Documentazione Peer Review #2

Documentazione protocollo di rete gruppo AM27

<Giuliano Crescimbeni, Mauro Leporace, Luca De Nicola, Cristian Dheskali>

L'implementazione descritta supporta entrambe le tecnologie di rete (Socket e RMI) e la funzionalità aggiuntiva ***Partite Multiple***

Il server è composto da 5 componenti per la gestione delle connessioni e delle partite:

• **ServerApp**: Si preoccupa di gestire le nuove connessioni in entrata, per ognuna di esse genera un **ClientHandler** su un nuovo thread

• **ClientHandler:** Classe addetta allo scambio di messaggi sulla rete, presenta due funzioni per lo scambio di messaggi, una per la ricezione e l'altra per l'invio. I messaggi in entrata vengono inoltrati al **GamesManager**

• **GamesManager:** Questa è la classe principale per le associazioni connessione->idPartita, viene istanziata una sola volta dalla **ServerApp.** Per ogni messaggio in entrata, tramite la funzione handleMessage(), il GamesManager ottiene il gameId del client ed il **controller** della partita, così da poter eseguire le azioni sul controller corretto

• **Controller**

• **Model**

Messaggistica

Lo scambio di messaggi avviene tramite la serializzazione di una classe chiamata **Command.** Le sottoclassi di Command rappresentano ognuna un azione distinta del client. Tutte le sottoclassi fanno override di un metodo Execute() per l'esecuzione del comando. La funzione richiede il passaggio di un Controller, che gli verrà fornito dal GamesManager.

Gli Update ai client vengono attuati tramite una classe **Update**, il loro funzionamento è simile a quello della classe Command perciò evitiamo la sua descrizione

Riportiamo qui sotto 3 casistiche di interazioni client-server: