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

Corso di Progettazione del Software

Esame del **18 febbraio 2015**

Tempo a disposizione: 3 ore

Requisiti. L'applicazione da progettare riguarda la gestione di un sistema per la gestione di incroci di auto senza guidatore. Un incrocio è caratterizzato da un codice (una stringa) e un insieme di strade che sono connesse ad esso, almeno 2. Ogni strada ha un nome, alcune strade sono ad alto scorrimento e per queste è di interesse la capacità nominale (un intero). In ogni momento, l'incrocio è impegnato da un insieme ordinato (evetualmente vuoto) di auto. Ogni auto che impegna l'incrocio proviene da una delle strade che sono connesse all'incrocio stesso. Delle auto interessa la targa. Ad ogni incrocio è associato esattamente un controllore automatico e viceversa ogni controllore automatico è associato ad un solo incrocio. Il controllore è caratterizzato da un codice (una stringa). Le auto e i controllori sono oggetti attivi. Delle auto non interessa qui il funzionamento; invece i controllori si comportano come segue.

Un controllore è inizialmente a *riposo*. Quando viene segnalato l'*arrivo* di un'auto, passa nello stato di *impegnato* e aggiorna l'insieme delle auto che impegnano il suo incrocio. Se nello stato di impegnato viene segnalato l'*arrivo* di un'altra auto, continua ad aggiornare l'insieme. Quando è impegnato e riceve il segnale *incrocio pronto* fa partire la prima auto, rimuovendola dall'insieme, e passa in uno stato di *attesa*. Quando nello stato di attesa riceve il segnale che l'*auto* è effettivamente *passata* torna nello stato di impegnato, se ci sono ancora auto, altrimenti torna nello stato di riposo.

Siamo interessati alla seguente attività principale. L'attività prende in input un incrocio I e concorrentemente esegue le seguenti due sottoattività: *(i)* funzionamento, e *(ii)* analisi. La sottoattività di funzionamento *(i)* avvia il controllore associato all'incrocio I, che inizia a gestire l'arrivo delle auto, e si mette in attesa del comando di fine da parte dell'utente che fa terminare il funzionamento dell'incrocio stesso. La sottoattività di analisi *(ii)* calcola e stampa l'insieme delle strade che sono connesse all'incrocio I, segnalando quali sono ad alto scorrimento. Una volta che tali sottoattività sono state completate, si stampa un messaggio di saluto e si termina.

Domanda 1. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in UML per l'applicazione, comprensivo del diagramma delle classi (inclusi vincoli non esprimibili in UML), diagramma stati e transizioni per la classe *Controllore*, diagramma delle attività, specifica del diagramma stati e transizioni, e specifica dell'attività principale e delle sottoattività NON atomiche (indicando in modo esplicito quali attività atomiche sono di I/O e quali sono Task), motivando, qualora ce ne fosse bisogno, le scelte di progetto.

Domanda 2. Effettuare il progetto, illustrando i prodotti rilevanti di tale fase e motivando, qualora ce ne fosse bisogno, le scelte di progetto. È obbligatorio definire solo le responsabilità sulle associazioni del diagramma delle classi.

Domanda 3. Effettuare la realizzazione, producendo un programma JAVA e motivando, qualora ce ne fosse bisogno, le scelte di progetto. È obbligatorio realizzare in JAVA solo i seguenti aspetti dello schema concettuale:

- La classe `Controllore`, con la classe `ControlloreFired`, e le classi JAVA per rappresentare le *associazioni* di cui `Controllore` ha responsabilità.
- L'*attività principale* (NON le sue sottoattività).