CONCORDIA UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE AND SOFTWARE
ENGINEERING

SOEN 6441, Winter 2019

Instructor: Amin Ranj Bar

RISK Game (Build-1)

Architectural Design

BY:

Team - 13

Jemish Kishor Paghadar	(40080723)
Deep Alpesh Patel	(40087798)
Raj Patel	(40085012)
Hardik Vora	(40087606)
Dolly Modha	(40084358)

Index

Introduction

Develop a RISK game using Model View Controller (MVC) software design architecture with iterative development to deliver working modules in small builds. It was an effort to use extreme programming key features such as Pair programming, Collective ownership, Coding Standards and many more.

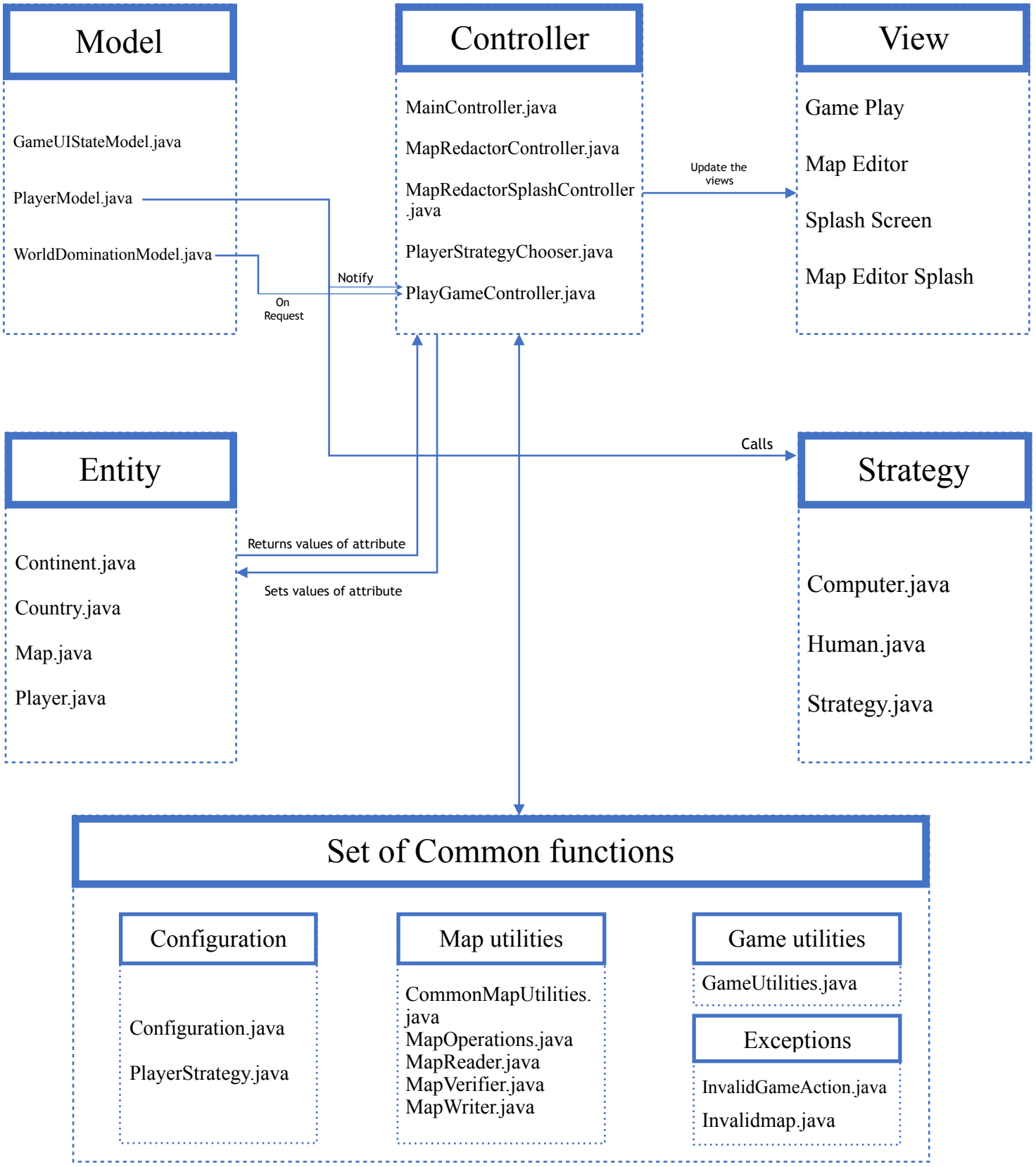
1.Scope

The scope of the build 1 is as per the instruction guidelines for the build:

- Map Editor:
    - Create a new map file
    - Edit an existing map file
    - Add/Update/Delete Continent , Country and Adjacent Country
    - Make sure that the integrity of the connected graph is maintained.

