OOP Project

# Related image**Project Implementation Report: Iteration 1**

CS 319 Object Oriented Software Engineering  
Eray TÜZÜN  
Group 1-I: Private Moon Inc.

Project Group Members

• Doğa Oruç – 21602022

• Çağrı Orhan - 21503102

• Fazilet Simge Er - 21602358

• Kaan Ünlü - 21602622

• Ege Turan – 21502441

Contents

[**Project Implementation Report: Iteration 1** 1](#_Toc4887029)

[1.0 Introduction 3](#_Toc4887030)

[2.0 Design Changes 3](#_Toc4887031)

[3.0 Lessons Learnt 4](#_Toc4887032)

[4.0 User’s Guide 5](#_Toc4887033)

[4.1 System Requirements & Installation 5](#_Toc4887034)

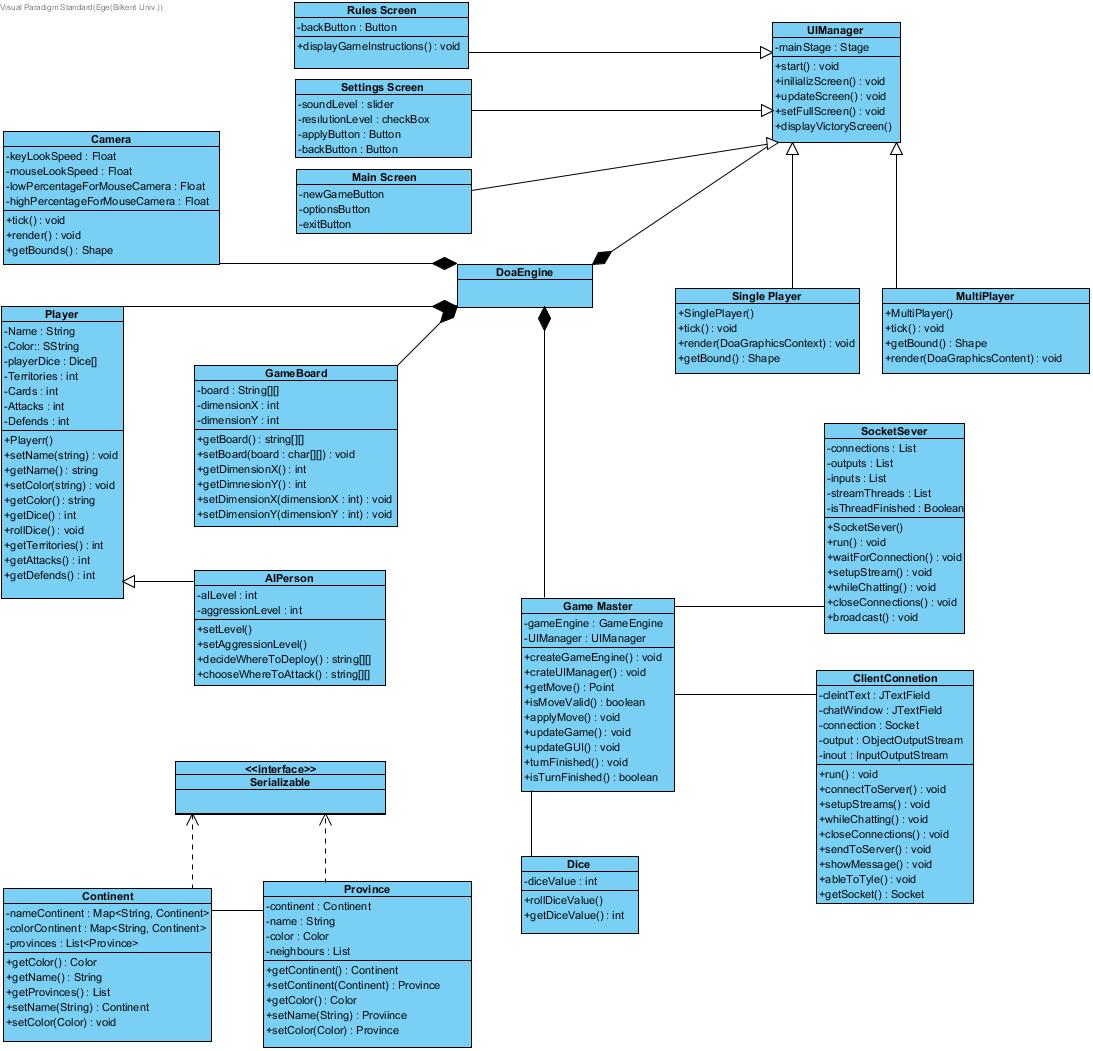
[5.0 Glossary & references 5](#_Toc4887035)

# Introduction

The implementation of the multiplayer infrastructure, the multiple map support (mod support) and some of the project’s Graphical User Interface (GUI) elements – main menu, main game map etc. – are mostly completed. So far, we’ve worked on the GUI most of the time, and are going to continue with the implementation of the game’s logic. In addition to this, we will be working on the implementation of language support, and other game mods which will include different maps. Furthermore, we will continue our work on delivering a colour-blind-friendly game.

