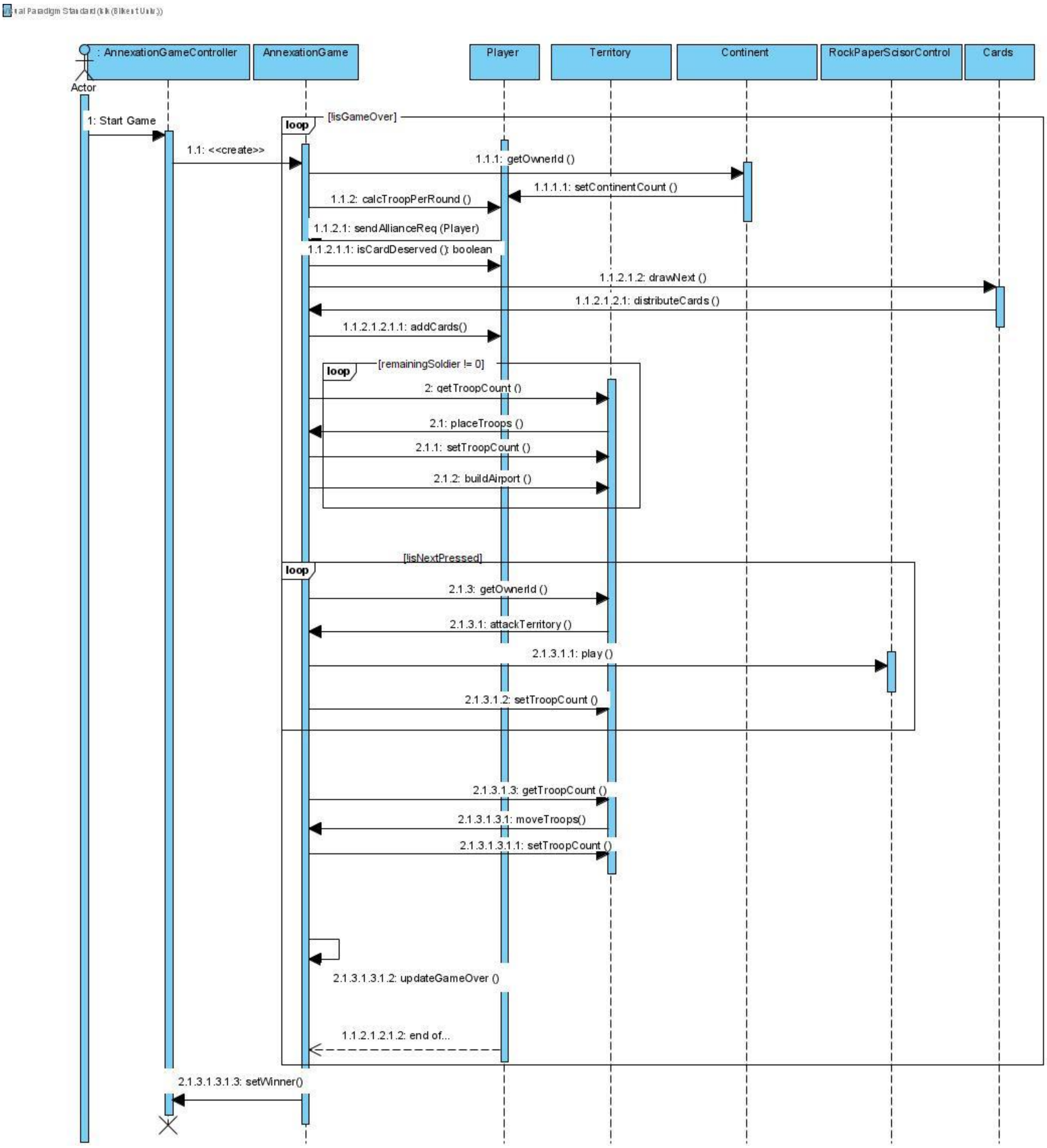
A rock paper scissors game is initiated whenever a player attacks another player. Firstly, player 1 chooses an action, then the player 2 also chooses an action. Then their actions are compared. If they have chosen the same actions, there occurs a draw and they choose actions again. If they have chosen different actions, the winner is announced. Followingly, the number of troops of the loser is checked. If the player has more troops than zero, another round is player, if not the game ends.

3.5.5 Sequence Diagram

3.5.5.1 Initializing The Game

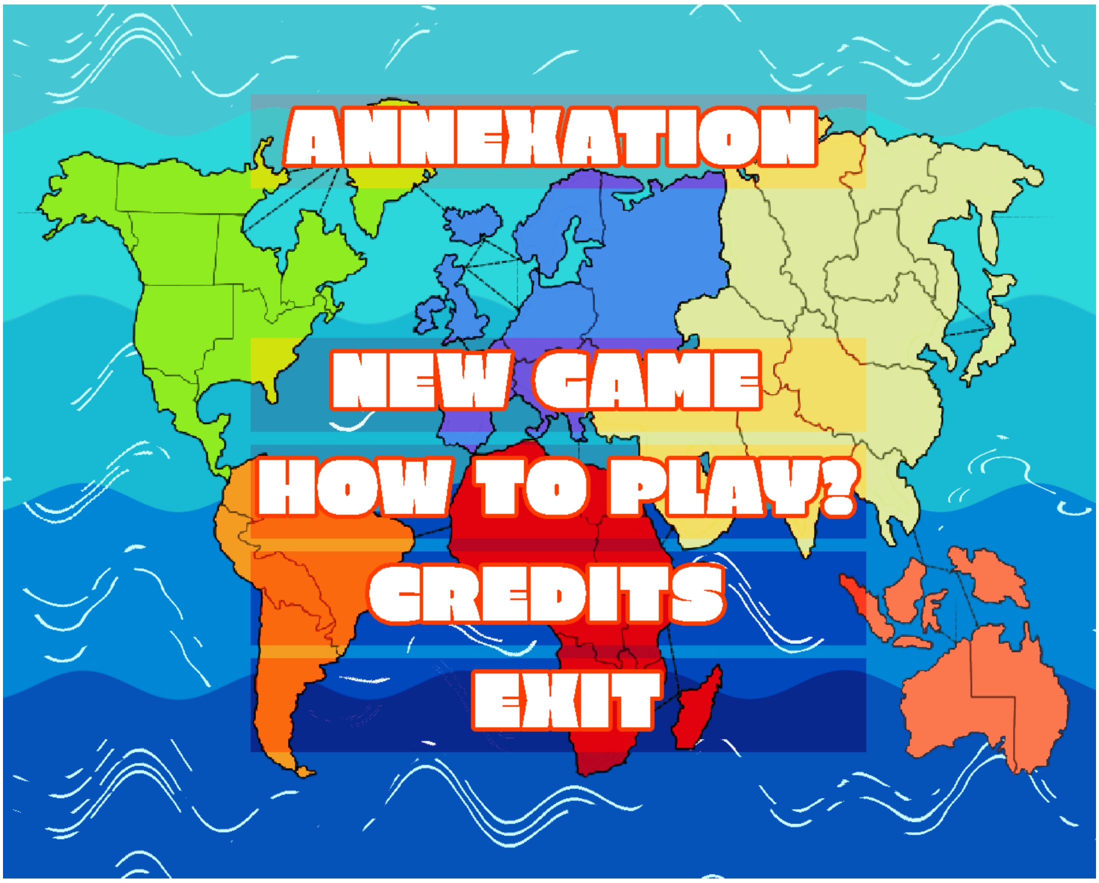


Initialization starts when the user wants to create a game. Firstly, the AnnexationGameController object will be created. The number of the players, their names and the target continent should be specified so that with that information AnnexationGameController calls create function to create an AnnexationGame object. AnnexationGame object also holds the number of players and their names. AnnexationGame object uses those attributes in creation of Player objects in its constructor and initially, all the players will have the same number of soldiers. AnnexationGame object also creates Territories, Continents, Card, Cards and RockPaperScissorGame. After creating all the required objects, cards are shuffled with shuffleCards method. AnnexationGame starts with territory allocation, players place soldiers one by one to different territories with the placeTroops method and this continues until all the territories are allocated. goToNextTurn method is used to increase the id current player. After all the territories are allocated, the players will place their soldiers until remainingSoldier is zero.3.5.5.2 Rounds

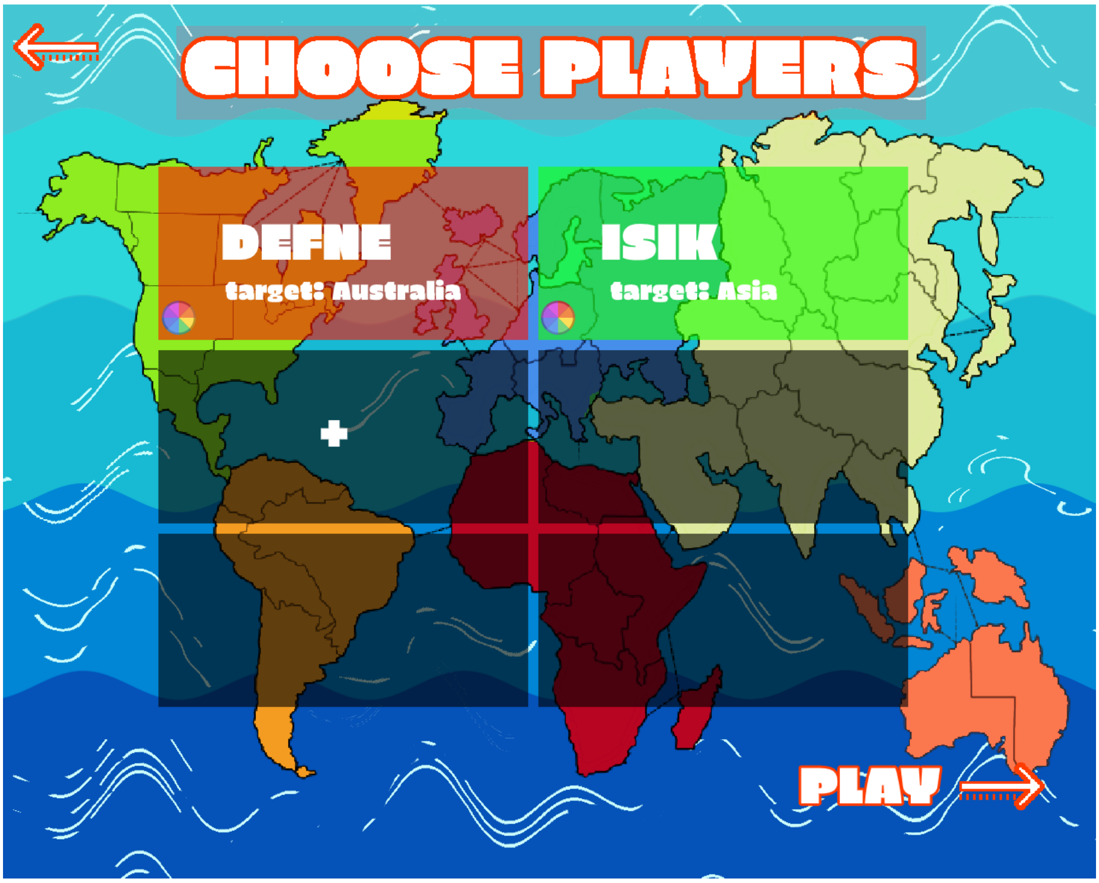

Actor
: AnnexationGameController
AnnexationGame
Player
Territory
Continent
RockPaperScisorControl
Cards
1: Start Game
1.1: <<create>>
loop
[!isGameOver]
1.1.1: getOwnerId ()
1.1.1.1: setContinentCount ()
1.1.2: calcTroopPerRound ()
1.1.2.1: sendAllianceReq (Player)
1.1.2.1.1: isCardDeserved (): boolean
1.1.2.1.2: drawNext ()
1.1.2.1.2.1: distributeCards ()
1.1.2.1.2.1.1: addCards()
loop
[remainingSoldier != 0]
2: getTroopCount ()
2.1: placeTroops ()
2.1.1: setTroopCount ()
2.1.2: buildAirport ()
loop
[!isNextPressed]
2.1.3: getOwnerId ()
2.1.3.1: attackTerritory ()
2.1.3.1.1: play ()
2.1.3.1.2: setTroopCount ()
2.1.3.1.3: getTroopCount ()
2.1.3.1.3.1: moveTroops()
2.1.3.1.3.1.1: setTroopCount ()
2.1.3.1.3.1.2: updateGameOver ()
1.1.2.1.2.1.2: end of...
2.1.3.1.3.1.3: setWinner()

When the user starts the game, AnnexationGameController object creates the AnnexationGame object and after finishing the initialization, the game starts. In a single round, firstly, it is checked whether the player has all the territories in the same continents or not with the getOwnerId() and setContinentCount method and 4 soldiers are given as a bonus for each continent the player has. Moreover, in calcTroopPerRound, the troop count is calculated by dividing territory number into three and it is also checked whether the player has the target continent, if so, the player will be given 4 more soldiers. After the calculation, the player will be able to send a request to become allies with sendAllianceReq method. After that, it is decided whether the player will get a card with isCardDeserved method, according to the return value of this method, the user will either get one card or none with the drawNext, the distributeCards and addCards methods. After, the soldier allocation phase is started. In order to place the gained soldiers getTroopCount, placeTroops and setTroopCount methods are used. Then, there is a method to build an airport on the selected territory which is buildAirport. This phase is repeated until there is no remaining soldier. Then, the attack phase is started. Firstly, the player selects two territories along with the soldier count to attack with attackTerritory. After the attackTerritory method is called, the play method is also called by RockPaperScissorController. For every lost round, a soldier of the losing side will die and the soldier counts in the territory will be updated by updateSoldierCount method. When the player clicks on the next button, the fortify phase will start. In this phase, the player chooses two territories and the number of soldiers s/he wants to move and counts will be updated with the setTroopCounts method. Moreover, the isGameOver value is updated with the updateGameOver method and it is checked whether the game is over or not. If it is not, the currentPlayerId is updated and the game continues with the next player.

