Peer-Review 1: UML

Massimiliano Botta, Riccardo Antonelli, Alessandro Assini, Enrico Alberti

Gruppo AM03

Valutazione del diagramma UML delle classi del gruppo AM12.

Lati positivi

1) Obiettivi Comuni e Personali:
    a) Interessante la gestione dei CommonObjectives e dei PrivateObjectives attraverso uno shuffle iniziale da cui vengono presi il numero di obiettivi desiderati rispettivamente.
    b) Sempre in riferimento ai CommonObjectives è ottima la separazione del completamento degli obiettivi ed il calcolo dei punti, in modo che l'aggiunta dei punti al giocatore sia indipendente dai vari algoritmi di verifica degli obiettivi.
2) Tile:
    a) Buona gestione delle carte, specialmente la carta Empty per gestire exceptions e casi particolari.

Lati negativi

1) Board:
    a) L'istanziazione di 3 diverse classi board per lo sviluppo della living room in base al numero dei giocatori è un po' pedante (overkill) poiché le tre classi non differiscono in nulla se non nelle coordinate utilizzabili.
2) Obiettivi Comuni e Personali:
    a) Non è molto chiaro come verrebbe utilizzata una "stack" per l'attributo "point" in commonObjectives.
    b) Per i private objectives sono da 2 a 4 e non 12 gli obbiettivi personali da istanziare.
3) Game e Controller:
    a) In generale non è chiaro come il flusso di gioco sia gestito, sembra sia fatto tutto in handleTurn() nel controller, potrebbe essere utile separarlo in diversi metodi.
    b) Non è chiaro come venga gestita la parte iniziale di gioco (aggiunta dei giocatori).
4) Architettura:
    a) A nostro parere dovreste delegare le operazioni a più classi di 'supporto', sia nel model ma specialmente nel controller dove solo 2 metodi gestiscono tutte le operazioni. Inoltre, non è esattamente chiaro come è gestito lo stato di gioco, ad esempio se si tratta dell'ultimo turno della partita.

Confronto tra le architetture

La nostra architettura implementa un thick client. Il model è nel server e gestisce le chiamate fatte dal controller unico per ogni client. Mentre quello del gruppo AM12 un thin client dove un controller gestisce il model.

Anche noi gestiamo diverse "board" (nel nostro caso "living room"), ma non attraverso 3 classi distinte.

Pensiamo sia giusto tenere i metodi del Game, pensando che essi debbano rispondere a tutte le possibili necessità che il controller possa avere in termini di controllo dello stato e risposta agli input degli utenti in arrivo dalla view.

Nella sezione di obiettivi comuni noi abbiamo trattato anche l'ordine dei giocatori che completano gli obiettivi comuni da cui dipende il punteggio che gli viene assegnato. Inoltre, la randomizzazione degli obiettivi scelti, sia personali che comuni, viene effettuata all'interno dell'inizializzazione e non tramite un metodo "shuffle".