# 2.0 Design Changes

When we first started to design our classes we came up with following class diagram:



##### Figure 2.1 Class diagram at the time of the first iteration analysis document

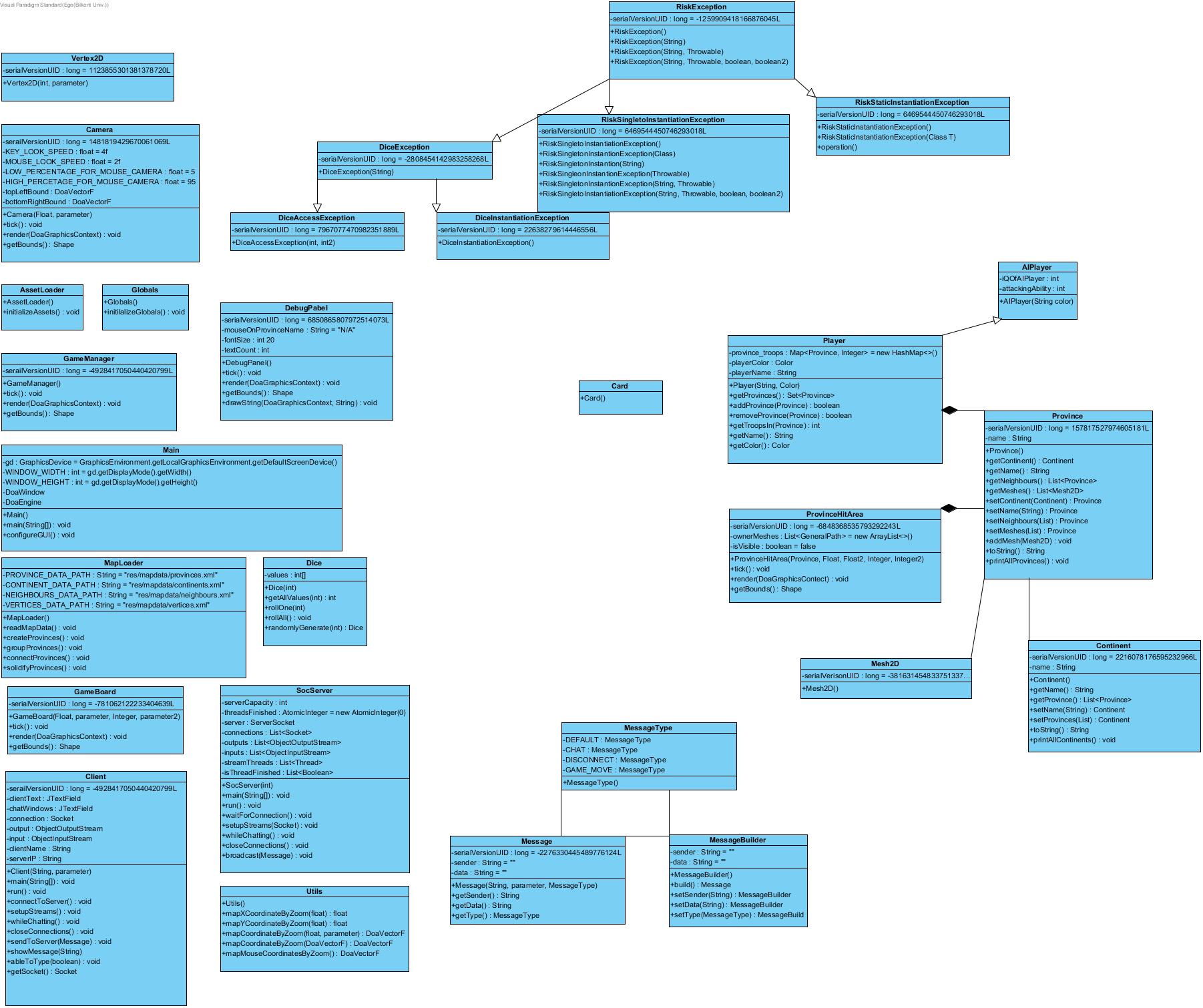
As time went on and we fleshed out our design more, we changed and added some classes and added Exception classes. All of the exception classes use inheritance and handle one of the problems that we faced during the development process.

Secondly, we added the new MessageType and Message classes to improve communication and data transfer more efficiently.

Thirdly, we added ProvinceHitArea to determine the vertices of the province. ProvinceHitArea takes provinces and with the read vertex values, it determines the physical span of that province.

Our class system is on the development process and project is still changing as this is still the first iteration. After we complete the first iteration process we will a give more detailed and developed class diagram and we will make more understandable explanations for the change we made.