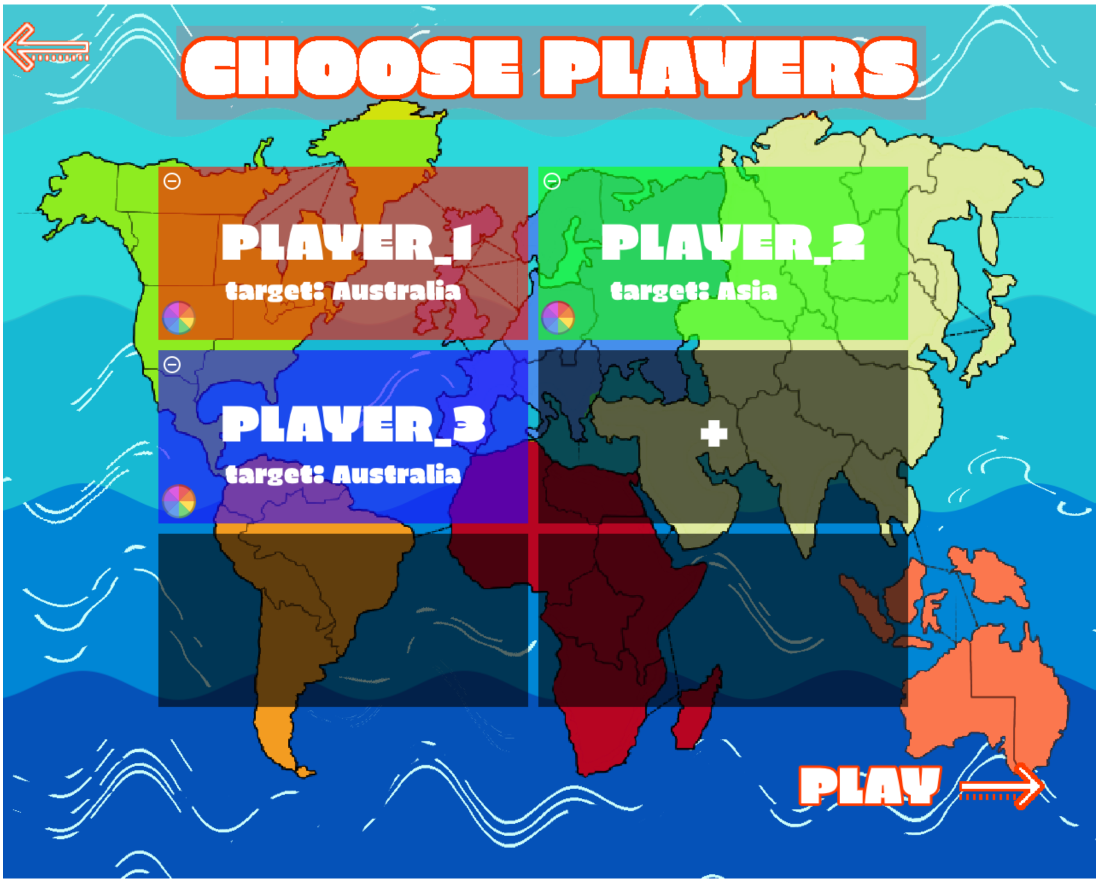
3.6 User Interface



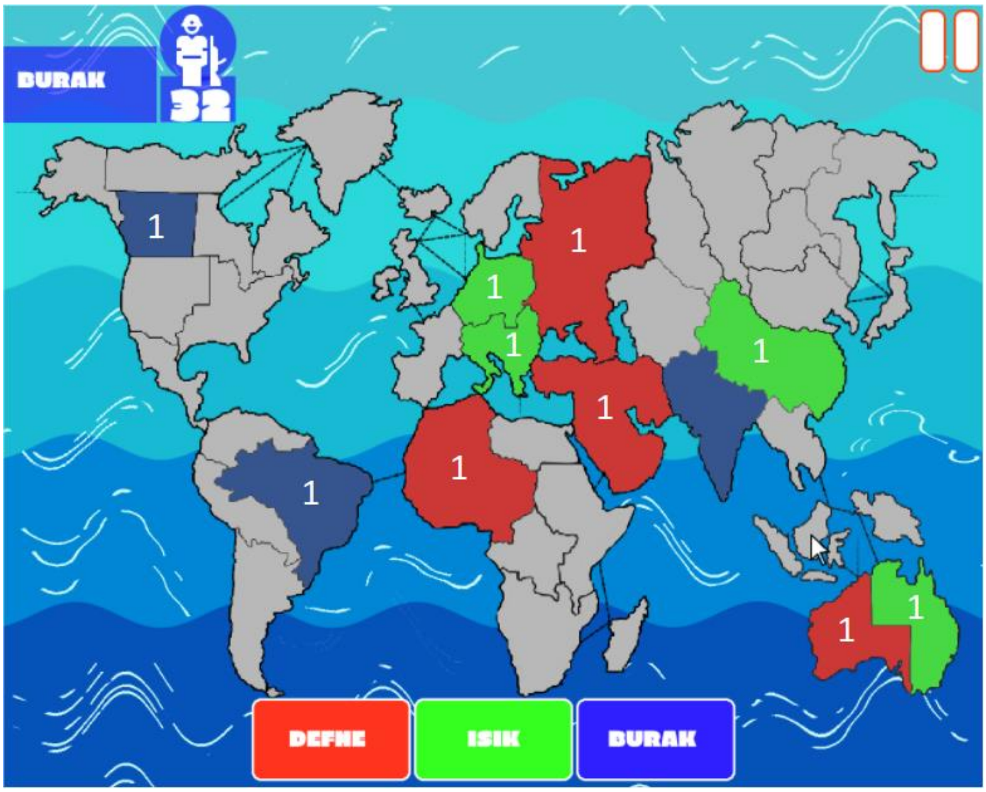
The main menu of Annexation includes a “New Game” button alongside a “How to Play?” button, a “Credits” button, all of which will be explained in the next sections, and an “Exit” button which will make the player exit the game interface safely.

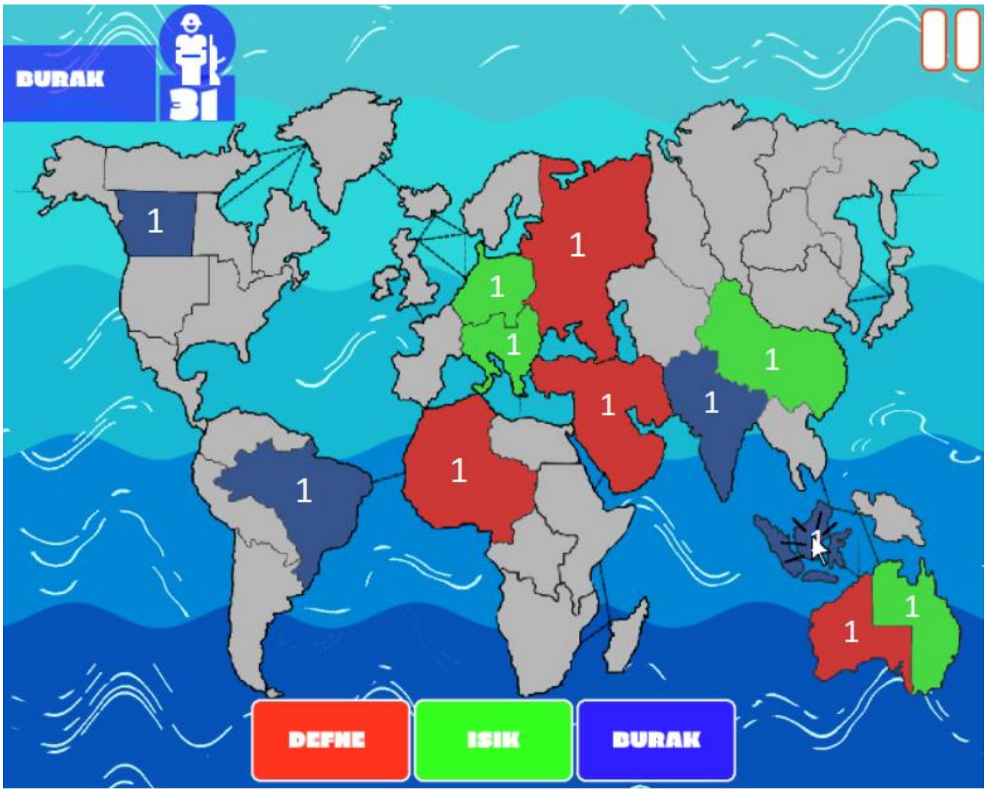


The “Choose Players” screen appears once the “New Game” button is pressed, and the user still has the option to go back with the back button above the screen, as shown with a left arrow sign, and the user can add players by pressing the + sign in the reserved space for other players. Changing player name will be done by double clicking on each of the user names, and target continents will appear in a dropdown menu style. The players can change their colors by clicking the circled rainbow sign in the left bottom side of each space reserved for a player.

