Peer-Review

UML

Rossi, Zuccherino, Scalmani, Simone
Gruppo GC51
Valutazione del diagramma UML delle classi del gruppo GC41.

Lati positivi

- Particolarizzazione approfondita dei metodi nelle classi
- Enumerazioni per i simboli degli angoli
- Utilizzo puntuale dei tipi e delle visibilità degli attributi

Lati negativi

- Uso eccessivo della modularità e del polimorfismo per le classi obiettivo
- Le classi Deck hanno stessi metodi e stessi attributi
  Una soluzione alternativa alla differenziazione delle due classi potrebbe essere quella di compattarle in un'unica classe (Deck) ed assegnare alla classe Game due attributi privati per differenziarne le istanze (resourceDeck, goldDeck entrambi di tipo Deck)

Confronto tra le architetture

- Dai lati positivi individuati, il gruppo ha specificato approfonditamente metodi ed attributi di ogni classe, il che renderà sicuramente piú veloce e chiara la stesura del codice. Tuttavia, a parer nostro, sebbene sia ben costruito il design pattern strategy, risultano troppo differenziate le classi obiettivo, in particolare quelle che riguardano il numero di risorse/oggetti piazzati dal giocatore, poiché, essendo il codice sempre lo stesso, a meno del tipo di risorsa/oggetto, sembrano essere ridondanti. Nel nostro caso abbiamo utilizzato solo 4 classi per definire i tipi principali di obiettivi (scala, L, conteggio risorse/oggetti) e degli attributi che specificano il particolare obiettivo (tipo di risorsa/oggetto, orientazione del pattern di carte), in questo modo si riduce notevolmente la quantitá di codice riutilizzato
- Il gruppo ha molto efficacemente utilizzato le enumerazioni compatte (in particolare la enumerazione Symbol) per definire il contenuto degli angoli delle carte, le quali nel nostro caso fanno riferimento ad una classe Angles che contiene strutture dati apposite per gli angoli ed enumerazioni specifiche per risorse ed oggetti che li caratterizzano, risultando cosí poco specifica ed efficiente
- L'utilizzo della struttura dati Queue per i deck permette efficacemente di controllare che la carta da pescare sia quella in cima al mazzo
- Come sottolineato nei lati negativi, le classi deck risultano, a parer nostro, troppo simili per essere differenziate. Nel nostro caso i Deck sono attributi privati (con struttura dati apposita) della classe Game, con i loro metodi direttamente implementati in essa (getter e setter anch'essi a carico del Game).
- A differenza del gruppo, il quale ha utilizzato un HashMap per la Board del giocatore, noi abbiamo optato per una matrice statica, la quale risulta meno efficiente in termini di memoria e tempo d'inizializzazione, ma piú

efficace nell'implementazione rendendola piú chiara e leggibile, e di piú semplice utilizzo nel conteggio degli obiettivi e della ricerca delle carte