Bilkent University

Department of Computer Engineering

CS 319

Object Oriented Software Engineering

*Game of “Risk” Final Report*

Group 1-E

Mehmet Ege Acıcan-21602186

Ali Ata Kılıçlı-21301530

Ekin Üstündağ-21602770

Mehmet Akif Kılıç-21502487

Seman Arda Yuvarlak-21501402

Contents

[1 Introduction 1](#_Toc254263637)

[2 Design Changes 1](#_Toc254263638)

[3 Lessons Learned 1](#_Toc254263639)

[4 User’s Guide 1](#_Toc254263650)

4.1 System Requirements and Installation\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1

4.2 How to Use\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1

4.2.1 Game Overview\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.2 Menu Operations\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.3 Starting a New Game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.4 Loading a Previous Game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.5 Adjusting the General Settings\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.6 Highlighting the Statistics\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.7 Adjusting the Game in Pre-Game-Settings Menu\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.3 Gameplay Options\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1.Introduction

In this report, our aim is to discuss the implementation process, the design decisions, design changes and offer an users guide on Game of “Risk”. This section specifically focuses on the implentation process and the current status of the implementation.

We started to implement the game after our design, therefore our design played a key role in the implementation of our game. We followed the specific criteria of our design report strictly: Decomposing and loose coupling. We decided to decompose our system into subsystems, assigning each of the subsystems as tasks. This was helpful for our work distribution. Also, even though the game is dependent on subsystems working cooperatively with each other in a synchronous way, we designed the subsystems in loose coupling way. This helped to reduce the possibility of a dependancy problem between the classes and this also allowed us to work independently of each other and thus, allowing each one of us to bring some unique and new features to our game.

Our meetings were face to face as much as possible. If face to face meeting was not possible, the meeting was done by whatsapp. In these meetings, we discussed the design, the structure and took specific design choices along with updating the implementation.

In this stage of the Game of “Risk” implementation, we have implemented the gameplay layer of the game. We have implemented the required classes such as Player,Map, Game and we have implemented their attributes along with their methods. Therefore, the main gameplay of the game is almost complete.

The GUI layer and the Data layer are not implemented so far. Therefore, the menus and the loading and saving of the game is still not implemented.

2.Design Changes

We have decided to make the system in MVC( Model View Control) so we figured we required specific model and controller classes. Therefore, in our design report we have included specific model, view and controller classes. However, as we have made progress with the implementation and thought again about specific aspects of our game, we realized that specific classes might not be necessary and thus they should be removed.

Firstly, based on our analysis feedback, we have decided to update the player- color situation. At first, we have thought that each color allows to start the game with different advantage however we have later figured out that this might cause problems in a fair gameplay and affects the number of players that are participating in the game. Therefore, we are thinking about removing the color aspect from our design and trying to come up with a new creative way to add uniqueness to our game.

Another possible design change is to remove the troop-rank system that we have promised in our analysis report. This is because we want to increase the simplicity of the game and therefore decided to let attack and defence mechanism to be purely basic mathematical operations. We are still unsure about this design change however if it occurs, we are going to remove the classes responsible for troops and their specific ranks.

Our primary concern was the gameplay layer that was described in our design report therefore we did not focus on the data layer right now. Thus, we are in the beginning steps of the Data layer part therefore no changes have been made to data layer so far.

However, in our design layer, we had figured out that it would be more beneficial if we had created another class for managing the main gameplay of our system. Therefore, each class would be responsible just for their own attributes and their own unique methods while gameplay aspects are being handled by another class. Therefore we are considering adding specific proxy classes that are going to make the implementation of the game much more faster and thus increase our games usability requirement.

We were mainly focused on the gameplay aspect of the game so far and therefore we did not focus on the GUI layer that was described in our design report. Since we are in the beginning steps of the GUI layer therefore no change has been made on the User Interface right now.

