PEER REVIEW DELL'UML DEL GRUPPO 52 DA PARTE DEL GRUPPO 53

Dopo aver analizzato con attenzione l'UML ricevuto ed esserci confrontati con il gruppo 52 stesso riguardo alcuni aspetti poco chiari, elenchiamo qui le osservazioni e le modifiche che consigliamo di fare:

1. Numerose classi non hanno metodi Getter e Setter nonostante gli attributi al loro interno siano tutti private. Il motivo di questo, come ci è stato spiegato è di semplificare lo schema UML, tuttavia noi consigliamo di includere i riferimenti a questi metodi per rendere facilmente comprensibile il flusso logico che seguirá il programma ad un occhio esterno. Questo consiglio si estende, per gli stessi motivi, anche ad altri tipi di metodi a cui sembrano mancare i parametri che il chiamante deve passare.
2. Nel caso specifico delle classi Professor e Student che fanno entrambe riferimento alla classe Disk_colour, noi pensiamo che si possa semplificare questa parte facendo in modo che Student stessa sia una classe Enum con le stesse enumeraioni di Disk_colour, mentre la classe Professor puó essere eliminata gestendola con l'array di booleani giá presente nella classe Board.
3. Nella classe Board oltre al metodo Add_student, pensiamo sia necessario aggiungere anche un Remove_student, visto che, per esempio, un'istanza di Student puó essere spostata dalla Board ad una Island durante il gioco. Inoltre, l'array arr_pos_students che per come abbiamo compreso serve a tenere conto di quanti studenti sono seduti ai tavoli, puó essere sostituito con una Map<Disk_colour, ArrayList<Student>> in modo da non aver bisogno del metodo colour_conversion, ed essere coerenti con la gestione delle istanze di Student che si spostano dalla entrance ai tavoli.
4. Specificare un range di lunghezze che un array puó assumere come in Game_table con Board[2…4], a nostro avviso non è corretto a livello sintattico. Capiamo tuttavia il motivo di questa scelta e riteniamo corretto l'uso di un array in quel contesto, anche se potrebbe essere sostituito senza alcun problema da una ArrayList.
5. Game e Turn_control sono per definizione dei Controller perció sarebbe errato lasciarli nel Model. Quanto sará fatto questo spostamento, riteniamo che si possa gestire in modo diverso la classe Player, collegandola a Game_Table, ed avendo un Player Controller al suo posto
6. È chiara l'intenzione di gestire quasi tutta la logica di gioco nei Controller, il che è giusto. Riteniamo peró che si possano implementare dei metodi all'interno di Model (richiamabili dai controller) per gestire gli spostamenti delle istanze (di Student per esempio) tra le classi del Model
7. In ultimo ci sono degli errori minori, per esempio la mancanza di una discard_pile nella classe Deck e del concetto di carte in mano al giocatore, ed il fatto che il metodo Play_assistance ritorni un tipo Deck, che (forse per una nostra mancanza) non siamo riusciti a spiegare.

In conclusione, riteniamo che l'UML vada perfezionato e reso piú leggibile ad occhi esterni al gruppo, rendendo espliciti i vari ragionamenti e processi su cui è basato. Detto questo, è una buona base per gli step successivi del progetto. Qui infatti sono elencati solo quelli che noi abbiamo visto come difetti, ma in realtá ci sono molte scelte interessanti, alcune cha abbiamo fatto noi stessi.