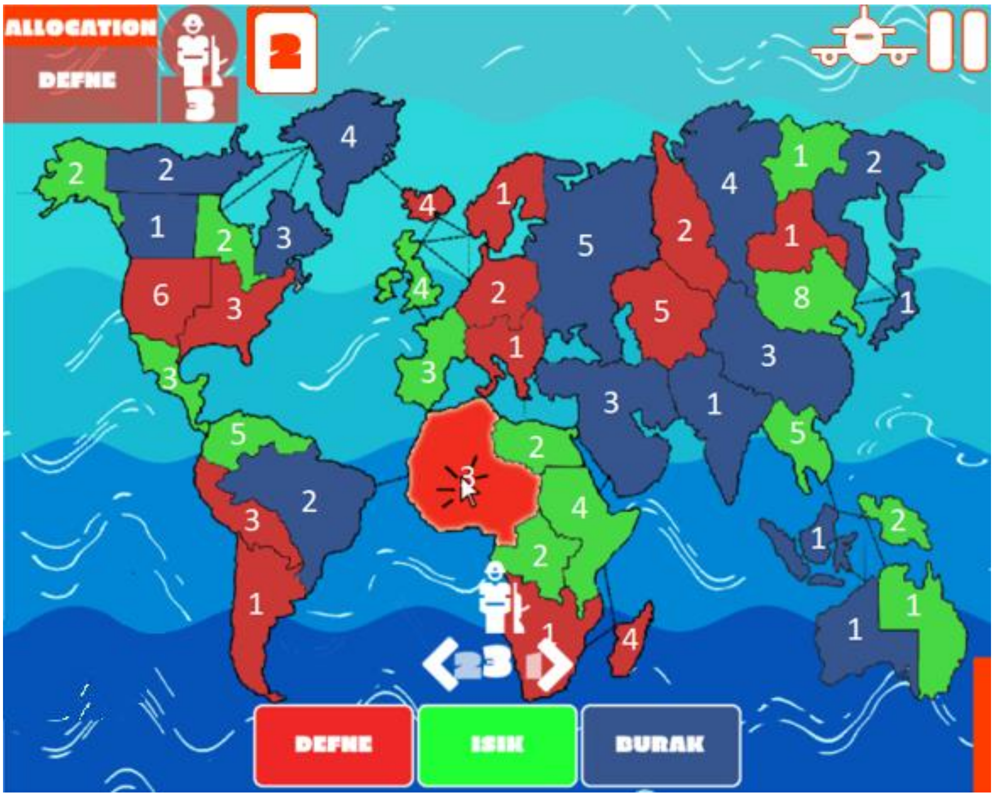


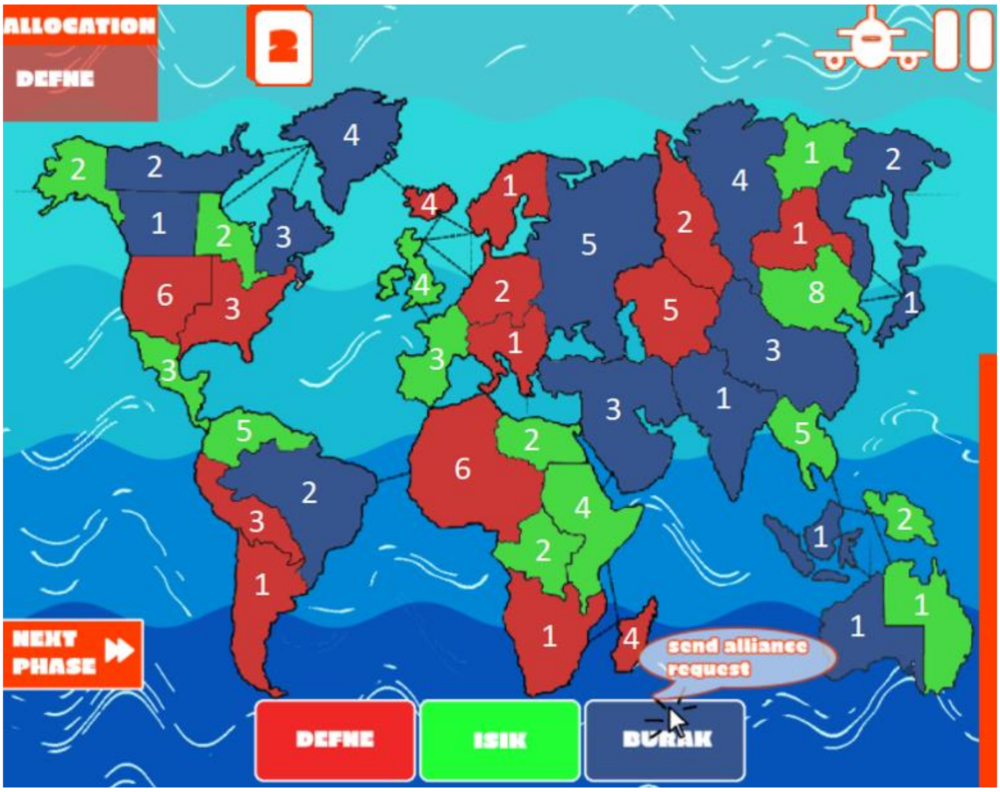
As an extension to the previous screen, once the player count exceeds 2 a new option to delete one player will appear which is signified by the minus (-) sign encircled in the left above side in the reserved space for each user.



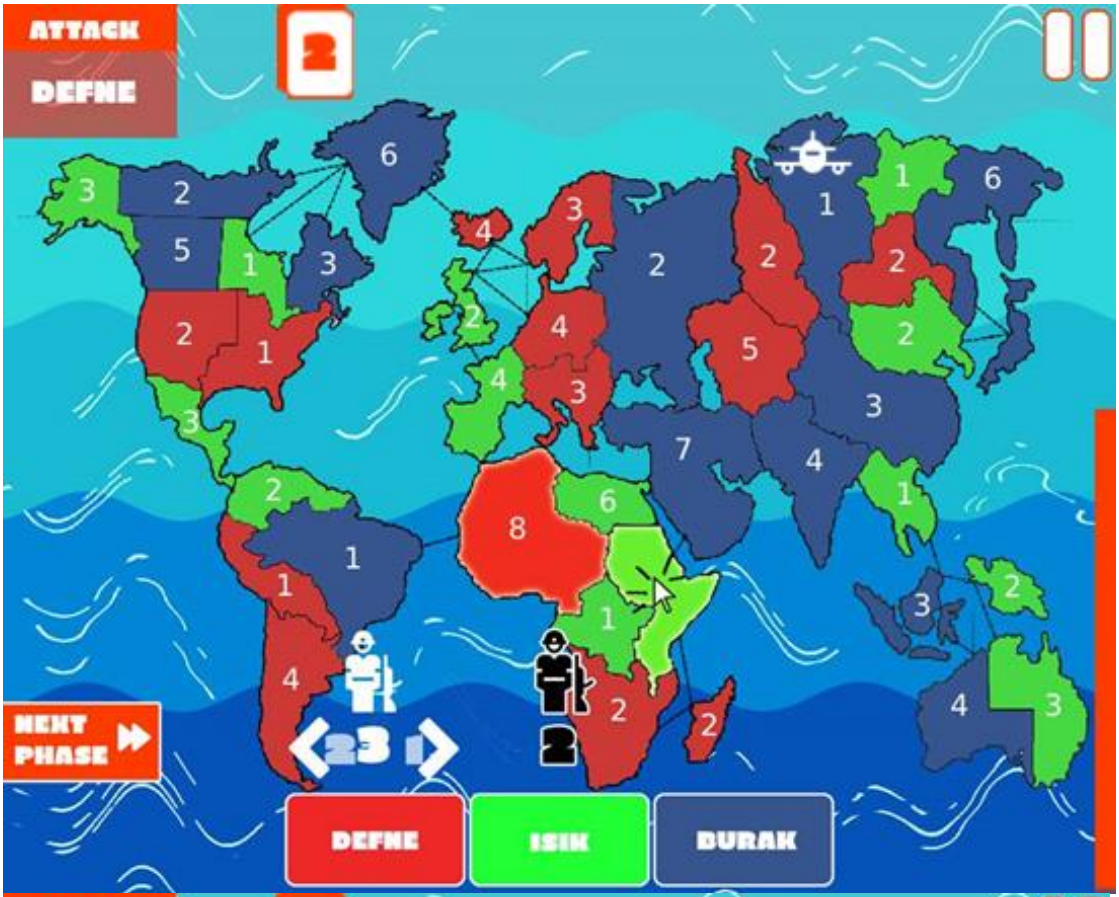


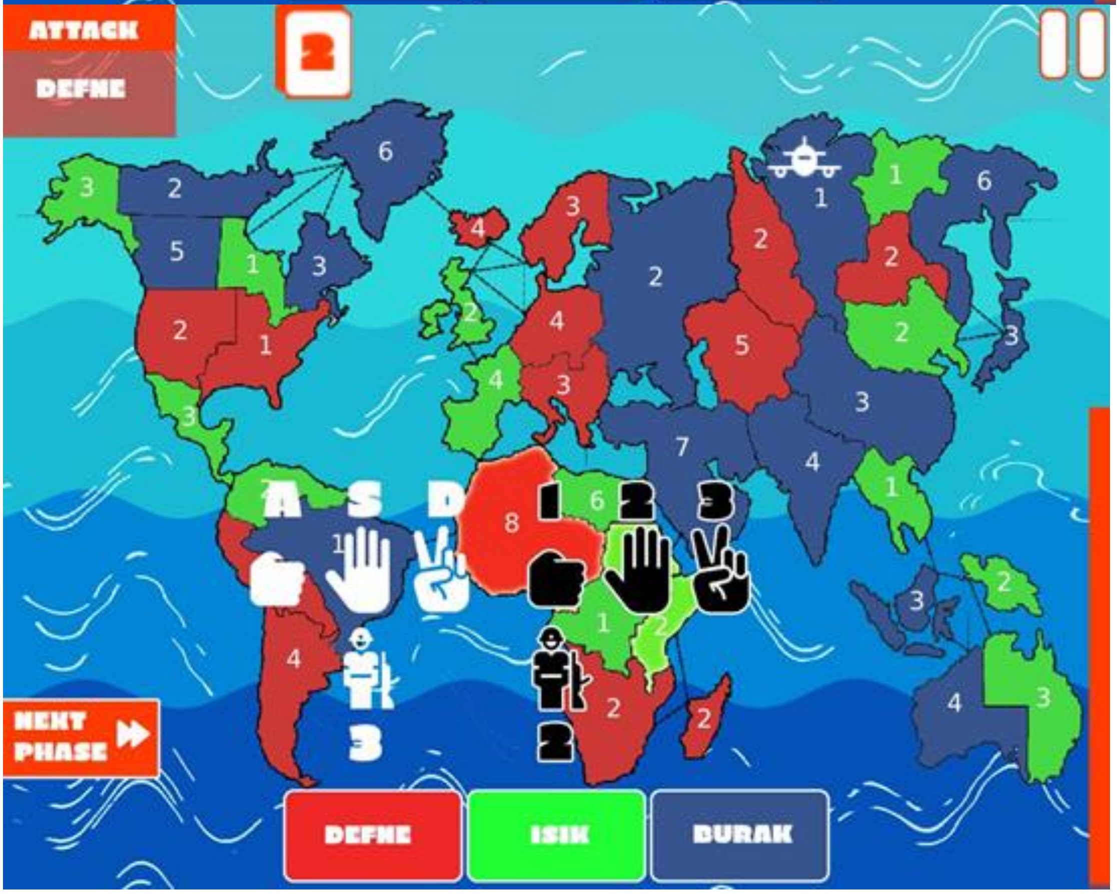
The above two screens show how the territory allocation phase will occur once the Play button is pressed on the Choose Players screen. The player names with their chosen colors will be shown on the bottom of the screen, and the player currently playing will be shown on the left above side of the screen next to a troop icon if the current phase requires a troop count to be held. The user will click on the territory that they want to occupy during this phase and their troops will lessen by one, and this interface will be shown to all players in this phase.

