Peer-Review 2: UML

Giovanni Stefanizzi, Giacomo Scortecci, Roberto Villa

Gruppo GC57

Valutazione del diagramma UML delle classi del gruppo GC 02.

# Lati positivi

* Per gestire la comunicazione tra Client e Server è stato deciso di realizzare degli oggetti di tipo Message che vengono serializzati in StreamObject e successivamente scambiati.
* I vari tipi di messaggi sono chiari e la descrizione inviata permette di comprendere facilmente la gestione delle varie fasi della partita, in particolare la fase di setup del gioco è spiegata molto dettagliatamente
* Sono state definite delle enumerazioni nel Model che permettono di capire in che fase di gioco ci si trova, questa soluzione permette di trovare errori con più semplicità e permette una gestione ordinata dei round della partita.
* Il Controller è stato pensato per poter gestire le funzionalità aggiuntive “partite multiple” e “Partita a 4 giocatori”, permettendo un setup della partita completo.

# Lati negativi

* Non è ben specificato come avvenga, nel messaggio UseCharacter la gestione dei diversi parametri che sono necessari per giocare carte differenti.
* L’utilizzo del pattern observer/observable non è strettamente necessario per quanto riguarda la comunicazione tra RemoteView e controller in quanto la prima necessita di agire su un unico oggetto ed è quindi possibile andare a chiamare direttamente il controller.

# Confronto tra le architetture

* Per quanto la gestione del gioco sia simile avviene in modi differenti, in particolare la gestione delle differenti fasi del turno e della partita avviene nel nostro caso a livello di controller e non di model.
* Anche nel nostro caso si è scelto l’uso di un interfaccia di tipo view che viene implementata sia dalla CLI che dalla GUI.
* Molto probabilmente implementeremo anche noi la fase di setup in modo simile a quanto descritto dal gruppo, vista la sua completezza e semplicità