Peer-Review 2: Sequence Diagram

<Riccardo Compagnoni Domingo>, <Emanuele Dossi>, <Eleonora Fidelia Chiefari>, <Michele D'Elia>

Gruppo <AM28>

Valutazione del sequence diagram del gruppo <AM37>.

Lati positivi

Abbiamo notato che *GameController* rimuove le tessere dalla plancia solo dopo aver verificato che l'intera mossa del giocatore è valida. Questo controllo garantisce che il modello resti consistente in tutti gli istanti temporali. In questo modo si evita una situazione in cui le tessere siano state rimosse, ma non inserite in nessuna *bookshelf*.

Lati negativi

1. Per quanto riguarda la fase di login, non riteniamo ottimale che il server invochi *GetGameController* prima della verifica dell'username, perché è necessario rieseguire il metodo ad ogni tentativo di login. Sarebbe meglio che il server invocasse un metodo *login(nickname)* su *MasterController* e che fosse quest'ultima classe a fare le invocazioni necessarie su *GameController* per stabilire se il login è valido o meno e poi rispondere al server con un messaggio di errore o ok. In seguito ad esito positivo, il server potrà eseguire il metodo getter.
2. Per quanto riguarda la fase di game, dal diagramma pare che tutte le informazioni del turno di un giocatore, tra cui le posizioni delle tessere, il loro ordine e la colonna selezionata, vengano mandate al server in un unico messaggio *positions*. Per questo motivo abbiamo dedotto che o il client ha una parte di logica sostanziosa o il giocatore scopre a fine turno che le proprie mosse non sono valide. Consigliamo di suddividere il messaggio in più parti, in modo che ognuna corrisponda a una determinata fase di gioco. Ciò ridurrebbe la logica applicativa contenuta nel client, permettendo comunque al giocatore di sapere subito se la mossa effettuata è andata a buon fine attraverso un messaggio di conferma dal server.

Confronto tra le architetture

Non è stato possibile realizzare una valutazione completa, poiché il materiale ricevuto non comprende il formato dei messaggi, né le interazioni client-server nei casi di eccezione (ad esempio quando la mossa non è valida o quando il nickname del messaggio non corrisponde al giocatore corrente). Questa parte costituisce una parte sostanziosa del protocollo di comunicazione, quindi il confronto è abbastanza limitato.
Abbiamo comunque notato una gestione differente nel comunicare la mossa da parte del client al server. Noi abbiamo infatti previsto tre messaggi distinti per ciascuna fase di gioco: la selezione delle tessere, il riordinamento e la selezione della colonna; l'implementazione in esame, invece, prevede un unico messaggio *positions*, che da una parte garantisce l'atomicità della comunicazione del turno al server, ma dall'altra comporta i lati negativi evidenziati sopra.