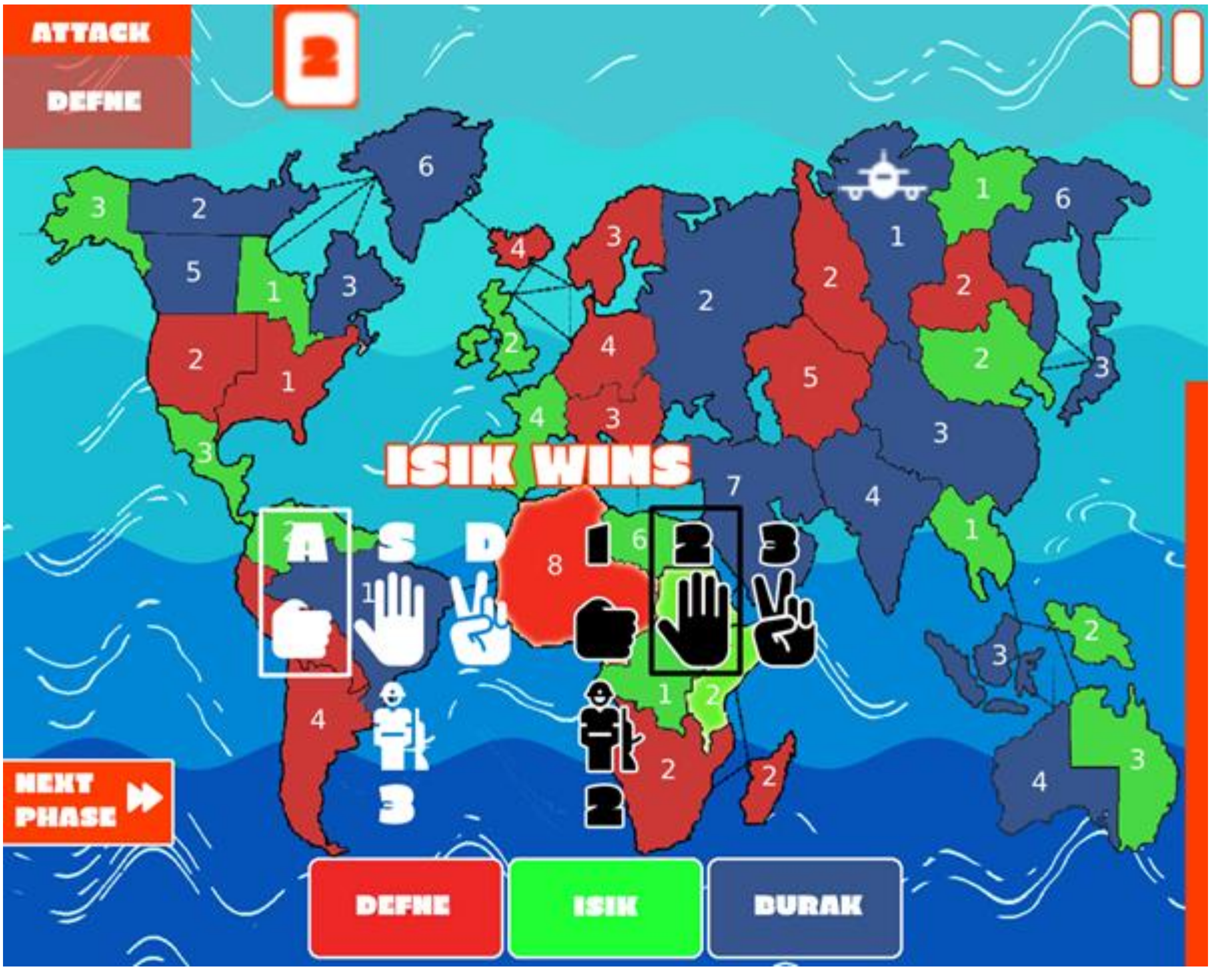




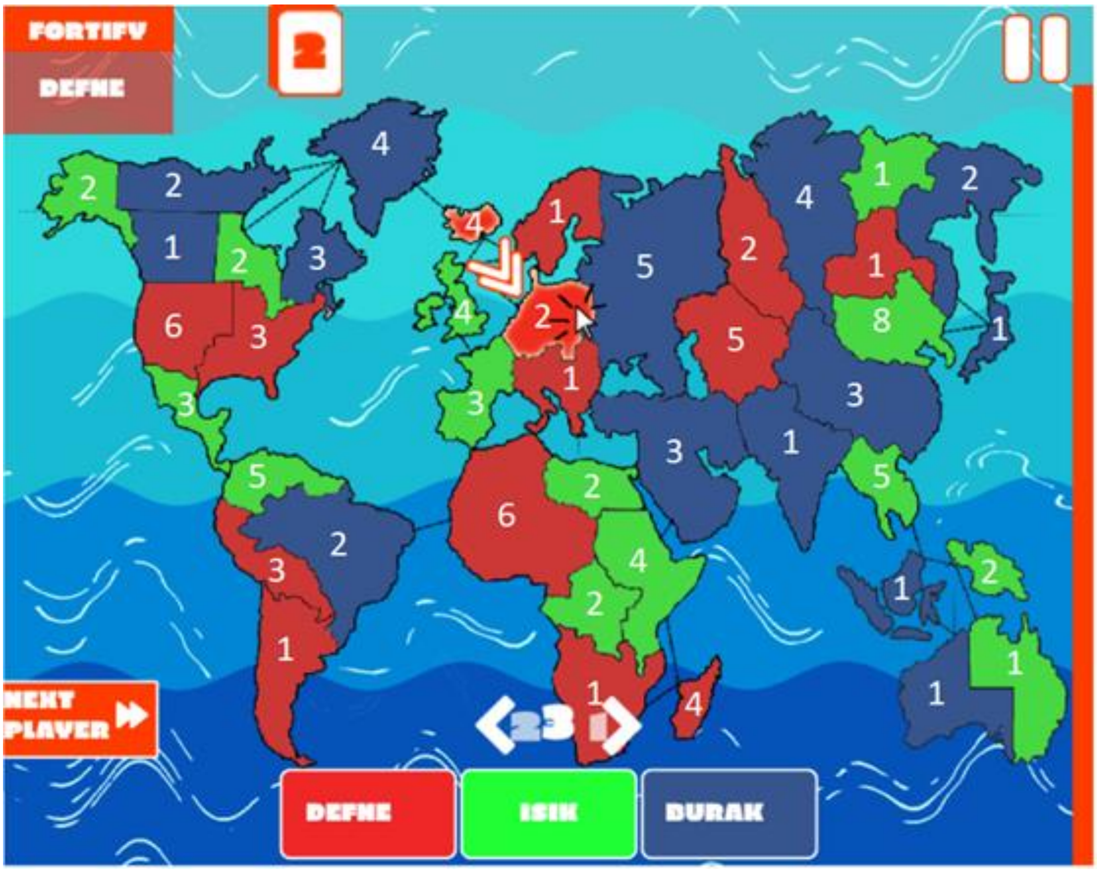
This screen shows how the troop allocation will be done for the Troop Allocation phase, as signified by the phase title in the upper left side of the screen and the player that plays currently. The time limit is shown on the right side of the screen, the players with their chosen colors are shown on the bottom, and the Next Phase button is on the left side of the screen. There is also a Pause button on the upper right side of the screen next to the airplane symbol which signifies the airport territory and will be clicked on when the user wants to turn a territory into an airport territory – from which the user will then have to click on a territory that has more than 5 (subject to change) troops placed. The orange card symbol with a number over, right next to the troop symbol in the first picture, is for cards that the user can exchange for troops and this symbol will be shown for all turns of each user. Troop counts with a troop icon will be shown in the upper left part of the screen as well for the player to keep track of the count of their newly acquired troops, and this signifier will be gone once the player has no troops to place. The user will click on a territory for which whether it has been clicked or not will be shown with a signifier (we have not decided yet, but for this interface it is shown as a glow, as seen above). Once it has been clicked, a troop icon with the number of troops to be placed will be shown in the middle below part of the map screen. The user will then select the number of their troops to place for which the user can click on the “<” and “>” buttons, and can choose the count of troops to place by clicking on the count on the screen (in this case, clicking on 3 in the middle). The second image shows the user interface for the alliance system, in which the user has to click on the user’s signifier below the screen and then click on the “Send alliance request” dialogue button above the other user’s name. The chosen player will be able to respond to this once it is their turn as its interface design will be shown later.


