Peer-Review 2: Network UML

Mario Pesca, Salvatore Tuand, Lorenzo Simone, Xuwen Ye
Gruppo 24

10 maggio 2024

Valutazione del diagramma UML delle classi di Network del **Gruppo 14**.

1 Lati positivi

- Adeguato l'utilizzo del Command Pattern, in quanto implementa in modo corretto il comportamento desiderato che si vuole instaurare tra Client e Server, così come prendere decisioni anche a runtime.
- Adeguata l'implementazione del Controller in modo da permettere la scelta da parte del Client di connettersi via RMI o via Socket.
- L'utilizzo dell'interfaccia Listener permette di aggiornare correttamente le classi del model nel client.

2 Lati negativi

- Non viene specificato un modo in cui si controlla se un Client si sia disconnesso o no.
- Non viene specificato il modo in cui si notifica ciascun Client se una partita inizia, se finisce o se un giocatore viene disconnesso.
- In generale, l'UML e la relativa documentazione forniscono soltanto una overview molto generica sull'implementazione del protocollo di rete, senza focalizzarsi molto su casi specifici o limite. Dei sequence diagrams sarebbero potuti essere utili ai fini della comprensione della gestione di questi eventi.
- Sarebbe utile avere un attributo in User (magari un enumeration) per rappresentare lo stato del client in maniera più esplicita (e.g. "InLobby", "Logged", "Disconnected", "Playing").
- Probabilmente la distinzione di LobbyServer e MatchServer in due classi separate non porta un vantaggio tale da compensare il dover dividere anche i controller e quindi le rispettive code (cosa realizzabile anche con i Thread).

3 Confronto tra le architetture

- Entrambe le architetture sono state sviluppate in modo da consentire sia una connessione Socket che una connessione RMI.
- Anche nel vostro caso RMI è stato reso "asincrono" mediante i comandi e relative code. Nella nostra architettura però è stato aggiunto un livello di astrazione in più rendendo qualsiasi messaggio tra client-server un "pacchetto", dove ciò che cambia tra client e server è solo nel come si gestisce un determinato tipo di pacchetto.
- Nel nostro caso la differenziazione tra LobbyServer e MatchServer avviene in un singolo attributo della classe ServerNetworkHandler.