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 **11 settembre 2015**

Tempo a disposizione: 3 ore

Requisiti. L'applicazione da progettare riguarda il gioco "AntWorld", descritto nel seguito. Un formicaio ha un nome a vi appartengono un insieme non vuoto di formiche. Ogni formica a sua volta ha un nome ed appartiene esattamente ad un formicaio. Le formiche trovano briciole (di cibo). Ogni briciola è caratterizzata da un codice (un intero). Le briciole possono essere piccole o grandi. Una briciola piccola può essere portata proprio (al formicaio) da una formica che l'ha trovata. Una briciola grande può essere portata (al proprio formicaio) da una formica che l'ha trovata solo se aiutata da un'altra formica.

Ciascuna formica inizialmente è nello stato di *ricerca* delle briciole. Se trova una bricola (segnalato da un evento generato automaticamente con modalità che non interessano) e la briciola è piccola, allora (dopo aver aggiorna opportunamente il diagramma delle classi) la porta al proprio formicaio passando in uno stato *inCammino* verso il formicaio. Quando arriva al formicaio (segnalato da un evento generato automaticamente con modalità che non interessano) torna nello stato di *ricerca*. Se, invece, la briciola trovata è grande, allora chiede aiuto ad un'altra formica del formicaio scelta casualmente (i dettagli della funzione random non interessano) che non sta già portando o aiutando a portare una briciola e si mette in *attesa* dell'aiuto (segnalato da un evento di risposta lanciatogli dalla formica richiesta). Quando questo avviene, porta la bricola al formicaio mettendosi nello stato di *inCammino*. Infine se nello stato di *ricerca* riceve una richiesta di aiuto da parte di un'altra formica per portare una bricola grande, dà il suo aiuto alla stessa (lanciandogli un evento di risposta), mettendosi nello stato di *inCammino*.

Siamo interessati alla seguente attività principale. L'attività prende come parametro di input un insieme di formicai F, con rispettive formiche, ed un insieme di bricole B. Come prima cosa inizializza i dati delle formiche rimuovendo qualsiasi link con briciole. Poi prosegue con due sottoattività concorrenti: *gioco* e *analisi*. La sottoattività di *gioco* fa iniziare a giocare tutte le formiche (i dettagli non interessano) e poi si mette in attesa del segnale di input di fine da parte dell'utente che fa terminare la partita. La sottoattività di *analisi* calcola e stampa (segnale di output) la percentuale di briciole grandi in B. Una volta che tali sottoattività sono state completate, l'attività principale invia un segnale di output "OK" e 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 *Formica*, diagramma delle attività, specifica del diagramma stati e transizioni, e specifica dell'attività principale (NON delle sottoattività), 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 `Formica`, con la classe `FormicaFired`, e le classi JAVA per rappresentare le *associazioni* di cui `Formica` ha responsabilità.
- L'*attività principale* (NON le sue sottoattività).