3.Lessons Learned

During our implementation of the game, we have understood the importance of analysis and design stages and most importantly its synchronous action with the implementation. We have tried to approach to the project in a Waterfall lifecycle approach which is focusing on speculative requirements and design processes before implementation and implementing the project based on the speculated requirements and designs. However, we have learned that iterative approach would be better for this project, that is, implementing and completing the requirements and design synchronously. We have learned that implementation can allow us to fix the errors that we could not foresee in the design and therefore allowing to make our design better.

Another valuable lesson that we have learned is the importance of groupwork and the importance of work distribution. In our project, we have separated our tasks and we tried to implement the loose coupling structure to our roles and works as well however we have realized that we have limited independency since our tasks are supposed to work cooperatively and therefore we had to be in contact with each other about our projects regularly. Also, we had understood that work distribution should be equal and therefore we should not rely on one person too heavily while not sharing the work with other members.

Other lessons that we have learned is that flexibility can be helpful for making progresses in group work. Also, we have witnessed the effect of change in a software project. We had to rethink about our design for multiple times and as we progress in the project, we are certain that we will reconsider specific updates on our project again

4. Users Guide

4.1 System Requirements and Installation

Game of “Risk” can be run in platform and it does not require any installation. It only requires Java Runtime environment to be installed in order to run since it is a single executable JAR.

4.2 How to Use

4.2.1 Game Overview

Game of “Risk” is a strategic conquest game which is designed for desktops and laptops. After the initialization, the user sees the main menu that allows the player to start a new game, load a previous game, adjust the settings, see the credits or exit the game. Players are able to play in 3 different gameplay mods and they are able to save their game, load their saved game and delete their previous games.

4.2.2 Menu Operations



Figure 1: Main menu of the Game of "Risk"

4.2.3 Starting a new game

Players are able to start a new game by selecting new game from the main menu, then they choose whiich gameplay mod they are going to play the game in. These gameplay mods can be Classical Mod, Time Mod and Golden Territory mod. After choosing the gameplay mod, the pre-game-settings menu opens up and player starts to adjust the settings just before the game.

4.2.4 Loading a previous game

Players are able to load a previous game that they have saved and continue to play their game from where it has been left. The players are able to save up to 6 games because there are 6 slots to save the game into.

4.2.5 Adjusting the general settings

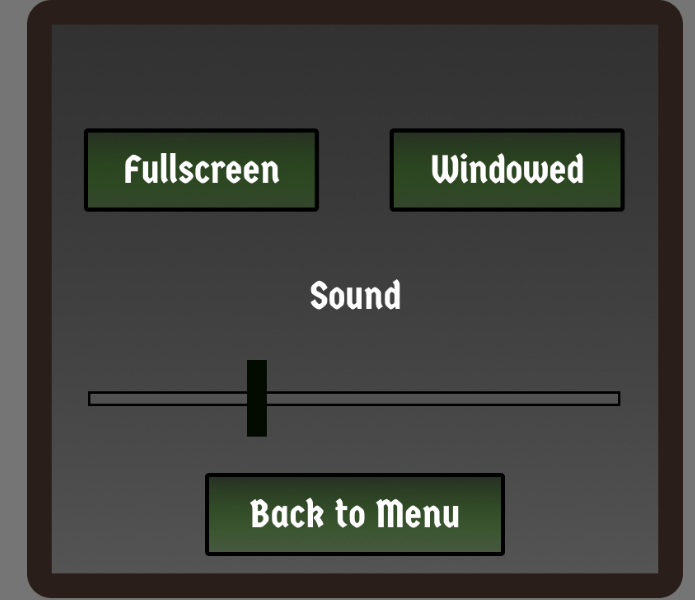


Figure 2: General Settings Menu

The players are able to change the game into fullscreen, or window mode and they are able to adjust the volume of the game.

