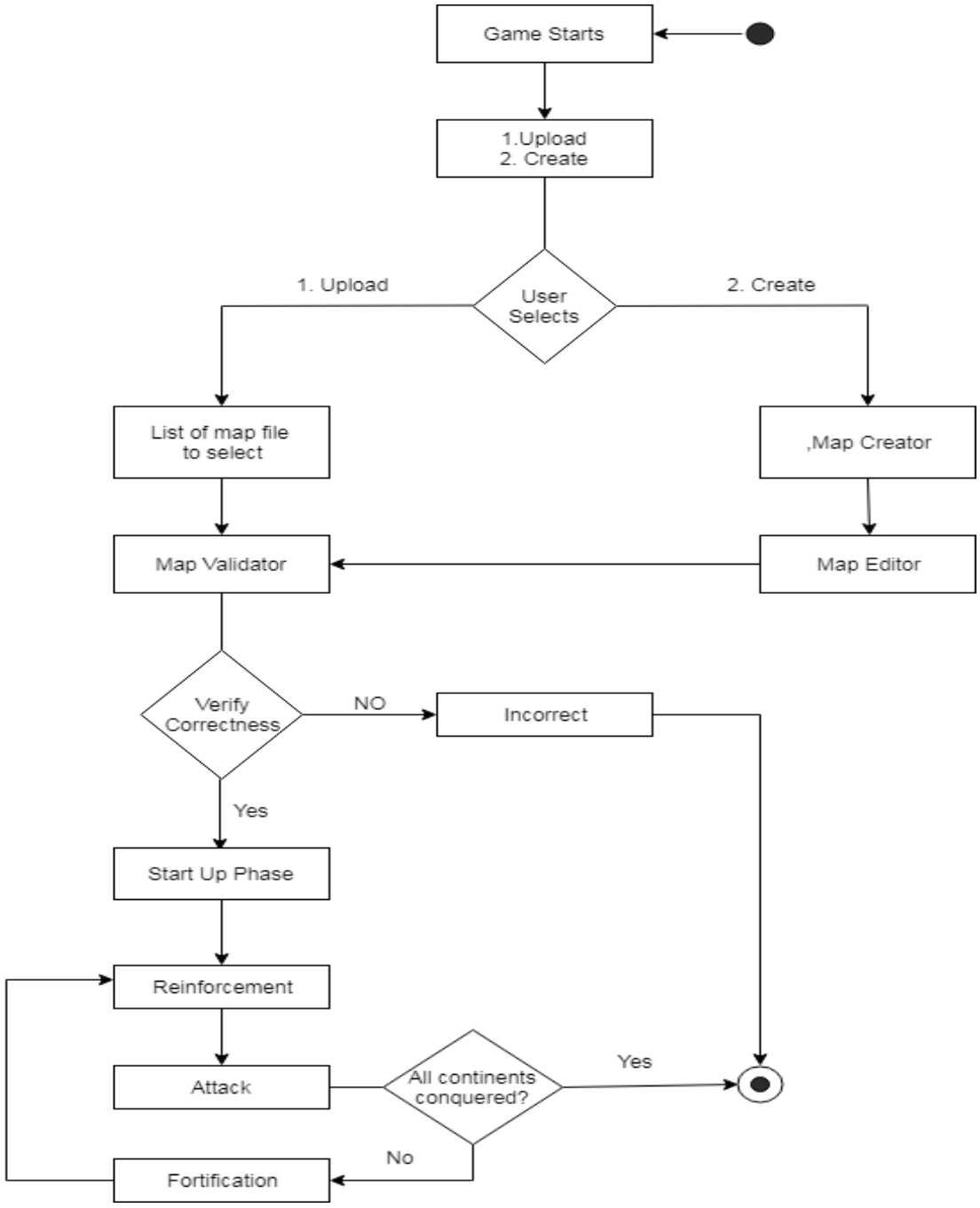
Architecture Design

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.

