Peer-Review 1: UML

Davide Tenedini, Tommaso Sprocati, Federico Toschi
Gruppo 17

2 aprile 2022

Valutazione del diagramma UML delle classi del gruppo 27.

1 Lati positivi

- Buon uso dei paradigmi Object-Oriented
- Pattern factory per la creazione delle carte personaggio
- Cimitero delle carte per vedere quelle già giocate
- Board<T> buona idea come "contenitore" di pedine, ma sarebbe da cambiare nome e "Pov" (Metodo moveInPawn(pawn, src) poco intuitivo, forse meglio moveTO...(..., dest))

2 Lati negativi

- Logica di gioco nel model (package "Round" e classe Game)
- Tipo parametrico in "Pawn" per il colore, ma assegnato a due tipi di colore diversi (T -> Color/TowerColor)
- Nomenclatura poco chiara di classi/metodi
- Poco incapsulamento (Tutti i metodi sono pubblici) e packaging da riorganizzare

- Pattern carte e deck richiedono sicuramente casting (e quindi instanceof o simili), classe astratta "Card" vuota ed ogni carta ha i suoi metodi specifici (Non c'è un "card.activate()" generico ereditato da tutte)
- Mancanza di una classe "interfaccia" del model (i.e. Game), il controller dovrà effettuare chiamate a metodi più intricate (i.e. Game.getBoard().getSchool()... piuttosto che Game.addToSchoolOf(...))
- Poca costanza nella progettazione di entità (i.e. per la classe Board<T>, Island e Cloud la ereditano, ma compone sia School che Island, o anche nella gestione dei coin)

3 Confronto tra le architetture

- Design molto più Object-Oriented (in maniera più rigida)
  - Mothernature ha una sua classe, così come i Coins, le pedine, e la "no-entry tile"
- Organizzazione delle carte in "Deck", noi la gestiamo più algoritmicamente