Peer-Review 1: UML

Ignazio Neto Dell'Acqua, Fernando Morea, Lorenzo Morelli

Gruppo 28

Valutazione del diagramma UML delle classi del gruppo 27.

Lati positivi

Il Gruppo 27 si presenta con un UML piuttosto avanzato.
Il package Model non mostra particolari criticità, anzi si presenta con alcune buone (se non ottime) caratteristiche di base:

- Studenti gestiti non come classe: array di color nell' ingresso della school e nelle nuvole, array di interi in isole e dining room (elemento che potrebbe semplificare per il controller il calcolo dell'influenza);
- Presenza di un bag per tenere traccia degli studenti rimanenti;
- Isole caratterizzate da molteplicità (si può notare nel controller la presenza di una funzione di merge che andrà a unificare due isole in una, a mio parere un buon approccio) raggruppate in un array di dimensione variabile presente in game Model (Island [] con presenza di Island count);
- Professori gestiti come classe globale (associazione professore-player, anche questo può semplificare in calcolo dell'influenza);
- Assistent Card gestiti come record (scelta furba dato che le carte sono fisse e non variano da partita in partita);
- Madre Natura gestita come semplice istanza di game Model indicante in quale posizione di Island[] si trova (approccio semplice e facile da gestire);
- Caratterizzazione dinamica dei player a seconda del tipo di modalità scelta (Expert, 2/3/4 giocatori…) a cui si aggiunge un uso corretto del pattern DECORATOR.

Gruppo 27 si è inoltre già portato avanti mettendo a punto quello che potrebbe essere un package primordiale per il Controller e per la View del sistema:

- Per quanto riguarda la view, nulla di particolare, giusto una differenziazione tra quello che dovrà rientrare a far parte della CLI e ciò che caratterizzerà la GUI (buona idea!);
- Per quanto riguarda il controller invece si ha un abbozzo più sofisticato e avanzato di tutte le classi che lo caratterizzano. Al riguardo è possibile segnalare alcuni buoni spunti:
    - Suddivisione del Controller nell'arco temporale del turno di gioco (un controller per ogni sotto-fase di setup turno, plan , action, ecc. …) ;
    - Suddivisione dei vari controlli in base alla sotto-fase in cui avvengono le varie funzionalità ;

    ➔ Questo approccio gerarchico delinea una linea molto chiara e ordinata del package: OTTIMA idea.

Lati negativi

Non da segnalare numerose criticità:

- Utilizzo degli studenti sia come array di color e sia come array di interi non chiaro e non uniforme ;
- Utilizzo dei pattern un po' forzato e non sempre chiaro (al posto di un decorator in player per esempio si potrebbe usare un factory method) ;
- Non facile leggibilità dello schema -> consigliamo maggiore ordine e chiarezza autoesplicativa delle classi/metodi/attributi ;
- Non si capisce utilità Player Simple.

Confronto tra le architetture

Il Gruppo 27 come già detto si presenta con un buon UML :

- Punti di forza: semplicità di alcune funzionalità, completezza (non manca nulla di particolarmente rilevante), buona suddivisione delle classi;
- Diversi sono gli spunti che questo modello ci ha fornito per migliorare il nostro progetto: implementazione di un Bag che tiene memoria degli studenti rimasti , utilizzo di Madre Natura come indice di posizione (piuttosto che un boolean dentro ogni isola), e anche un idea di base su come sviluppare il controller in futuro .