Peer-Review 2: Sequence Diagram

Abdallah Alkhetiar, Daniel Bonardi, Caterina Cardenas, Simone Ceccherini

AM17

Valutazione del Sequence Diagram del gruppo <AM 26>.

# Lati positivi

Nonostante ci siano alcuni punti in cui manchino alcuni lanci di eccezioni, il flusso e l’invio dei messaggi per la gestione del gioco sono praticamente completi, e coprono anche il caso di disconnessione degli utenti.  
A livello grafico, i sequence diagram sono stati realizzati molto bene, anche grazie alla suddivisione a colori tra i messaggi di successo (evidenziati in verde o blu) e quelli di fallimento o lanci di eccezioni (evidenziati in rosso). Anche l’utilizzo di riquadri per indicare le varie alternative a disposizione dell’utente ad un certo punto del flusso aiutano a individuare i vari rami dell’esecuzione, anche se a volte rischiano di diventare un po' confusionali.

# Lati negativi

**Login**

* Nel caso di creazione della lobby per il primo giocatore, manca il passaggio dell’informazione di numero di giocatori da server a controller. Inoltre, manca il controllo del caso in cui il numero di giocatori inserito sia invalido (n < 2 or n > 4).
* Nel caso in cui un giocatore entra nel gioco quando una lobby non è stata creata, ma non è comunque il primo giocatore, la funzione wait(), non sembra avere molta importanza. Basterebbe fare in odo che l’utente non possa compiere azioni fino a quando una lobby non viene creata.
* Nella conferma della creazione della lobby, il messaggio “Ok, answer” non è molto chiaro. Si tratta di un aknowledgement message generico e che quindi richiede una specifica di quale richiesta sta confermando o è un messaggio specifico per la creazione di una lobby?
* I messaggi dal server di creazione della lobby e invio della lista di colori disponibili, potrebbero essere unificati anche per evitare errori nel caso vengano utilizzati thread che invialo in modo parallelo messaggi dal server.
* Nella selezione del colore, non è molto chiaro da dove salti fuori il nickname del giocatore. Se non è l’utente stesso a inviarlo (anche implicitamente) come ha fatto il controller a mandarlo al model? Invece cosa succede se l’utente sceglie un colore che non esiste? (Questo è un problema che può presentarsi solo da TUI, mentre in GUI può essere evitato limitando la scelta dell’utente con dei bottoni)

**Game start**

* Nella descrizione non viene citato il fatto che ad ogni giocatore viene passata la propria carta iniziale, gli obbiettivi comuni e le carte nella propria mano, ma non viene menzionato il fatto che dovrebbero essere passate anche le 4 carte visibili (e comuni) a tutti e la carta coperta in cima ad ogni mazzo.
* Ancora una volta, quando vengono inviati più messaggi di fila dal server al client sarebbe meglio unificarli in un unico messaggio.

**Game flow**

* Non sembra esserci nessun lancio di eccezione nel caso un giocatore compia un azione in un turno al di fuori del proprio. E nemmeno nel caso in cui un giocatore decida di pescare una carta prima ancora di averne posizionata una.

**Game end**

* In questo caso sembra che i lanci di eccezioni in caso di fallimento siano stati lasciati sottintesi.
* Di nuovo, si potrebbero unificare i messaggi di fila lanciati dal server.

**Connection lost**

* Non si capisce in che modo si controlla l’avvenimento della disconnessione. C’è un messaggio che viene inviato periodicamente dal client e se il server non ne riceve più di una certa quantità di fila, marca il giocatore come disconnesso? Oppure esiste un messaggio unico che viene inviato dal client al server il momento prima della disconnessione?
* Nel caso di disconnessione durante una partita non è molto chiaro cosa succede al giocatore, cioè in che senso il server decide a random quello che succede allo stato del giocatore?

Per fare un riassunto generale dei problemi più rilevanti, per prima cosa riteniamo sia poco chiaro cosa permetta la ricezione e le chiamate a metodi tra client e server, infatti nel diagramma non è presente alcuna distinzione tra socket e RMI. Inoltre, non è stato modellizzato quello che accade lato client. Viene tutto interpretato da un'unica classe che si occupa unicamente dell’invio e della ricezione dei messaggi?  
Inoltre è poco chiaro se i due diversi tipi di connessione sono gestiti tramite l’invio di messaggi sottoforma di oggetti contenenti i parametri necessari, sfruttando almeno in parte le potenzialità della programmazione ad oggetti, oppure il gruppo ha deciso di optare per un protocollo di comunicazione più simile al funzionamento di RMI inviando per intero i parametri richiesti dalla funzione “chiamata” sul server.

# Confronto tra le architetture

Avendo scelto FA diverse i due sequence diagram sono simili solo nei punti comuni ad entrambi i gruppi come, ad esempio, il flusso di un gioco e la sua conclusione. Per quanto riguarda la parte di disconnessioni, non essendo una funzionalità implementata nel nostro progetto non possiamo fare un paragone. In fin dei conti non ci sono particolari modifiche che apporteremmo al nostro sequence diagram.