Peer-Review 2: Network

Alessandro Romito, Gioele Crespi, Matteo Roman
Gruppo GC08

9 maggio 2022

Valutazione del diagramma UML della parte network del gruppo GC18.

1 Lati positivi

- Ci sembra giusta la scelta di creare una classe SocketClientConnection che si occupa di ascoltare i messaggi del client e associare a quest'ultimo una virtual view.
- Architettura semplice e modulare in quanto le classi sono poche ed efficaci.
- Rispetta tutti gli obiettivi della parte network.

2 Lati negativi

- La descrizione è molto complicata da capire, come se le idee fossero poco chiare.
- Manca un metodo per verificare lo stato di connessione dell'utente (tipo ping).
- Invece di passare dei messaggi tramite stringhe, conviene differenziare il tipo di messaggio utilizzando per esempio classi Messaggio1, Messaggio2 etc., magari gestendo il tipo di messaggio tramite un'enumerazione.

- Passare il modello serializzandolo e deserializzandolo per ogni view per aggiornare ogni singolo cambiamento sembra eccessivo e dato che il model non è leggero potrebbe incidere a livello di computazione e di connessione, meglio optare per una soluzione più leggera.

RemoteView

- La RemoteView non dovrebbe occuparsi dello scambio dei messaggi tra client e server: MessageReceiver dovrebbe essere una classe a parte.

Non abbiamo rilevato problemi critici, ci è piaciuto l'approccio con cui vi siete mossi.

3 Confronto tra le architetture

Il nostro UML è più espansivo in quanto le classi sono più numerose ma hanno compiti più specifici.

Da un punto di vista il meccanismo di scambio dei messaggi è simile al nostro: quando arriva un messaggio, la SocketClientConnection notifica la MessageReceiver che chiama la View, che a sua volta notifica il Controller (che esegue la mossa sul Model). Da un altro punto di vista il vostro programma invia messaggi di stringhe, mentre il nostro ha tanti tipi di classi messaggio diverse.

La grossa differenza tra i nostri UML consiste, come specificato precedentemente, nel passaggio del model tramite serializzazione e deserializzazione. Nel nostro caso verranno inviate solo le modifiche al model, quindi la view a lato client si occuperà soltanto di aggiornare la visualizzazione, senza influire negativamente sui canali di scambio dati.