Peer-Review 1: UML

Alessandro Romito, Gioele Crespi, Matteo Roman
Gruppo GC08

4 aprile 2022

Valutazione del diagramma UML delle classi del gruppo GC18.

1 Lati positivi

- Buona suddivisione delle classi. L'UML grazie alla descrizione ci è risultato semplice da capire.
- Molto intuitiva la classe "Parameters" quanto veloce l'implementazione e la facilità di utilizzo per gestire l'inizializzazione e gli eventi di gioco.
- Lo sviluppo delle carte personaggio è simile a come è stato portato avanti nel nostro gruppo, in base alle nostre discussioni crediamo sia la soluzione migliore.
- Le isole sono state implementate chiaramente. Evidenziamo soprattutto sul calcolo dell'influenza e fusione delle isole (gestione degli studenti, attributo islandRank, etc...).

2 Lati negativi

Game

- Non abbiamo capito come si setta il current player e quale classe lo gestisce durante il turno, ipotizziamo Game ma non capiamo come con i metodi dati, riusciate a gestire l'ordine delle azioni del turno.

- Manca la fase iniziale e finale del gioco, come da voi specificato.

SchoolBoard

- Non capiamo il motivo della separazione delle classi SchoolBoard e DiningRoom.
- Presupponiamo uno spreco di memoria e svantaggio per un'eventuale autosave() o analisi per i valori int di ritorno ai metodi di DiningRoom: secondo il nostro parere sarebbe meglio utilizzare un array di int invece che di Student nella DiningRoom.

CharacterDeck

- Ridondanza tra classe CharacterDeck e GameBoard per l'attributo characterCards, secondo il nostro parere la classe CharacterDeck è inutile (è più conveniente un attributo solo dentro gameboard).

Expert mode

- Come riferitoci, la expert mode è in fase in fase di implementazione.

I lati negativi, più che lati negativi sono spunti o consigli. Non abbiamo rilevato problemi critici in quanto, secondo noi, la parte di design è stata affrontata con il giusto approccio.

3 Confronto tra le architetture

Il vostro UML è molto simile a quello del nostro gruppo, soprattutto per quanto riguarda la logica delle classi e dei metodi implementati. Rispetto al nostro UML è più espansivo e richiede meno memoria, di conseguenza è più "snello" sia a livello computazionale che per quanto riguarda la chiarezza nel leggerlo e capirlo da un lettore. Il nostro gruppo non ha usato una classe apposita "Parameters" a cui riferirsi per capire le dinamiche di gioco ma abbiamo implementato il pattern factory method su SchoolBoard, che viene applicato per costruire l'ambiente di gioco durante l'inizializzazione. Prenderemo spunto da come avete implementato le CharacterCards per migliorare la nostra architettura.

Per quanto riguarda la expert mode, ci è stato riferito che verrà implementata tramite if. La differenza sostanziale ripetto alla nostra implementazione è il fatto di aver usato una classe apposita "ExpertGame" che estende la versione normale.