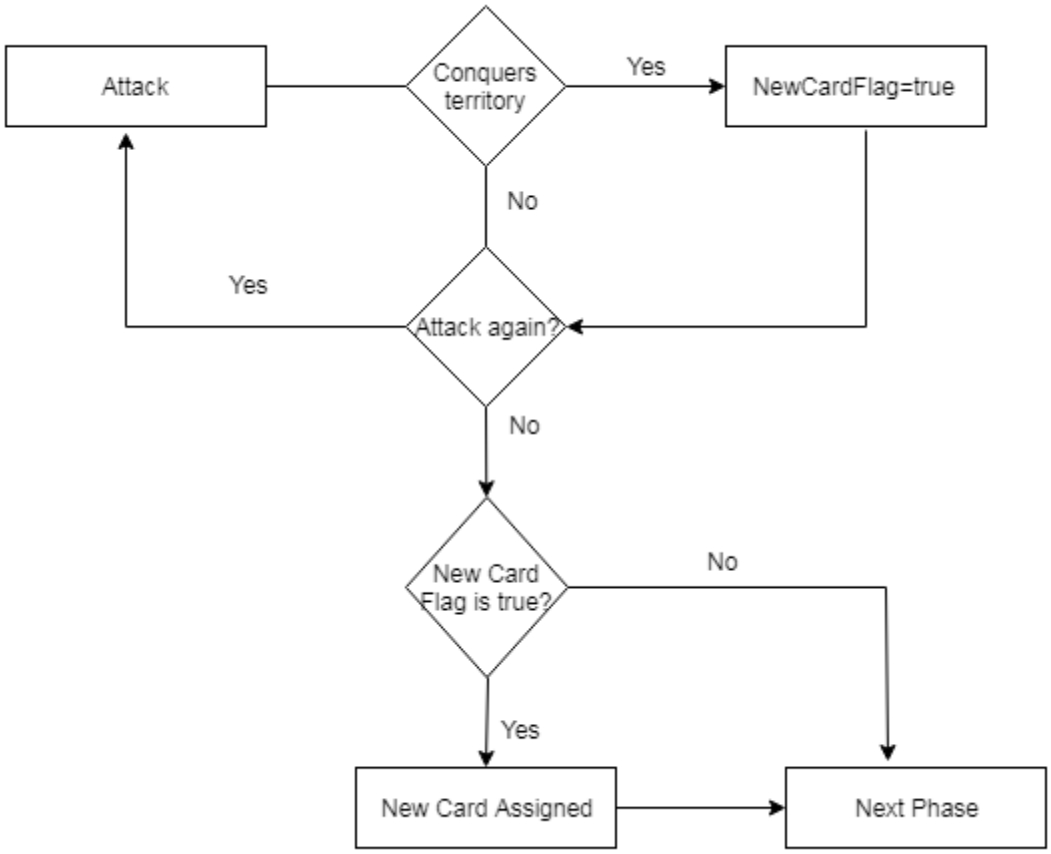
Build 2 – The objective of this build is to implement reinforcement , attack and fortification phases along with the card logic according to the risk rules. This build also implement the observer design pattern and involves the refactoring operations.

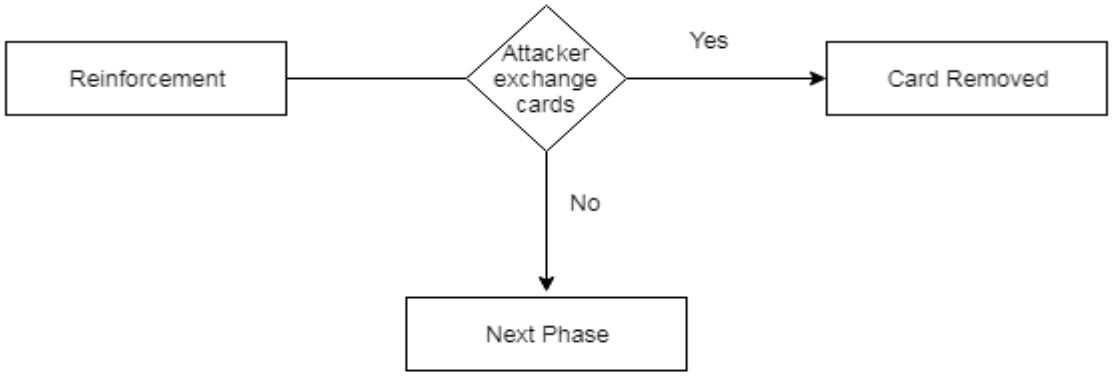
State Diagram Of Game



State Diagram Of Card Exchange View

A player receives a card at the end of attack turn if successfully conquered at least one territory during attack phase.





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 in round robin 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, attack phase begins. In this phase the player will select one of his owned territories to attack from. The player is given a list of adjacent countries to attack. The attacking player and the defending player roll the dice to simulate an attack. The attacker wins only when the defender has lost all of his defending armies. Whenever the territory is won, the attacker receives a card.
  - Once the attack phase is completed. 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.
  - Game ends when all the players are done with their phases.

