Peer-Review 2: Protocollo di comunicazione

Claudio Arione, Riccardo Begliomini, Giuseppe Boccia

Gruppo AM17

Valutazione del protocollo di comunicazione del gruppo AM47.

Lati positivi

Dopo aver analizzato con attenzione il documento fornitoci dal gruppo revisionato, abbiamo evidenziato i seguenti aspetti positivi:

- gestione delle partite multiple: abbiamo apprezzato il modo in cui vengono gestite le partite multiple, tramite *Lobby*, *Match* e le *ConcurrentHashMap*, che permettono una gestione thread-safe delle connessioni con i client;
- classe *State*: l'astrazione implementata dalla classe *State* del controllo sulla validità dell'azione che l'utente tenta di eseguire. In particolare, poiché le carte personaggio possono essere giocate in qualunque momento, il controllo su *CharacterCardCommand* che ha sempre esito positivo;
- distinzione tra *VisitorCommand* e *PlayerVisitorCommand*: questa distinzione tra i comandi che provengono da un giocatore e quelli che vengono impartiti alla partita in modo automatico rispecchia la nostra idea di aggiornamento di stato della partita e l'evoluzione concettuale del gioco stesso.

Lati negativi

Abbiamo osservato alcuni aspetti del progetto che a nostro parere possono essere migliorati:

- poca chiarezza nella definizione dei messaggi: nel file che ci è stato fornito non è presente alcuna specifica riguardo la struttura dei messaggi scambiati. Nonostante sia riportato che i messaggi sono in formato JSON, non sono indicati né i campi né i corrispettivi valori che possono assumere.
- stack di chiamate vicendevoli tra *State* e *Command*: il metodo che aggiorna lo stato della partita invoca un metodo del *Game* che a sua volta invoca un metodo del *Command*, poi uno di *Board* e infine di nuovo un metodo di *Command*. Non ci è chiaro dal documento fornito il motivo di questo zig-zag di invocazioni.

Confronto tra le architetture

Entrambi i gruppi hanno scelto di serializzare i messaggi in formato JSON, tuttavia, sono state prese decisioni differenti sia per la gestione dello stato della partita che per la gestione stessa dei messaggi. Il gruppo revisionato, infatti, ha optato per un metodo del *Game* che accetta un messaggio e modifica direttamente lo stato della partita, se il comando è consentito. Un comando non consentito viene quindi intercettato e rifiutato dal controller. Nella nostra implementazione, invece, l'azione viene inoltrata al model e, se non consentita, viene sollevata un'eccezione e mandato il relativo messaggio di errore al client.