4.2.6 Highlighting the Statistics

If there was a game that was being played on time mod, the players are able to see the highscore of the winners that is , the least time that it took for a player to finish the game, regardless of the gameplay mod

4.2.7 Adjusting the game in Pre-Game-Settings Menu

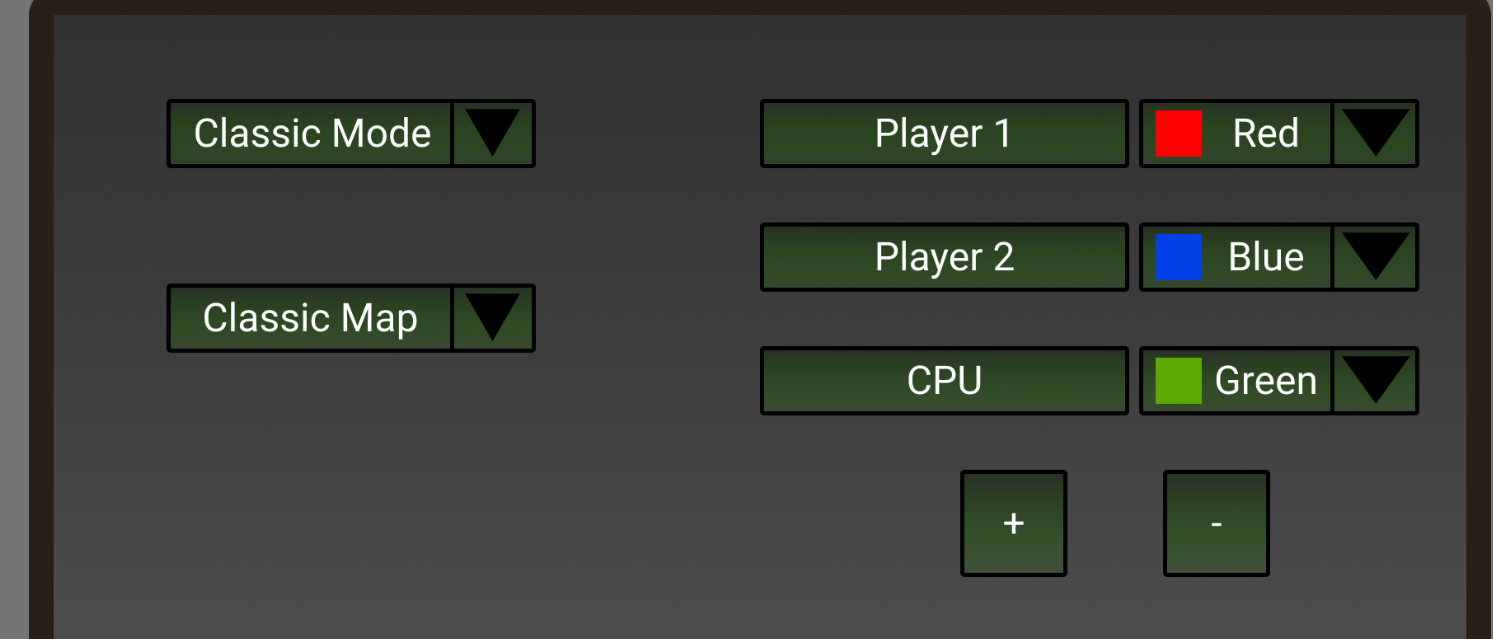


Figure 3:Pre-Game-Settings Menu

The players are able to change the gameplay mod and they are able to change the map that is being played( Regular Risk map or Tamriel Map) . The players are able to add or deccrease the number of players from + and – buttons.The players are able to adjust the colors of themselves.

4.3 Gameplay Mechanics



Figure 4: In Game Gameplay Menu

After the game starts, the players have three stages of their turn: Recruit, attack and distribute. These stages follow each other respectively. Which stage the player is in is highlighted on the right corner. Which territory that is occupied by which player is highlighted by the players colors. Information shows the total territory number that the player owns and how many troops that the territories has specfically. In the attack stage, the player first selects which territory that he/she is going to attack from by clicking on to the territory and then he/she selects the territory that he/she is going to attack, again by clicking to the desired territory. The player selects with how many troops he/she is going to attack to the desired territory and then the dices are rolled. If the attacking players dice is higher than the defending player, the number of troops in the attacked territory decreases by the number of troops that has attacked to the territory.