Peer-Review 1: UML

Chiara Angileri, Nicholas Beltramini, Stefano Arcaro

Gruppo AM06

Valutazione del diagramma UML delle classi del gruppo AM36

1 Lati positivi

- La gestione delle fasi è molto esplicativa e determina una facile suddivisione del turno oltre che dell'inizio e termine del gioco. Inoltre, il susseguirsi delle fasi attraverso l'utilizzo dell'interfaccia UserEvent rende la soluzione molto elegante.

- Abbiamo apprezzato l'utilizzo della classi IslandList e PlayerList all'interno della classe Game che permette di mantenere quest'ultima più 'pulita' dal momento che la gestione dell'iterazione, dell'aggiunta e della rimozione degli elementi viene effettuata all'interno delle relative classi.

- Utilizzo estensivo di interfacce, alcune delle quali vengono passate come parametri astraendo in tal modo dall'implementazione.

- Utilizzo di pattern quali factory e decorator per gestire efficacemente il calcolo dell'influenza, la creazione di game in base al numero di giocatori e alla modalità scelta, e la creazione dei personaggi.

2 Lati negativi

- Gestione delle isole e di madre natura: crediamo che potrebbe essere utile per il movimento di madre natura iterare sui gruppi di isole piuttosto che sulle singole isole (evitando quindi di fare un controllo durante l'iterazione per individuare l'appartenenza o meno di un'isola ad un determinato gruppo).

- Gestione dei personaggi: vi consigliamo di ridurre il numero delle classi raggruppando tra loro i personaggi con caratteristiche e/o effetti simili. Inoltre, potrebbe essere utile assegnare a ciascun personaggio un ID univoco.

- Potrebbe avere senso raggruppare le classi *Entrance* e *Hall* in una classe contenitore (per esempio *Board*) alla quale farebbe riferimento il player: questo perché ad un livello concettuale il giocatore possiede una plancia, appunto composta da un'entrata, una sala e l'insieme di torri, che quindi potrebbe essere tolto dalla classe *Player* e messo nella classe contenitore.

- Si potrebbero evitare le classi *BlockCard* e *Tower* dal momento che la gestione delle torri può essere effettuata attraverso un intero associato ad ogni giocatore e un attributo di tipo *TowerColor* per le singole isole oltre che per i giocatori; lo stesso discorso vale per le carte divieto che possono essere rappresentate da un intero sia per il personaggio ad esse relativo, sia all'interno delle isole.

3 Confronto tra le architetture

Rispetto alla nostra architettura crediamo che i seguenti aspetti siano stati gestiti in maniera migliore:

- Fasi di gioco: sostituiremo i metodi che abbiamo adibito alla gestione delle fasi della partita in più classi, ciascuna delle quali rappresenta le singole fasi del turno.

- Utilizzo del pattern factory per la creazione del game: avremmo intenzione di integrare questa scelta nella nostra architettura utilizzando come discriminante la modalità di gioco piuttosto che il numero di giocatori.

- L'utilizzo delle interfacce *studentMoveSource* e *studentMoveDestination* per uniformare lo spostamento degli studenti.