- 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 in round robin 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, attack phase begins. In this phase the player will select one of his owned territories to attack from. The player is given a list of adjacent countries to attack. The attacking player and the

defending player roll the dice to simulate an attack. The attacker wins only when the defender has lost all of his defending armies. Whenever the territory is won, the attacker receives a card.

- Once the attack phase is completed. 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.
- Game ends when all the players are done with their phases.

MVC ARCHITECTURE

- Controller

RiskGameBuilder Controller
main( )

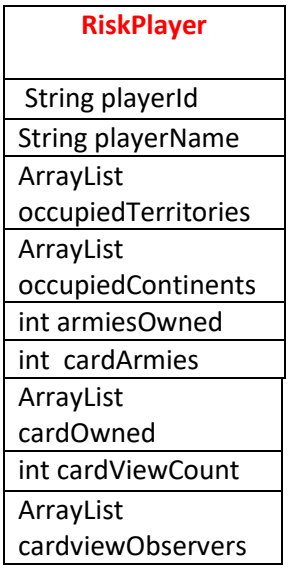
RiskPlayerBuilder Controller
setUpPlayers( )
getRiskPlayerList ( )
setRiskPlayerList ( )
getPlayersNameList( )
setPlayersNameList( )

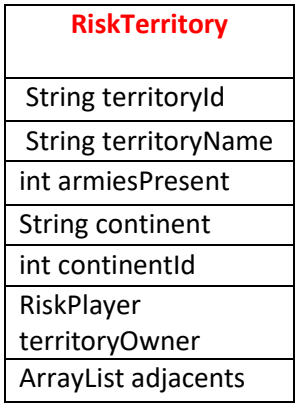
RiskMapBuilder Controller
loadMapData( )
parseMapFile( )
addContinents( )
addTerritories( )
addTerritoriesToContinents( )
buildAdjancencyMap( )
getTerritoryList( )
getContinentList( )
getAdjancencyList( )
getMapUploadStatus( )
setMapUploadStatus( )
getIdByTerritoryName( )

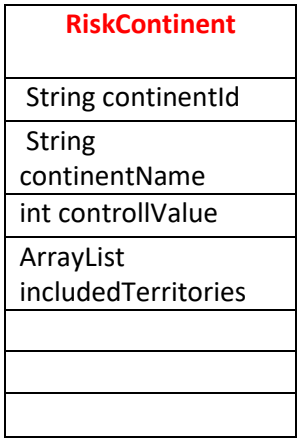
RiskReinforcement Phase Controller
calculateArmy ( )
CardExchangeView( )

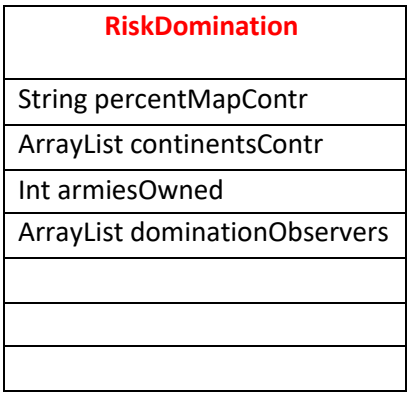
RiskAttackPhase Controller
attackPhaseInput( )
rollDiceForNormalAttackMode( )
rollDice( )
rollDice( )
rollDiceForAllOutAttackMode( )

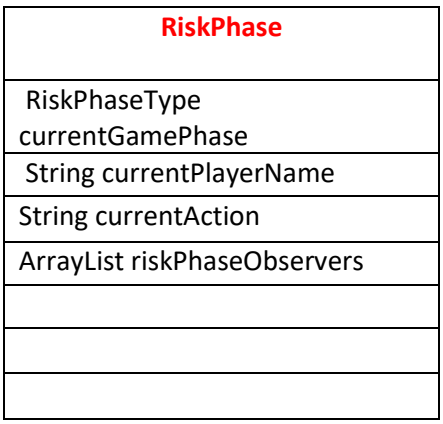
- Models

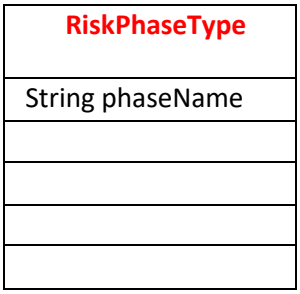












MAP VALIDATION FUNCTION

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 provides a platform to developers, users and businesses to develop their products quickly and efficiently. Eclipse provides a platform for several languages such as Java, C/C++, Javascript and many more.

4) Junit Framework

In Java, the standard unit testing framework is known as Junit. Junit helps to build a test suite, that will help to measure the progress of our project and spot the side effect so one can easily focus on the development process.