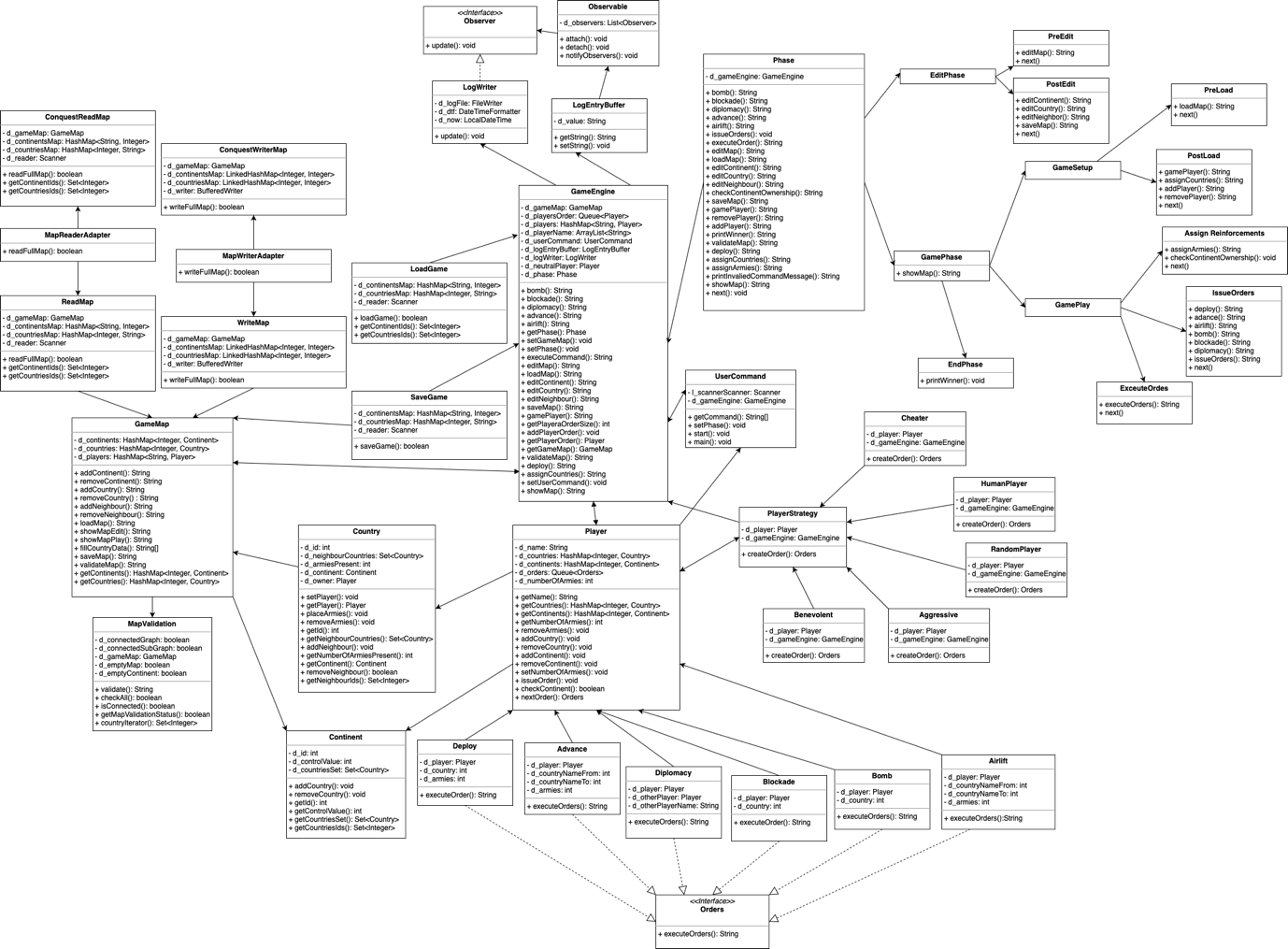
Architecture Design Document



*Description:*

**Class GameEngine**

This class handles most part of the game. It is controller of the game and all the operations will execute from this class.

**Class LogEntryBuffer**

This will inherits Observable class whose function is to obtain the outputs that are obtained after executing commands.

**Class LogWriter**

This class will inherits Observer and whose function is to write the output to the log file.

**Class Observable**

This Class will implements the connection/disconnection mechanism between observers and observables (subject). It also implements the notification mechanism that the observable will trigger when its state changes.

