COMMENTO UML INIZIALE

Gruppo 6 - Prof. San Pietro

Lorenzo Moretti
Matteo Mormile
Antonino Orofino

Di seguito vogliamo spiegare le scelte che ci hanno portato alla realizzazione del nostro design dell'aspetto model del gioco. In questo momento non abbiamo pensato di implementare la partita con quattro giocatori.

Innanzitutto la linea guida che abbiamo seguito maggiormente è stata quella di utilizzare le classi del model come sola rappresentazione dello stato dei vari componenti (personaggi, carte, il gioco stesso) a discapito della logica di interazione che abbiamo posto nelle classi del controller.

Abbiamo deciso di utilizzare la classe enum **Piece** per rappresentare una singola pedina a prescindere dal fatto che sia uno studente oppure un insegnante. Questa scelta ci ha permesso di ottimizzare tutte le altre classi che hanno interazioni con questi tue tipi di oggetto. Ad esempio, al posto di utilizzare un array contente tutte le pedine, abbiamo ritenuto più comodo utilizzare un oggetto che implementi Map, come HashMap, dove le chiavi sono rappresentate del valore enumerativo di Piece, mentre i valori sono di tipo intero (il numero di studenti per quel tipo).

Il gioco è rappresentato dalla classe **GameModel** che contiene gli elementi principali ed essenziali per descrivere un'intera partita. Tra questi componenti ci sono:

- **PlayerHandler** classe che si occupa di gestire i giocatori della partita, fornendo metodi per impostare il player corrente e restituire il player successivo in base all'ordine di connessione al server ed in base al numero sulla propria carta assistente giocata;
- **IslandsHandler** si occupa invece di gestire le isole e la posizione della pedina di Madre Natura nella plancia di gioco (si veda più avanti per capire meglio l'interazione con le isole);
- **TeachersHandler** classe che memorizza quale player possiede ciascun insegnate. Abbiamo deciso di inserire nella classe GameModel i professori, invece che nel singolo player, per poter sapere se un professore è già stato piazzato in maniera più veloce (non abbiamo infatti bisogno di iterare su tutti i player);
- **Bag** classe che rappresenta il sacchetto dal quale gli studenti sono pescati e fornisce metodi per restituire uno studente random dall'interno.

Il costruttore di **GameModel** si occupa di effettuare il setup iniziale in accordo alla modalità di gioco che gli viene passata come argomento.

La classe **Player** racchiude tutte le informazione di un singolo giocatore come il colore delle torri, le carte assistenti, il numero di monete e, se presente, la carta personaggio attiva. Il player inoltre possiede una **Board** che contiene la suddivisione degli studenti tra l'entrata e la sala da pranzo.

Per quanto riguardo le carte personaggio abbiamo creato la classe **Character** che conserva il nome, la descrizione ed altri dati utili per queste carte (come ad esempio il numero di NoEntry Flags e gli studenti che possono essere posizionati sopra la carta stessa). Questa classe inoltre contiene un attributo di tipo Rules (presente nel controller) che rappresenta come la logica del gioco viene cambiata quando la carta viene attivata.

L'ultimo aspetto riguarda le isole e loro gestione a seguito di unioni. La classe **Island** rappresenta un isola ed IslandsHandler ne gestisce le interazioni. Ogni isola possiede l'attributo *dim* che ne indica la dimensione: ognuna di esse ha una dimensione iniziale pari ad uno che può aumentare a seguito di unioni. IslandsHandler si occupa proprio di verificare se le condizioni che portano all'unificazione di più isole siano soddisfatte e procede all'unione in caso di esito positivo.