Progetti

Software Engineering

AY 2023/24

Enrico Tronci
Computer Science Department, Sapienza University of Rome
Via Salaria 113 - 00198 Roma - Italy

tronci@di.uniroma1.it

Version: 2023-12-10

1 Introduzione

Nel seguito si propongono alcuni progetti d'esame per il corso di *Ingegneria del Software*.

Oltre ai progetti proposti dal docente gli studenti possono proporre dei loro progetti.

I progetti possono essere svolti in gruppo. Ciascun gruppo è composto da almeno uno studente ed al più 3 studenti.

I progetti possono essere discussi prima o dopo la prova a scelta multipla. Il voto viene verbalizzato quando sono completati sia la prova a scelta multipla che il progetto.

Ciascun gruppo, una volta scelto il progetto (sia tra quelli proposti dal docente che tra quelli proposti dagli studenti) deve mandare una email al docente ed attendere l'approvazione. Questo perché viene implementato una sorta di *load balancing* per evitare che tutti i gruppi si concentrino sugli stessi progetti.

Ciascun gruppo deve consegnare:

- Una relazione sul progetto.
- Il software per il progetto.

2 Struttura della relazione

La relazione conterrà le seguenti sezioni.

2.1 Descrizione generale

Una descrizione generale del sistema che si vuole realizzare.

Questa descrizione dovrebbe anche contenere almeno una figura che illustra il sistema ed il suo ambiente operativo.

2.2 User requirements

Una descrizione dei requisiti utenti.

Ogni requisito deve essere numerato. Si ponga attenzione al fatto che ogni requisito esprima un solo vincolo sul sistema.

Questa sezione deve contenere i diagrammi *Use Case* dell'UML per almeno due use cases.

2.3 System requirements

Una descrizione dei requisiti di sistema.

Ogni requisito deve essere numerato. Si ponga attenzione al fatto che ogni requisito esprima un solo vincolo sul sistema.

Questa sezione conterrà inoltre:

- Il diagramma dell'architettura del sistema.
- Almeno un *Activity Diagram* UML per mostrare come le componenti del sistema concorrono ad soddisfare i requisiti utenti.
- Almeno uno *State Diagram* UML per una delle componenti del sistema.
- Almeno un *Message Sequence Chart* UML per la comunicazione tra le componenti del sistema.

2.4 Implementation

Una descrizione generale dell'implementazione.

Una descrizione con pseudo-codice per tutte le componenti del sistema.

Lo schema del (o dei) DB usati.

Una descrizione delle connessioni con Redis.

2.5 Risultati Sperimentali

Descrivere i risultati ottenuti dalla simulazione del sistema.

3 Struttura del software

Il software di ogni progetto deve contenere i seguenti elementi.

3.1 Test generator

Il *test generator*, cioè un modello dell'ambiente in cui il software opera. Tale test generator nel seguito verrà anche chiamato *Environment* per evidenziare il fatto che esso modello inputs dall'ambiente operativo per il software.

Assicurarsi che l'environment sia completo, cioè sia in grado con probabilità non nulla di generare qualsiasi sequenza di test.

3.2 System Under Design (SUD)

Il *System Under Design* (SUD), cioè il sistema che si sta progettando.

Il SUD è realizzato in C++ e consiste in processi comunicanti con Redis e con un DB PostgreSQL per i dati (se necessario) e per i log delle esecuzioni (sempre necessari per la realizzazione dei monitors).

Ciascun processo è una macchina a stati che legge da streams Redis, esegue computazioni e ritorna outputs su streams Redis.

Si raccomanda di realizzare processi piccoli in modo che siano facilmente verificabili e validabili.

3.3 Monitor

Un monitor per ciascuno dei requisiti di interesse. Si realizzino almeno tre monitors per i requisiti funzionali ed almeno due monitors per i requisiti funzionali.

I monitor prendono input dal DB con i log e ritornano il proprio output sul medesimo DB.