This is our updated class diagram for now:



##### Figure 2.2 Class diagram at the time of the first iteration design document

The way the system is implemented now, the sequence diagram for attacking a province is as below:

##### C:\Users\musta\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Sequence Diagram1.jpg Figure 2.3 Sequence diagram for attacking a province at the time of the first iteration implementation document

The diagram shows the simple flow of how attacking a province would work in game. As this aspect of the game is not complete yet however, it is, similar to the class diagrams, subject to change.

# 3.0 Lessons Learnt

We encountered a lot of problems while implementing and designing this project. The first important issue was about a lack of communication. As a group, we were meeting only 2 times a week. This allowed occurring disruptions in the project due to lack of sufficient communication. There were problems while bringing the parts of the project together. We decided to meet more often because it was obvious that something was going wrong. We've increased our group meetings’ frequency. Also, we have started to use the “Discord” app which is supplying free voice chat and screen sharing. We saw that more and instant communication can solve many problems. We started to do our jobs with better synchronization and faster. The solution process gained speed through communication.

Another issue was about disagreements. As is known, the Risk is a board game and there is no single official version of the game [1]. Designing the game on a digital platform required many decisions about graphics and mechanics. At some points, we had difficulties in the decision-making process. When deciding on the mechanics and graphics of the game different ideas were raised. Our solution was making assessments and trying to find out which one would look and feel the most impressive with the least amount of work, so that our final product would be as complete as it could be (and hopefully with the most extra features) by the final deadline. We were meeting at a common point after long reviews. This really helped us to maintain the decision-making process.

Another compromise example occurred because different group members are using different coding styles (such as how everyone placed their curly parentheses). When merging different individual’s codes into the same project, it was hard to read and write. We decided to choose the coding style by voting. Decisions were determined by the votes of the members of the group so that no one would resist the decisions. This provided a democratic solution to the discussions. We had set a consistent coding style that everyone will follow. This aims to increase the maintainability of the code.

We also realized the psychological effect of working as a group. The bystander effect occurs when the presence of others discourages an individual from intervening in an emergency situation. When an emergency situation occurs, everyone has a tendency to put the responsibility on someone else. We encountered with bystander effect so much on the beginning of the project. Work distribution in advance of deadlines helped us solve this problem.

All in all, starting to work with a group can lead to many problems inside the group. We learned that a problem within the group can only be solved by within a group. We saw that synchronous communication, compromise, reviews, and brainstorming is essential tools for managing a group project.

# 4.0 User’s Guide

## 4.1 System Requirements & Installation

When the implementation is complete, Risk will be able to be run on any computer environment, since it will be released both as a JAR (.jar) file and an EXE (.exe) file. To install the game, one should have either of the game’s executables, and simply run it. This action will load the game, while displaying a splash screen to the user. After the game is fully loaded, the user will be redirected to the game’s main menu.

A more detailed version of the installation guide will come when it is implemented.

## 4.2 How to Use

So far, the game can only be started from an Integrated Development Environment (IDE). The user will run the project from an IDE, and the main menu will appear. There are 5 buttons with only the 2 of them functioning at the moment, on the main menu. The first button is one of the functioning buttons, called the “Play Offline” button, which will redirect the user to the original Risk map which the game will be played on. The second functioning button is the “Exit Button” which is the last button of the main menu, and closes the game when clicked.

On the Risk map (the main game screen), each province and continent with different colour can be seen. On the left top of the page, the position of the camera, the top left and bottom right bounds of the camera, absolute and mapped mouse positions, zoom degree and the name of the province which the mouse was last on can be seen. If a province is right-clicked, the vertices of the province will be visible, and if right-clicked again, they will be invisible. Similarly, if a province is middle-clicked, the whole province will become invisible, and if middle-clicked again, it will be visible.

As for the starting point of the mod support, in the Risk map, when the “V” button on the keyboard is clicked, a warning-like text which says “VERTEX LOGGING ENABLED!” will appear on the top left of the screen, and with every left-click on the page, the x’s and y’s of the clicked point will be written in XML format in a text file named “newHopes.txt” which is located in the project directory. With pressing the “V” button again, the warning-like text will disappear, and the left clicks after that will not have any effect on “newHopes.txt”. At the start and end of each of the vertex logging action, a one-line log which contains the day and hour-minute-second will be written on the text file as a separator. The contents of this file can be pasted to wanted spaces in the provinces XML file to apply the vertex data to the game, making the creation of functional maps in game easy.

# 5.0 Glossary & references

|  |  |
| --- | --- |
| [1] | Brilliant Maps, “27 Best Risk Board Game Versions Based On Real Player Reviews,” 28 August 2016. [Online]. Available: https://brilliantmaps.com/risk/. [Accessed 31 March 2019]. |