**Interface Observer**

Interface Observer, which forces all views to implement the update method.

**Class UserCommand**

This class will get user commands and execute in sequence.

**Class Phase**

An abstract class for the Phase implementation of state pattern.

**Class EditPhase**

The EditPhase class is inherited from the Phase class to support commands valid in editing phase.

**Class PostEdit**

The PostEdit class is inherited from the EditPhase class to support commands valid in post editing phase.

**Class PreEdit**

The PreEdit class is inherited from the EditPhase class to support commands valid in pre editing phase.

**Class GamePhase**

The GamePhase class is inherited from the Phase class to support commands valid in game phase(general commands).

**Class EndPhase**

This is the class that inherits GamePhase class whose fucntion is to print the winner of the game.

**Class AssignArmies**

AssignArmies class that inherits GamePlay class to support functions during this command.

**Class GamePlay**

GamePlay class that inherits GamePhase to support gameplay compatible commands.

**Class ExecuteOrders**

ExecuteOrder class that inherits GamePlay class to support functions during this command.

**Class IssueOrders**

IssueOrder class that inherits GamePlay class to support functions during this command.

**Class GameSetup**

Abstract class of game setup the represents the initial process of entering the gameplay.

**Class PostLoad**

This class will setup the game before game load.

**Class PreLoad**

This class will do necessary steps after game load.

**Class Country**

This class contain all details of country like ID, Continent, Neighbours, Owner, No. of Armies etc.

**Class Continent**

This class contain all details of country like IDs of Country , Control Value.

**Class Player**

This class contain all details of country like name of player, ID, dominated Countries and Continents, No. of Armies, Orders etc.

**Class GameMap**

This class will handle all the operations related to map like add/remove Countries and/or Continents, add/remove neighbours, load map, save map, validate map, show map.

**Class WriteMap**

This class will write game map as text in map file.

**Class ReadMap**

This class will read data from map file.

**Class ConquestReadMap**

This class will read data from conquest map file

**Class ConquestWriteMap**

This class will write game map as text in conquest map file.

**Class MapReaderAdapter**

Adapter class to translate domination map file into conquest file.

**Class MapWriterAdapter**

Adapter class to translate domination map file into conquest file.

**Class MapValidation**

This class will validate map which is created by user, according to rules of the game.

**Interface Orders**

This interface provide structure for execute order.

**Class Bomb**

This class will execute order given by user to place Bomb Card on specified country. This card will destroy 50% army of country.

**Class Blocakde**

This class will execute order given by user to place Blockade Card on specified country. This card will increase army by 3 times and assign owner as a Neutral Player.

**Class Airlift**

This class will execute order given by user to place Airlift Card on specified country. This card will transfer specified no. of amies from one country to other.

**Class Advance**

This class will execute Advance order given by user on specified country. This will transfer specified no. of amies from one country to other if player owns the country otherwise it will attack with specified armies.

**Class Diplomacy**

This class will execute order given by user to place Diplomacy Card on specified country. This card will be used to negotiate between two players so that neither can attack on each other during their turn.

**Class Deploy**

This class will deploy armies according to orders issued by the player.

**Class ShowMap**

This class will execute show map command at any time in the game.

**Class PlayerStrategy**

Abstract class that is implemented by concrete strategy classes. This is the class that declares the method to be implemented by the various strategies.

**Class Aggressive**

Concrete strategy class of Aggressive player. An aggressive computer player strategy that focuses on centralization of forces and then attack, i.e. it deploys on its strongest country,then always attack with its strongest country, then moves its armies in order to maximize aggregation of forces in one country.

**Class Benevolent**

Concrete strategy class of Benevolent player. A benevolent computer player strategy that focuses on protecting its weak countries (deploys on its weakest country, never attacks, then moves its armies in order to reinforce its weaker country).

**Class Cheater**

This class has the behavior of the Cheater Player. All the neighbor countries are acquired by this Cheater Player and armies are doubled in those neighbor countries that are recently conquered.

**Class HumanPlayer**

Concrete strategy class of Human player. A human computer player strategy that deploys, attacks, moves armies according to his need. This player also plays with all types of cards.

**Class RandomPlayer**

Concrete strategy class of Random player. A random computer player strategy that deploys on a random country, attacks random neighboring countries, and moves armies randomly between its countries.