Relazione delle Responsabilità

del progetto di metodologie di programmazione

di Raganini Tommaso

Come richiesto, di seguito verrà proposta una breve descrizione delle responsabilità del progetto, da me implementato, di formula1.
In questa specifica era stato richiesto di sviluppare un programma che: “permetta di gestire una gara dove *giocatori interattivi* (umani) e *giocatori bot* (programmati) concorrono per vincere la gara. “
Di seguito l’elenco delle responsabilità che sono state individuate:

- Gestione del tracciato
  Accesso alle componenti della pista e generazione della matrice del tracciato
  (Interfaccia IRaceTrack fornisce i metodi che permettono di accedere ai componenti del tracciato, Classe TrackGenerator genera la matrice di stringhe)

- Gestione dell’input utente
  Sincronizzazione col programma per quanto riguarda l’attesa che si verifica quando viene richiesta la direzione all’utente da GUI. Il thread va in wait fino a che non viene sbloccato da un altro che lo notifica che la direzione è stata inserita e la gara può continuare.
  (Interfaccia MoveListener implementata da MoveviewController mette a disposizione i metodi necessari alla sincronizzazione per questo processo)

- Gestione dei giocatori
  I giocatori contengono un metodo move che è implementato in maniera diversa a seconda delle classi e cambia il modo in cui viene acquisita la direzione in cui spostarsi, richiamando poi il metodo move della classe Car che procede allo spostamento di posizione
  (Classe astratta Player estesa da Bot e HumanPlayer. In HumanPlayer il metodo move è implementato per prendere la direzione direttamente da inserimento tramite GUI)

- Gestione del veicolo
  La gestione del veicolo è una responsabilità che comporta la gestione dell’inerzia della macchina in base alla velocità a cui sta andando, tutto ciò è gestito grazie al metodo move nella classe Car che permette di aggiornare il vettore accelerazione in base alla direzione passata per parametro (Classe Car e Classe astratta DirectionVector estesa da Velocity)

- Gestione della Lettura dei file
  La lettura dei file è fondamentale per l’inizializzazione del tracciato e dei giocatori bot all’interno del programma. I file vengono letti dalla cartella resources e da uno viene presa la matrice di stringhe mentre dall’altro viene preso il numero di bot che vengono poi utilizzati per inizializzare il gioco. (Classe FileIOTrack)

- Gestione dell’inizializzazione del gioco
  Questa responsabilità concerne l’inizializzazione dell’array di player che poi sarà utilizzato per accedere allo stato di ogni giocatore durante lo svolgimento della gara.
  (Classe GameSetup)

- Gestione della logica di gioco
  La logica di gioco riguarda come deve essere effettuato lo svolgimento del turno per ogni giocatore, ciò e controllato da delle classi che decidono la strategia conseguentemente il motore di gioco si preoccupa di far comunicare le classi che decidono le regole di svolgimento del round insieme alla classe che effettua l’aggiornamento della GUI ad ogni turno (Classe Game, Interfaccia GameStrategy implementata da FirstRoundStrategy e NormalRoundStrategy)

- Gestione dei controlli
  I controlli che vengono effettuati su ogni singolo giocatore riguardano la lista di mosse disponibili che si hanno ad ogni turno (escludendo quelle con cui si va a sbattere) e il controllo che si accerta della vittoria o meno del giocatore
  (Interfaccia ICheker implementata da Gamechecker)

- Gestione dell’aggiornamento della GUI
  L’aggiornamento della GUI è fondamentale per far sì che l’utente riesca sia a seguire l’andamento della gara che ad avere ad ogni turno delle mosse aggiornate disponibili dalla Choicebox. (Ciò è gestito dal Controller di javafx: GraphicController che implementa l’interfaccia GameUIUpdater e si serve della classe di supporto Util che contiene dei metodi statici per disegnare e aggiornare il tracciato in base alle posizioni delle macchine in gara)