Quoridor2D

Relazione progetto OOP

Becci Alessandro,
Castellani Thomas,
D'Ambrosio Stefano,
Fabri Luca 0000892878

24 giugno 2020

Indice

1 Analisi **2**
1.1 Requisiti 2
1.2 Analisi e modello del dominio 3

Capitolo 1

Analisi

1.1 Requisiti

Il software Quoridor2D mira alla creazione di un videogioco basato sul tradizionale gioco da tavola Quoridor, il quale è giocabile da due giocatori su una griglia quadrata di 9x9 celle. L'obiettivo è quello di realizzare il gioco nella modalità standard e in una nuova modalità da noi introdotta con Power Ups.

Requisiti funzionali

- *Partita standard*: i giocatori, che partono dai lati opposti della griglia, hanno lo scopo di raggiungere la linea di partenza del rispettivo avversario, piazzando barriere che impediscono il passaggio del giocatore da una cella ad un'altra.
- *Partita con power ups*: questa modalità di gioco presenta le stesse regole della partita standard ma prevede, da parte della logica del software, il piazzemento di power ups all'interno della griglia, che permette di accumulare vantaggi al fine della vittoria.
- Una partita varia da due a tre rounds: vince il giocatore che ne ha vinti due.
- I giocatori giocano a turno e possono decidere di muoversi o piazzare una barriera.
- La partita deve essere giocata affinchè ogni avversario possa raggiungere la rispettiva destinazione. Una barriera viene considerata illegale, e quindi non verrebbe piazzata, se violerebbe la suddetta regola.

Requisiti non funzionali

- Una partita deve avere la possibilità di essere salvata e ripresa in un secondo momento.
- Il software deve prevedere la visualizzazione della leaderboard con lo storico dei vincitori.
- Il software deve essere dotato di una GUI semplice ed intuitiva.
- Il software deve essere portabile nei sistemi Linux, Mac e Windows.

1.2 Analisi e modello del dominio

In un round di una partita di una qualsiasi modalità, i due giocatori effettuano alternativamente mosse e piazzamenti finchè certe condizioni di vittoria si verificano. Ognuno di essi, escludendo il bordo della griglia, può muoversi di una cella a destra, o a sinistra, o in alto o in basso. Se i due giocatori si trovano uno di fronte all'altro, quello del turno corrente può effettuare un salto in direzione dell'avversario o, nell'impossibilità, effettuare una mossa diagonale.

Ogni round di una partita è a sè stante: al cambio di ognuno la griglia viene ripulita dalle barriere e le posizioni iniziali dei giocatori rigenerate.

Analogamente, le entità presenti nel round corrente (pedine, barriere, power ups), non presentano dipendenze e quindi, nel modello di dominio possono essere gestite indipendentemente.

Nella Nella Figura 1.1 vengono riportate le componenti principali del modello del dominio. `RoundEnvironment` è la componente che gestisce il singolo round ed il suo scopo è di tenere traccia delle entità precedentemente descritte, mentre `BarrierPlacer` e `PlayerMover` tutelano rispettivamente il posizionamento delle barriere e lo spostamento dei giocatori.

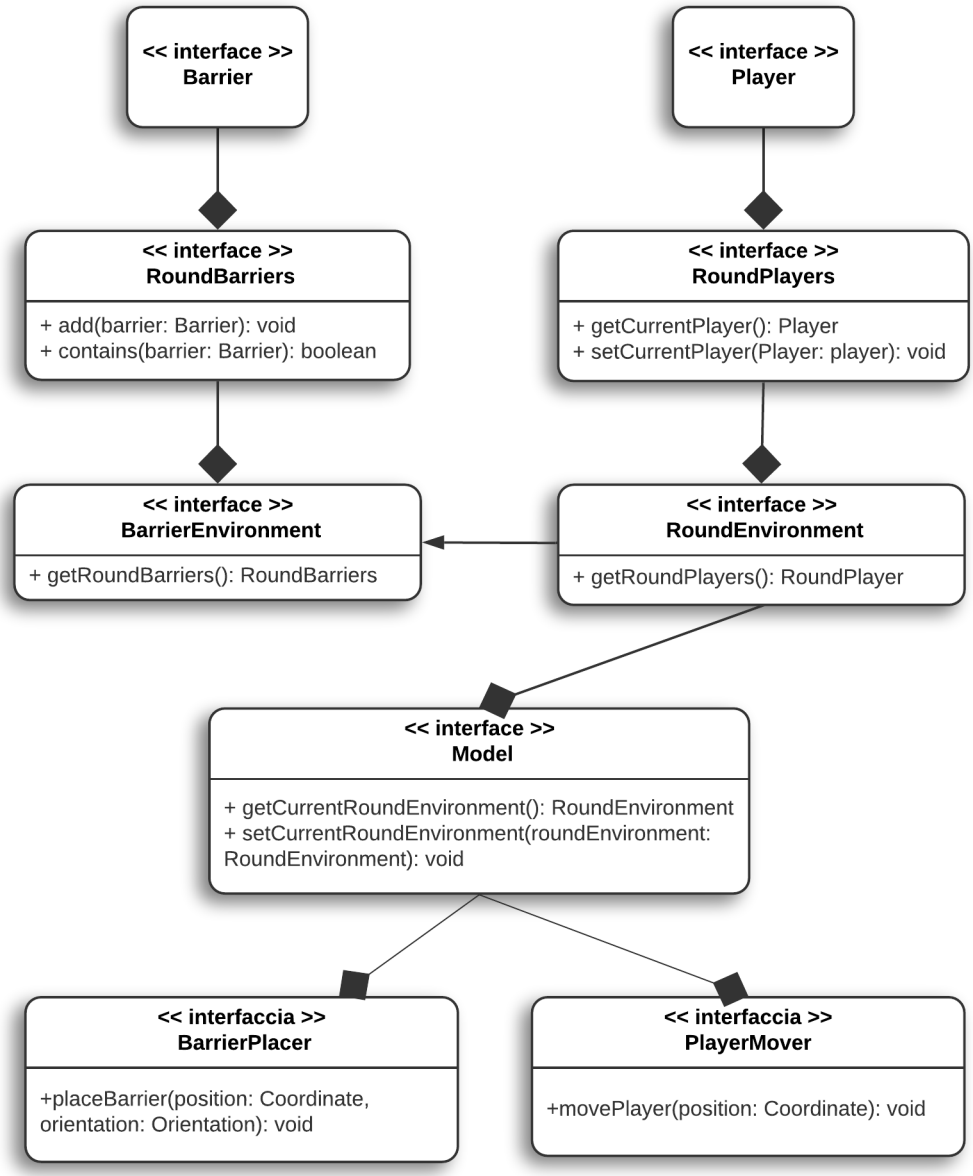

Figura 1.1: schema UML raffigurante la rappresentazione del problema attraverso le entità principali della partita standard