SAPIENZA Università di Roma
Facoltà di Ingegneria e Facoltà di Ingegneria dell'Informazione
Corsi di Laurea in Ingegneria Informatica ed Automatica ed in Ingegneria dei Sistemi Informatici

Corso di Progettazione del Software

Esame del **21 Giugno 2010**

Tempo a disposizione: 3 ore

Requisiti. L'applicazione da progettare riguarda una versione modificata del gioco calcistico del torello. Un giocatore ha un nome (una stringa) ed un avatar (una stringa che rappresenta il path al file `.jpg`). I giocatori sono poi suddivisi in gialli e rossi. Un torello consiste di due insiemi ordinati, uno di giocatori rossi (almeno uno) ed uno di giocatori gialli (almeno uno). Un torello ha un nome (una stringa). Esiste una variante, il torello multipalla, in cui si gioca con N palle (N $\geq$ 2).

Un giocatore è inizialmente "non in gioco". Quando riceve l'evento "inizio partita" passa allo stato "in gioco". Ciascun giocatore in gioco riceve una palla (evento "palla") e la rilancia ad un altro giocatore (nuovo evento "palla"). Un giocatore non può rilanciare la palla a se stesso, e deve rilanciarla solamente ad un giocatore dell'altro colore (ad es., se è un giocatore rosso, può aver ricevuto la palla solo da un giocatore giallo e può rilanciarla solamente ad un giocatore giallo, non rosso). Quando un giocatore riceve l'evento di "fine partita" il giocatore torna nello stato "non in gioco". Il giocatore può essere modificato solo quando è in "non in gioco".

Siamo interessati a progettare l'attività del gioco, che prende come parametro un torello (con le palle, ma senza i giocatori) e procede leggendo in input M giocatori e poi inserendoli nel torello. Quindi, contemporaneamente si procede a: *(i)* giocare, attraverso l'invio in broadcasting a tutti i giocatori dell evento "inizio partita" e dell'ulteriore invio dell'evento"palla" ad un giocatore qualsiasi[1]; *(ii)* mettersi in attesa (attraverso un'opportuna attività di input/output) del comando di fine esecuzione da parte dell'utente, che termina il gioco riportando tutti i giocatori nello stato "non in gioco".

Domanda 1. Basandosi sui requisiti riportati sopra, effettuare la fase di analisi producendo lo schema concettuale in UML per l'applicazione, comprensivo del diagramma delle classi, diagramma stati e transizioni per la classe *Giocatore*, diagramma delle attività, specifica del diagramma stati e transizioni, specifica delle attività complesse e specifica della attività atomica di inserimento dei giocatori nel torello, motivando, qualora ce ne fosse bisogno, le scelte effettuate.

Domanda 2. Effettuare la fase di progetto, illustrando i prodotti rilevanti di tale fase e motivando, qualora ce ne fosse bisogno, le scelte effettuate.

È obbligatorio definire solo le responsabilità sulle associazioni del diagramma delle classi.

Domanda 3. Effettuare la fase di realizzazione, producendo un programma JAVA e motivando, qualora ce ne fosse bisogno, le scelte effettuate.

È obbligatorio realizzare in JAVA solo i seguenti aspetti dello schema concettuale:

- Le classi *Giocatore* e *GiocatoreRosso* (con le eventuali classi *fired* corrispondenti) e le *associazioni* che le legano alla classe *Torello*.
- L'*attività principale*, l'*attività atomica di inserimento dei giocatori nel torello*.

[1] Se il torello Ã¨ multipalla, si inviano tanti eventi "palla" ad altrettanti giocatori differenti del torello.