ATTACK
DEFNE
NEXT PHASE
DEFNE
ISIK
BURAK

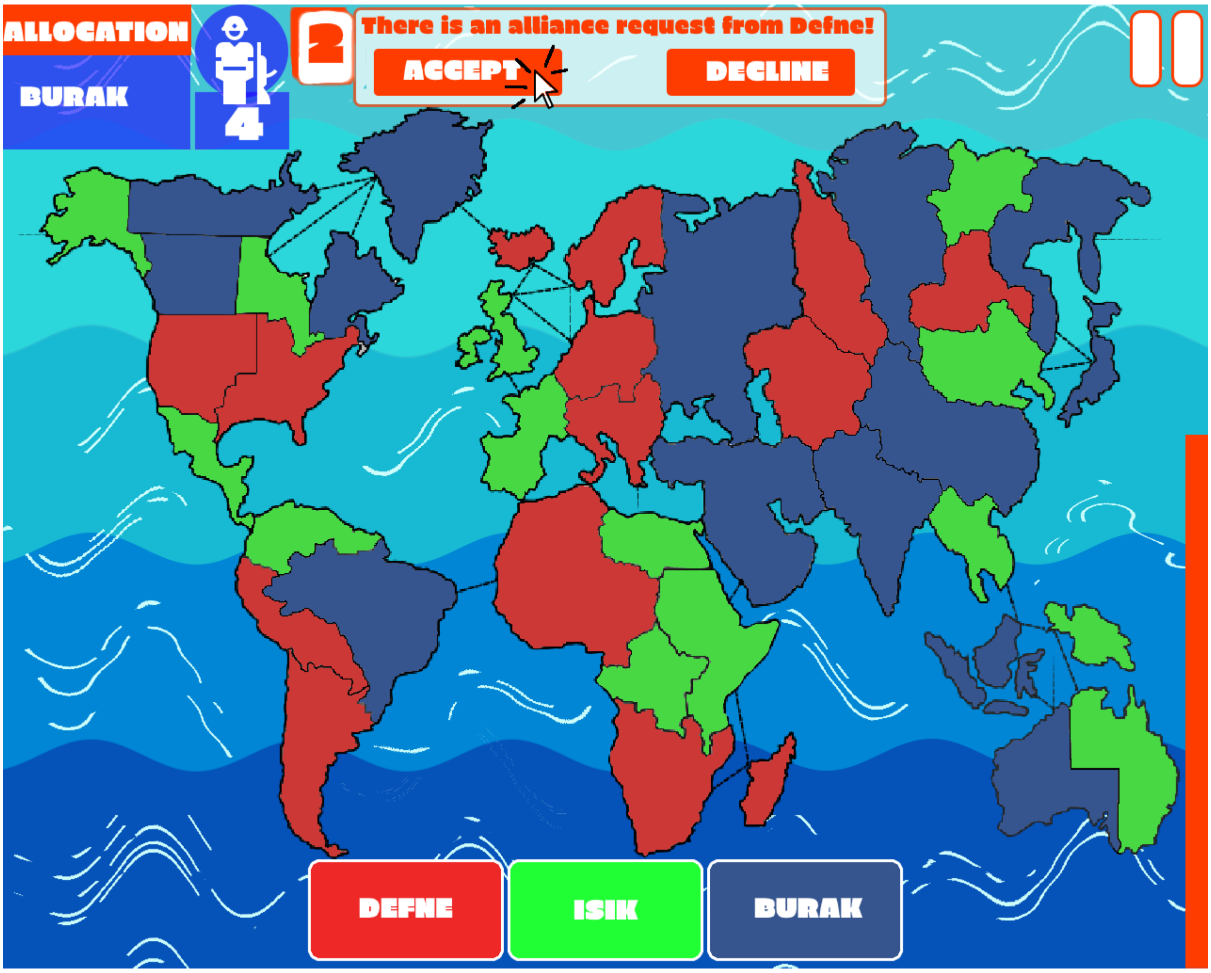

ATTACK
DEFNE
A
S
D
1
2
3
NEXT PHASE
DEFNE
ISIK
BURAK



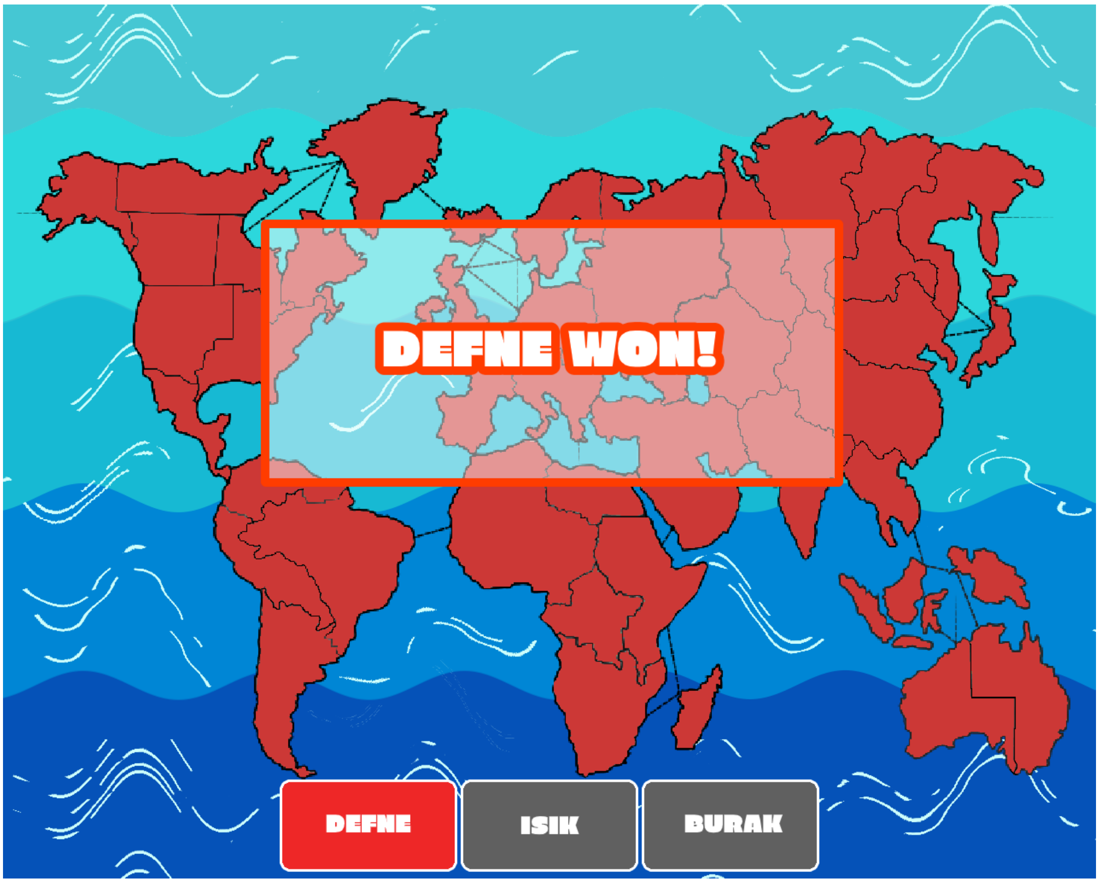
The 3 images above show how the interface for the attack mechanism will be shown in the interface. The first territory clicked will be the attacker territory and the second territory clicked with the left mouse button will be the attacked territory - the selected territories will be shown with a signifier, which is glowing as chosen for this interface. Again, an option for the user to select the number of troops that they want to attack with will be shown with a troop icon and the troop count with left and right arrow signs, the user will select their troop count by clicking and once the troop count has been set, the rock-paper-scissors icons with their corresponding keyboard hits will show up in the screen. Once the computer has both inputs, the inputs selected will be shown with a border over them and the winner of the round will be shown on the screen. A signifier for airport territories will also be placed over the territories which are shown as having airports – the little airplane symbols in these interfaces were shown as a substitute for we as a group have not yet decided on how to signify this and as such, this symbol was not used for the rest of this report for aesthetic concerns of ours but was put in these three interface models to show the system in a more clear way.



