**Architecture Design**

**GOAL**

Risk game is console based application in which goal is to occupy every territory on the map and in doing so, eliminate the other players.

**GAME PLAN**

* Game starts in console window
* It has two options
* Upload Map
* Create Map
* If upload map is selected
* User is given an option to select from a list of available map files.
* Map file is loaded from user’s choice and validated for correctness.
* Once the file is verified and is correct user will enter number of players and their names
* Assignment of the player number in random order.
* Assignment of territories in random order.
* Start up phase comprises of assignment of armies and territories in random fashion.
* Once start-up phase is done, reinforcement phase starts. A player is given a set of armies based on risk rules and is asked to place the armies in the territory the player owns.
* On completion of Reinforcement phase by the player, fortification phase begins. In this phase the player is asked to move armies between the territories he owns. Fortification is done only between the immediate adjacent countries the player owns.
* If selected create map
* Player has option to create a map from scratch.
* Once player creates the map as per his choice, player will be given an option to edit the created map.
* In the edit option player is given a choice to add or delete either continent or territory based on his choice.
* Once edit is done the map is verified for correctness.
* If the file is correct user will enter number of players and their names.
* Assignment of the player number in random order.
* Assignment of territories in random order.
* Start up phase comprises of assignment of armies and territories in random fashion.
* Once start-up phase is done, reinforcement phase starts. A player is given a set of armies based on risk rules and is asked to place the armies in the territory the player owns.
* On completion of Reinforcement phase by the player, fortification phase begins. In this phase the player is asked to move armies between the territories he owns. Fortification is done only between the immediate adjacent countries the player owns.

**MVC ARCHITECTURE**

Controller

Model

View

* Controller

|  |
| --- |
| M**ap** **Builder**  **Controller** |
| loadMapData( ) |
| getTerritoryList( ) |
| getContinentList( ) |
| getAdjucencyMap( ) |
| getMapUploadStatus( ) |
| setMapUploadStatus( ) |
| getIdByTerritoryName( ) |
| parseMapFile( ) |
| addContinents( ) |
| addTerritories( ) |
| addTerritoriesToContinents() |
| buildAdjancencyMap( ) |

|  |
| --- |
| **Game**  **Controller** |
| main( ) |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Player**  **Controller** |
| addPlayers( ) |
| getRiskPlayerList( ) |
| getPlayerNameList( ) |
| setRiskPlayerList( ) |
| setPlayersNameList() |
|  |
|  |

|  |
| --- |
| **Reinforcement**  **Controller** |
| getReinforcedMap( ) |
| calculateArmy( ) |
|  |
|  |
|  |
|  |

* Models

|  |
| --- |
| **Player** |
| String playerId |
| String playerName |
| ArrayList occupiedTerritories |
| ArrayList occupiedContinents |
| int armiesOwned |
|  |
|  |

|  |
| --- |
| **Territory** |
| String territoryId |
| String territoryName |
| int armiesPresent |
| String continent |
| int continentId |
| RiskPlayer territoryOwner |
| ArrayList adjacents |

|  |
| --- |
| **Continent** |
| String continentId |
| String continentName |
| int controllValue |
| ArrayList includedTerritories |
|  |
|  |
|  |

**MAP VALIDATION FUNCTION**

Function Name: validateMap(File file)

Return Type: Boolean

🡪validateSyntax(file) – Validate the tags and structure of file

🡪processFile(file) – Scan each line from the file

🡪validateDuplicacy(fileContent) – Check for territory, continent and adjacency duplicacy

🡪processContinents(fileContent) – Creates ArrayList of Continents

🡪processTerritories(fileContent) – Creates ArrayList of Territories

🡪processAdjancancy(fileContent) – Creates ArrayList of Adjacent territories

🡪addAdjacentTerritories(Territories) – Adds Adjacent territories to each

🡪territoriesToContinents – Add Each territories to the corresponding continents

🡪validateConectedMap(territoriesArray) – Validate the connection between territories

Use DFS to check the connection between territories

**TOOLS USED**

**1) Git**

Git is an open source distributed revision control system which brings together the world's largest community of developers to discover, share, and build better software.

**2) Javadoc**

Javadoc is a software tool part of Java SDK for generating API documentation from Java source code augmented with special tags in the code’s comments.

**3) Eclipse IDE**

Eclipse is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), and is the most widely used Java IDE. It contains a base [workspace](https://en.wikipedia.org/wiki/Workspace) and an extensible [plug-in](https://en.wikipedia.org/wiki/Plug-in_(computing)) system for customizing the environment.

**4) Junit Framework**

Junit framework lets a programmer associate classes and methods to corresponding test classes and test methods. Automation is achieved by automatically setting up a testing context, calling each test case, verifying their corresponding expected result and reporting the status of all tests.

**State Diagram**

