Peer-Review 2

Martino Piaggi, Amrit Singh, Lorenzo Perini
Group GC23

9 maggio 2022

Peer Review UML diagram Eriantys Group GC33.

1 Positives

The choice of setting up a limited model in the player is the correct one, as sending the game objects over the connection sends a lot of useless data. The LobbyController class may be conceptually redundant, but could work as an interface to the different controllers. The server-side part for sending messages on connection seems a little bit confused and unnecessary, but we really like the usage of the factory method for messages. The sequence diagram is correct.

2 Negatives

Dividing the controller into various tasks is interesting, but we fear it may result in unnecessary complexity; separating them totally into different classes can generate difficulties since all phases are connected to each other and also the client has to know exactly which method of which class it has to call.

Also, there is some game-related logic implemented inside the controller, which is not appropriate: since those rules are part of the game, they should be implemented in the model.

3 Comparison between architectures

We have implemented the turn logic, the card generators, and tiles generator inside the model. In our project, the controller is just a middleman and doesn't compute game dynamics. The controller basically just knows what the parameters should be and asks the model if the move is legal.

Instead of the multiple controllers approach, you could use a single controller with the Command Pattern in order to facilitate the controller architecture.

The network design is very similar to our design. We just use different names, but the logic is basically the same. We suggest adding some lobby functionalities that we have already implemented and that are not present in yours, such as listing the available lobbies on the server, players in the lobby, and the ability to select expert mode.