SOEN 6441: Advanced Programming Practice

Winter 2019

Build 1

Submitted To: **Amin Ranj Bar**

Submitted By: **Team 18**

Sr. No.	Name	Student ID
1	Nikitha Jayant Bangera	40088393
2	Shresthi Garg	40105918
3	Anusha Keralapura Thandavamurthy	40102962
4	Sumeetha Kasulabad	40078178
5	Shiva Shamloo	40094742

Architecture for Risk Game – Model-View-Controller

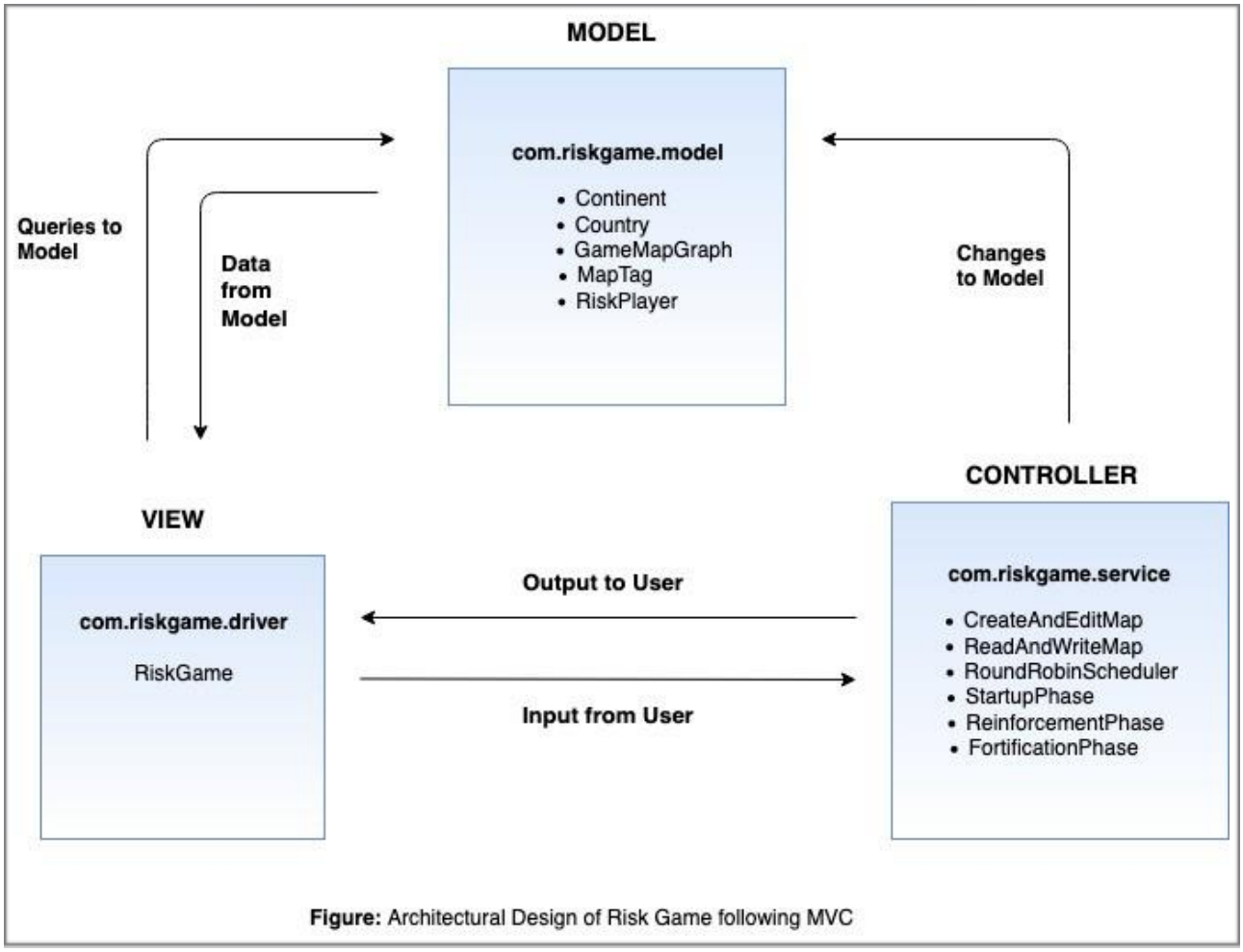

Figure: Architectural Design of Risk Game following MVC

Following are description for each layer

1. Model – Has core classes for the game and related logic

2. View – Has classes related to view

3. Controller – Acts as a bridge between model and view.

- **Modules in View:** View has only one driver.