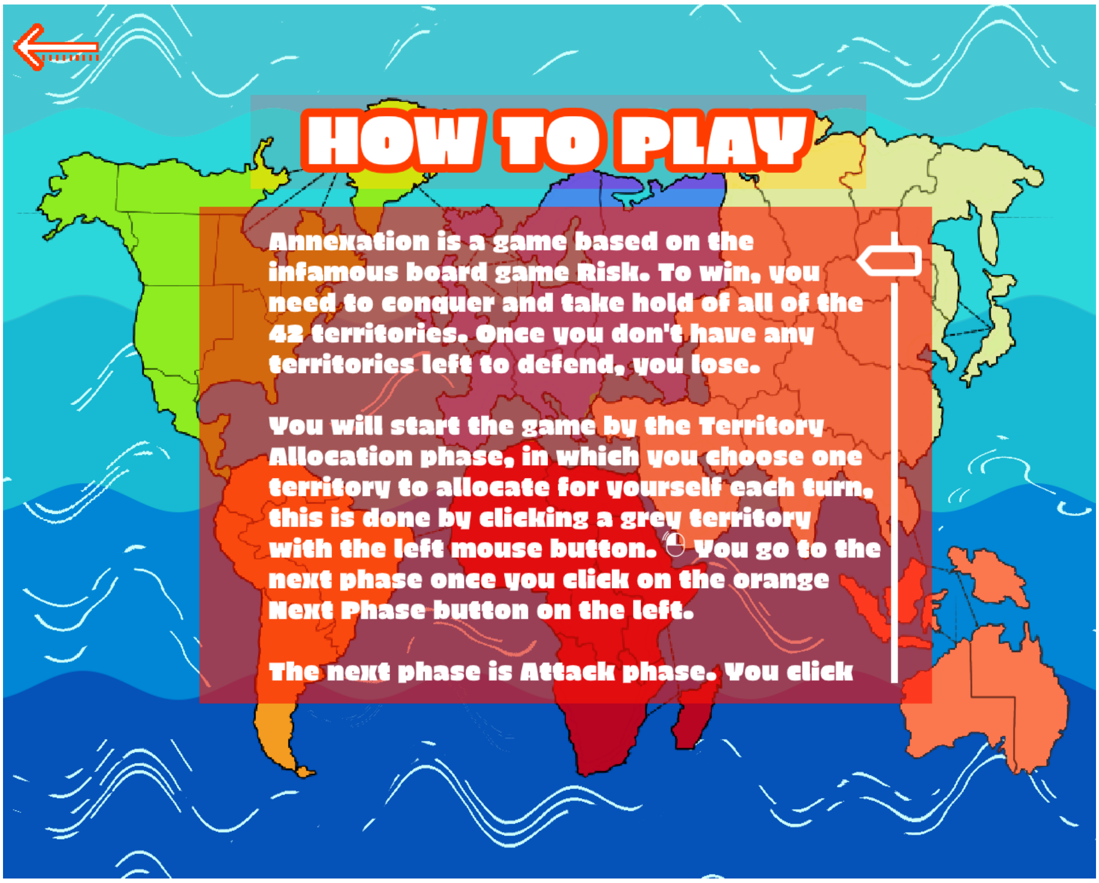
Once the player clicks on Next Phase in the Attack phase, it will switch to the Fortify phase in which the Next Phase button changes to “Next Player.” The user clicks on two territories of theirs, shown with signifiers, and then select the number of their troops as shown in the below middle part of the map screen. The transaction between territories will be shown by a “>>” sign between them.



The above interface design shows how the alliance acceptance option will be shown in the screen for the player on the receiving end of the request. As one can see, the Next Phase option is not visible yet and it will be seen once all the troops have been allocated to territories.



Once a player is out of the game, their name in the below part of the screen will be grayed out. When a player wins, the winner’s name will be shown as seen in the above design, and the game will switch to the main menu once the user clicks on the screen.



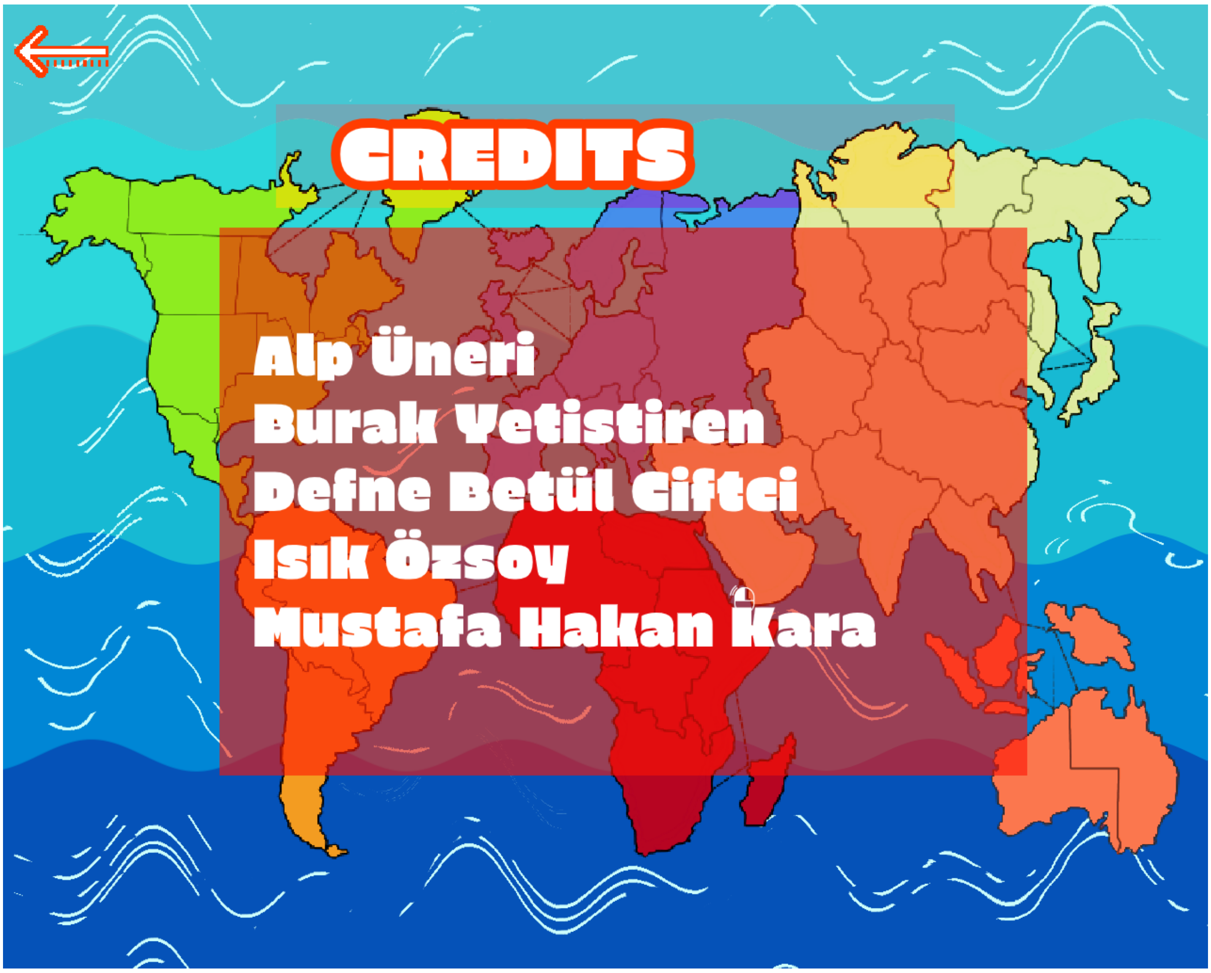
HOW TO PLAY

Annexation is a game based on the infamous board game Risk. To win, you need to conquer and take hold of all of the 42 territories. Once you don't have any territories left to defend, you lose.

