Peer review - network

Aspetti positivi

Buona l'idea di utilizzare un messaggio di HeartBeat con il settaggio di un timer per verificare le connessioni da parte dei diversi client.

Troviamo ottima l'idea di effettuare i doppi controlli, sia lato client per gestire subito gli errori client-side che lato server per gestire messaggi non validi e /o client malevoli. Inoltre questo permette di eseguire controlli più stringenti lato server avendo esso accesso a tutti i dati della partita.

La gestione delle 3 fasi di gioco in maniera separata con una macchina a stati dedicata per ognuno è una funzione ottimale per risolvere il problema di gestire diversi scenari e permettere di fare azioni coerenti da parte del client per lo stato in cui si trova. Nel nostro progetto abbiamo deciso di approcciare questa problematica realizzando un MVC *ad hoc* per la gestione della connessione e creazione della Lobby, per poi svincolare le partite una volta iniziate e trasferirla allo schema MVC relativo allo svolgimento del gioco, nonostante ciò anche noi realizziamo una sorta di macchina di stati in funzione della scelta precedente.

Aspetti negativi

Non è chiaro se il client sia observer/listener della view e viceversa e, dualmente sul server, se la classe server sia observer/listener del model con il controller listener/observer del server (sappiamo che non riguarda propriamente la rete, argomento di questa peer review, ma forse renderebbe più chiaro il funzionamento).

Vorremmo inoltre capire come avete gestito la comunicazione col Server di client con tipologia di rete differente (RMI o Socket) affinchè il Server da lato suo comunichi con essi senza dover fare sempre uno switch case in base alla connessione e gestisca tutto in maniera piu astratta senza dover duplicare il codice.

Per quanto riguarda il metodo execute che viene overridato in base alla ricezione di un messaggio di tipo MessageToServer o MessageToClient non ci è ben chiaro il suo funzionamento, sarebbe stato gradito un eventuale Sequence Diagram che mostrasse la problematica che riesce a risolvere in maniera elegante; ci chiedevamo se uno switch sia così svantaggioso rispetto alla creazione di molte classi.

La decisione di realizzare un MVC della Lobby solo per la funzionalità multi partita va bene ma probabilmente la si può realizzare anche solo per realizzare più controller modulari molto più chiari e leggibili, anche se si giocasse una partita sola.

Ci piacerebbe avere un confronto per comprendere come riuscireste a compattare le 12 commonGoalCard tutte quante in sole 3 classi.

In generale, seppur presenti nel diagramma UML, qualche commento sul funzionamento delle varie classi avrebbe aiutato molto a capire come funziona nella pratica la rete.