  1. RiskGame Class: Class used to provide an interface for the players to interact with the game. It launches the game and window is displayed where user can create or load map to begin the game.

- **Modules in Controller:** We have six services in controller with following functionality**.**

1. **CreateAndEditMap**: This class is used to create a connected map from the user input and provides a method to modify the map data.
2. **ReadAndWriteMap**: This class is used to write all the map details to the file and read the contents of the file to validate or edit.
3. **RoundRobinScheduler**: This class provides functionality for round robin traversal among the players of the game.
4. **StartupPhase**: This class takes the data from GameMapGraph module and initialises data for country, players and armies. It has a method which allocates the countries and armies to players.
5. **ReinforcementPhase**: This class has methods which gives some extra armies to the player based on assigned countries.
6. **FortificationPhase**: This class has a method in which player can move armies from one country to another.

- **Modules in Model:** We have five model classes which have following functionality.

1. **Continent**: This class model the continent. It is a set of countries where each country belongs to only one continent.
2. **Country**: Used to model the country. Each player will be assigned with few countries and player aims at occupying all the countries.
3. **GameMapGraph**: Class used to model the map. It has all the details of the map and its content.
4. **MapTag**: Used to model the Map tag data present in map.
5. **RiskPlayer**: Used to model the players of the game.

Flow of the game as per the Build 1:

- The Risk game starts by giving options to the user to
  - create a new map which is done by calling the **newMapCreation()** method.
  - load an existing map into the game by calling the **uploadMap(fileName) method**.
- The user is given a choice to proceed with the game or discontinue once the map is created or loaded.
- **startGame():** This method redirects the user to either play the game or quit as per the user's choice.

- **gamePlay(mapGraph):** If the user wishes to proceed then the application prompts to enter the number of players to play the number. There can be two to six players who can play the game. Once the number of players is chosen, then names must be given for each player. The mapGraph object contains all the details of the map which will be used for the game.
- **allocationOfCountry(mapGraph)**: The game begins with the **Start-Up Phase** wherein the players will be randomly assigned with the available number of countries from the map.
- **allocationOfArmyToPlayers():** The initial army count to be assigned to be assigned to each player will be decided based on the total number of players.
- **allocationOfArmyToCountriesInitially(mapGraph):** One army each will be deployed in each of the player's owned countries.
- **allocationOfRemainingArmyToCountries():** Then the players are asked to deploy their available number of armies on each of their countries, as per their choice and game strategy.
- Once the initial game setup is complete, then the war begins!
- The players play their turn in a round-robin fashion. Each player has to play the Reinforcement phase, Attack phase (not included in build 1) and Fortification phase.
- **startReinforcement(player, mapGraph):** In the **Reinforcement Phase**, each player gets additional armies based on the number of countries owned and also if a player happens to own an entire continent, then they will get extra armies as per the control value of the continent owned. This is achieved using ***armiesToBeAssigned*(player, continent)** method. The player can then deploy these additional armies along with the remaining number of armies on each of the player's countries. The player is given a choice to proceed with Reinforcement phase or proceed further with the next phase.
- **startGameFortification(player, mapGraph):** After the **Attack Phase** (Not part of build1), comes the **Fortification Phase**. In this phase, the player can move armies from one country to another, if the latter country has fewer armies left after the attack phase.
- **moveArmies(givingCountry, receivingCountry, countOfArmies)**: The player can move armies only if the countries is owned by him/her and also there must be adjacency between the two countries in-order to move the armies between them. The player also has a choice whether or not to play the Fortification phase.