- Game Play:
    - Assigning country to player
    - Player can assign armies to each country in round robin manner
    - With proper calculation of armies, Reinforcement phase is implemented
    - With a valid fortification move, Fortification phase is implemented

2.Architecture Design



3.Modules Description

3.1.Controllers

File_Name	Description
PlayGameController.java	It is a mediator between the GameUtilities class and gameplay.fxml file. It captures all the user action like creation of player, assigning armies, all the three phases of the risk game etc.
MapRedactorController.java	It is a mediator between the MapOperations and the mapeditor.fxml. It captures all the user actions like add, update and delete continent or country or adjacent countries.
MapRedactorSplashController.java	It controls the user action from mapeditorsplash.fxml and calls new controller to open a pane of javafx.
MainController.java	This file handles the user action from main screen.
PlayerStrategyChooserController.java	It allows to choose a player Strategies for selected number of player (i.e Human or Computer)

3.2.Entity

File Name	Description
Map.java	It contains all the information of the Map and a list of the continents.
Continent.java	It contains all the information of the continent and a list of all the countries that belong to a continent.
Country.java	It contains the information of the country like name, a reference to which continent the country belongs, list of all the adjacent country, count of armies currently residing on the country.
Player.java	It contains all information related to a player and the number of armies assigned to the player.

3.3. Maputilities

File Name	Description
MapReader.java	It reads the map file format and parsing in to Map object and also checks for the validity of the data of the map file.
MapWriter.java	It is responsible for writing the Map object to the file.
CommonMapUtilities.java	Contains all the common method of the map like: saving map object, opening a dialogue box etc.
MapOperations.java	It perform operation like addition or update or deletation of country or continent.
MapVerifier.java	This class validates the map.

3.4 Configuration

File Name	Description
Configuration.java	It defines global constant variables and messages for the application
CardType.java	Enumeration for defining different card types.
PlayerStrategy.java	Enumeration for defining player strategies.

3.5 Gameutilities

File Name	Description
Gameutilities.java	It handles the common operations for the game play phase.

3.6.Model

File_Name	Description
PlayerModel.java	It handles operations of the players such as reinforce, fortification and many others.
WorldDominationModel.java	It provides the data for the chart to the PlayGameController.
GameUIStateModel.java	It stores the state of the button.

3.7.Main

File_Name	Description
Main.java	Entry point for the application

3.8.Exception

File_Name	Description
InvalidMap.java	It manages exception of the map validation.
InvalidGameAction.java	It manages exception of the Gameplay phase.

3.9.Strategy

File_Name	Description
Human.java	It implements Human Strategy when player want to play manually.
Computer.java	It implements Computer Strategy when player want to play with computer.
IStrategy.java	This interface defines methods which are to be implemented by the strategies.

4.Test Cases (Junit) Description

Gameutilities	
File_Name	**Description**
GameUtilitiesTest.java	This is a test class for GameUtilities. i.e. one of its method tests the allocation of country to player is working or not.
GameUtilitiesTestSuite.java	

Main	
File_Name	**Description**
MainTestSuite.java	This is a test class for running all test suits. (MapUtilitiesTestSuite.class, GameUtilitiesTestSuite.class, ModelsTestSuite.class, StrategyTestSuite.class)

Maputilities	
File_Name	**Description**
MapOperationsTest.java	This is a test class for Map operations. i.e. one of its method tests the functionality to map a Country with the continent.
MapReaderTest.java	This is a test class for Map Reader. i.e. one of its method tests the map which has countries without continents.
MapVerifierTest.java	This is a test class for Map varifier. i.e. one of its method tests if the program can load unconnected continent map.
MapUtilitiesTestSuite.java	This is a test class for running all test suits in Maputilities i.e. (MapReaderTest.class, MapOperationsTest.class, MapVerifierTest.class)

Models	
File_Name	**Description**
PlayerModelTest.java	This is a test class for Player Model. i.e. one of its method tests the number of armies for different players.
ModelsTestSuite.java	

Strategy	
File_Name	**Description**
ComputerTest.java	This is a test class for the strategy Computer. i.e. one of its method tests the normal functionality of the getRandomCountry method.
StrategyTestSuite.java	

5.Tools

Tools	Description
Eclipse	IDE for the game development
Scene Builder	It is an open source JavaFX system used for UI design and gives a skeleton of the events to be implemented in controller.
JavaFx	Library to control the UI component
FXML Editor	To generate the UI components
Junit 4	Junit 4 for writing test cases
Maven	Maven as a build automation tool to manage all project dependencies.