You will start the game by the Territory Allocation phase, in which you choose one territory to allocate for yourself each turn, this is done by clicking a grey territory with the left mouse button. 🖱 You go to the next phase once you click on the orange Next Phase button on the left.

The next phase is Attack phase. You click

How to Play page consists of a lengthy explanation of the gameplay with mini icons for the mouse button if the explanation still isn't thorough enough and a Back button on the upper left side of the page. There is also a scroll bar for the How to Play page for which the scrolling is done by the user clicking on the bar and dragging it downwards or upwards.



CREDITS

Alp Üneri
Burak Yetistiren
Defne Betül Ciftci
Isık Özsoy
Mustafa Hakan Kara

The Credits page includes the names of the contributors to the final game, and a Back button.

4 Glossary

4.1 Territory

A territory is any individual region on the map of the game. A player wins once they conquer all of the territories within the board. Territories are painted in the color of the player controlling them on the map to show who they belong to, with as well a number in the middle of them showing the number of troops present on that territory. A territory will also be painted grey with a zero indicating no troops are present if no player is currently holding control of the territory.

4.2 Continent

A continent is a collection of territories. Continents range from small one such as Australia to large ones such as Asia. Once a player is able to conquer an entire continent for more than a turn, they will receive additional troops at the start of their turn, the number of which will be proportional to the size of the continent.

4.2.1 Target Continent

Players will be able to declare target continents at the start of the game. This will allow them to declare which continent they will be aiming to conquer before the game, and if they are able to conquer their target continent they will receive additional troops in addition to the ones they will have already received for conquering said continent. Two players may declare the same target continent at the start of the game. Sometimes referred to as “Goal Continent” as well.

4.3 Cards

If a player is able to conquer at least one territory during their turn, at the end of their turn they will be awarded a random card. These cards will be of any of the four following types: Infantry, Cavalry, Cannon, or Joker. If a player has three cards of matching type in their possession (with the joker cards being able to be substituted for any other type), they will be able to turn those cards in to receive a number of troops.

4.4 Player

A player will be the individual in control of an army, waging battles in order to achieve global domination and win the game. Each player will have their own starting territories, the same number of starting troops, as well as a color that will be used to mark territories currently under their control. A player will lose the game and be eliminated if they lose control of all of their territories and thus run out of soldiers, and will win the game by achieving global domination.

4.5 Troops/Soldiers/Unit of Troops

These all refer to the same thing, the troops/soldiers/unit of troops a player has on any territory is the number of deployable units that they have on that territory which they can use to attack any neighboring territories or defend that territory should another player decide to attack there.

4.6 Turn

Our game is turn-based, with play revolving around turns of players. A turn of a player is separated into three distinct phases; the troop allocation phase, the attack phase, and the fortification phase. At the beginning of their turn, players will also have the option to propose an alliance to any other player that is not already an ally, or secede from already formed alliances.

4.7 Troop Allocation Phase

This phase is the first phase of a player’s turn. In this phase players will decide where to place the troops that they will receive at the start of their turn. They will be asked to allocate all of the troops that they have received, and will be able to do so in any manner that they wish.

4.8 Attack Phase

This will be the second phase of a player’s turn. In this phase players will be able to attack neighboring territories of any non-allied other player. A player will have to have at least two units of troops present on any territory that they wish to launch an attack from. Players will also have the option to skip this phase entirely if they decide not to attack any territory during that turn.

4.9 Fortify Phase

This will be the third and last phase of a player’s turn. In this phase players will be able to choose a number of soldiers from one of their territories and transfer those soldiers to any other of their territories, provided that those two territories are connected, with either the player’s own or allied players’ territories in between the two territories.

4.10 Airport

Players will have the option to exchange five units of troops with an airport on any territory in which they have at least six soldiers. Doing so will place an airport on that territory, allowing players to attack territories of up to three units away from the territory in which an airport is present. If a territory with an airport is conquered by a different player, the airport will be removed; presumed to have been destroyed during the battle.

4.11 Alliance

Players will have the ability to form alliances with each other, provided that there are more than two players in the game. Once two players are allied, they will be able to freely cross each other’s borders to either attack a territory or during their fortification phase. Alliances can be broken during the start of either player's turn, and allying players will not be able to attack each other for the duration of their alliance.

4.12 Rock-Paper-Scissors

Once a player declares combat on another player, battles will take place in the form of rounds of rock-paper-scissors. After each round, the player who has lost the round will lose a single unit of troops. The battle will be over once the defending side has run out of troops or all the attacking troops have been defeated.

4.13 Global Domination

When a player is able to conquer all of the territories on the map, they will have achieved global domination. This is the winning condition for any player, the game ends once a player has achieved global domination and that player will be declared the winner of the game.

5 Resources

“The #1 Development Tool Suite,” *Ideal Modeling & Diagramming Tool for Agile Team Collaboration*. [Online]. Available: https://www.visual-paradigm.com/. [Accessed: 28-Oct-2020].

“advanced photo editor,” *Photopea*. [Online]. Available: https://www.photopea.com/. [Accessed: 28-Oct-2020].

Author Amanda Athuraliya Amanda Athuraliya is the communication specialist/content writer at Creately, A. Athuraliya Amanda Athuraliya is the communication specialist/content writer at Creately, and A. Athuraliya, *Sequence Diagram Tutorial: Complete Guide with Examples*, 19-Aug-2020. [Online]. Available: https://creately.com/blog/diagrams/sequence-diagram-tutorial/. [Accessed: 29-Oct-2020].

B. Myers, *Requirements Specification*. [Online]. Available: http://www.cs.fsu.edu/~myers/cop3331/notes/requirements.html. [Accessed: 20-Oct-2020].

Concept: Use-Case Model. [Online]. Available: http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use_case_model_CD178AF9.html. [Accessed: 29-Oct-2020].

Download free icons, music, stock photos, vectors. [Online]. Available: https://icons8.com/. [Accessed: 28-Oct-2020].

G. Robinson, “The Strategy of Risk.” [Online]. Available: https://web.mit.edu/sp.268/www/risk.pdf.

I. Alexander, “A range of techniques for engineering better systems,” *Scenarios in Systems Engineering*. [Online]. Available: http://www.scenarioplus.org.uk/papers/scenarios/intro_to_scenarios.htm. [Accessed: 29-Oct-2020].

K. Fakhroutdinov, “State Machine Diagrams,” *UML graphical notation overview, examples, and reference*. [Online]. Available: https://www.uml-diagrams.org/state-machine-diagrams.html. [Accessed: 29-Oct-2020].

N. T, “What is Class Model? Objects, Class, Relations,” *Binary Terms*, 28-May-2020. [Online]. Available: https://binaryterms.com/class-model.html. [Accessed: 29-Oct-2020].

“Risk (game),” *Wikipedia*, 21-Oct-2020. [Online]. Available: https://en.wikipedia.org/wiki/Risk_(game). [Accessed: 28-Oct-2020].

Risk Oyun Kılavuzu. [Online]. Available: https://www.hasbro.com/common/documents/dad2886d1c4311ddbd0b0800200c9a66/D3A98A4650569047F5C718754E1236CF.pdf.

“UML - Activity Diagrams,” *Tutorialspoint*. [Online]. Available: https://www.tutorialspoint.com/uml/uml_activity_diagram.htm. [Accessed: 29-Oct-2020].

UML Class Diagram Tutorial. [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/. [Accessed: 28-Oct-2020].