4 Lista Progetti

Questa sezione contiene una lista non esaustiva di possibili progetti.

4.1 Backend sito e-commerce

Si progetti il backend di un sito di e-commerce.

Il progetto deve includere i seguenti componenti.

1. Un modello (test generator) per i customers, ciò coloro che acquistano i prodotti in vendita.
2. Un modello per i fornitori, ciò coloro che inseriscono nel sito prodotti da vendere.
3. Un modello per i trasportatori, cioè coloro che consegnano il prodotto al customer.
4. Uno o più server ai quali i customers si connettono per interagire con il sistema.
5. Uno o più server ai quali i fornitori si connettono per interagire con il sistema.
6. Uno o più server ai quali i produttori si connettono per interagire con il sistema.
7. Un DB per i dati (ad esempio, prodotti disponibili, etc) ed i log (ad esempio, lista delle transazioni).
8. Monitors per almeno tre proprietà funzionali.

9. Monitors per almeno due proprietà non-funzionali.

4.2 Controllo formazione droni

Si progetti il centro di controllo per una formazione di droni che deve sorvegliare un data area.

Ogni drone ha un autonomia di 30 minuti di volo ed impiega un tempo di minimo 2h massimo 3h per ricaricarsi. Il tempo di ricarica è scelto ad ogni ricarica uniformemente a random nell'intervallo [2h, 3h].

Ogni drone si muove alla velocità di 30 Km/h. L'area da monitorare misura 6×6 Km.

Il centro di controllo e ricarica si trova al centro dell'area da sorvegliare.

Il centro di controllo manda istruzioni ai droni in modo da garantire che per ogni punto dell'area sorvegliata sia *verificato* almeno ogni 5 minuti.

Un punto è *verificato* al tempo t se al tempo t c'è almeno un drone a distanza inferiore a 10m dal punto.

Il progetto deve includere i seguenti componenti.

1. Un modello (test generator) per i droni.
2. Un modello per il centro di controllo.
3. Un DB per i dati (ad esempio, stato di carica dei droni) ed i log).
4. Monitors per almeno tre proprietà funzionali.
5. Monitors per almeno due proprietà non-funzionali.

4.3 Anomaly Detection

Si progetti un semplice sistema di anomaly detection per uno stream di dati.

Il rilevatore di anomalie calcola il valor medio di ogni stream e la covarianza dell'insieme di streams (visto come una serie temporale multivariata) su una finestra temporale di ampiezza W configurabile.

Ogni volta che uno dei valori medi o delle covarianze si discosta significativamente da quelli correnti lancia un allarme.

Il progetto deve includere i seguenti componenti.

1. Un modello (test generator) che legge uno stream di dati da file csv e lo manda su altrettante streams Redis. Un possibile file csv con dati reali verrà fornito dal docente.
2. Un sistema che dalle stream Redis calcola il valor medio per ogni stream e lo salva nel DB.
3. Un sistema che dalle stream Redis calcola la coviaranza per ogni coppia di stream e la salva nel DB.
4. Monitors per almeno tre proprietà funzionali.
5. Monitors per almeno due proprietà non-funzionali.

4.4 Insulin Pump

Si progetti e verifichi una semplice pompa per l'insulina usando la strategia di controllo presentata nel libro di testi od altra di vostra scelta.

Il progetto deve includere i seguenti componenti.

1. Un modello (*virtual patient*) che lega il livello glicemico all'insulina iniettata, all'esercizio fisico ed all'alimentazione. Questo verrà preso dalla letteratura e fornito dal docente.
2. Un modello per l'alimentazione (quando e quanto il paziente si alimenta).
3. Un modello per l'esercizio fisico (quando e quanto il paziente compie esercizio fisico).
4. Un modello per la strategia di controllo della pompa dell'insulina.
5. Monitors per almeno tre proprietà funzionali.
6. Monitors per almeno due proprietà non-funzionali.