Software Engineering
process report template

Murgia Antonio

Alma Mater Studiorum – University of Bologna
viale Risorgimento 2, 40136 Bologna, Italy
antonio.murgia2@studio.unibo.it

1 Introduction

2 Vision

Non esiste codice senza progetto, non esiste progetto senza analisi del problema e non esiste problema senza requisiti.
Non c'è codice senza test.
Mantenere disaccopiati logica realizzativa del progetto e testing.
Individuare diverse tipologie di test da applicare in diversi momenti della realizzazione:

- Unit testing (singolo componente, white box)
- Integration Testing (di interazione, white box)
- Functional Testing (sull'intero sistema, black box)
- Stress Load Testing (performance, black box)
- User Acceptance Testing (feedback utente, black box)

Utilizzare approccio top-down in fase di progettazione e bottom-up in fase di implementazione (zooming).

3 Goals

4 Requirements

Design and build a ButtonLed software system in which a Led is turned on and off each time a Button is pressed (by an human user). The system should run on a single support, e.g. a conventional PC, a Raspberry Pi or Arduino.

5 Requirement analysis

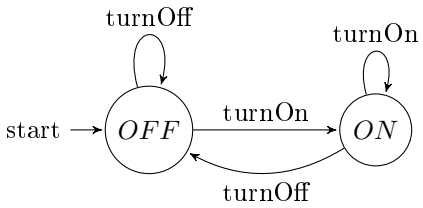
5.1 Use cases

5.2 Scenarios

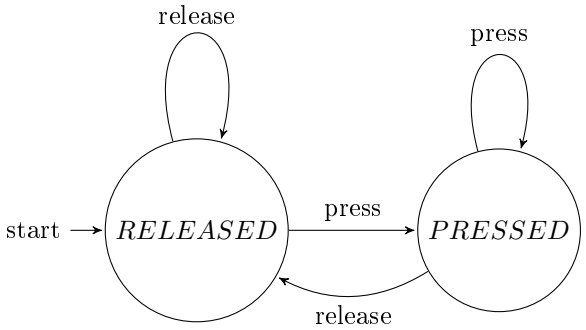
5.3 (Domain)model

LED:
Struttura: entità atomica, passiva.

Interazione: espone le operazioni void turnOn, void turnOff e boolean getState. Queste operazioni sono anche metodi nell'accezione OOP.
Comportamento: alla creazione un LED è spento. L'esecuzione dell'operazione turnOn accende il Led, l'esecuzione dell'operazione turnOff lo spegne. L'operazione getState ritorna lo stato del Led (false -> spento, true -> acceso).
Viene esplicitato dal seguente automa a stati finiti:



BUTTON:
Struttura: entità atomica, attiva.
Interazione: una sola operazione per conoscere lo stato, boolean getState˙ Comportamento: alla creazione un button è released.
Viene esplicitato dal seguente automa a stati finiti:



Button e Led non sono correlati -> il button non ha alcuna conoscenza del led e viceversa, per realizzare il sistema è necessaria un'entità coordinatrice.

5.4 Test plan

LED:
Il test plan (di tipo unit) deve verificare che inizialmente getState ritorni il valore false. Dopo aver eseguito l'operazione turnOn, getState deve restituire il valore true.
BUTTON:
É necessaria una nuova entità (DebuggableButton) per poter modificare lo stato del button, tale entità esporra le operazioni, void press, void release.

See [1] until page 11 (CMM) and pages 96-105.

13 Information about the author

Photo of the author

References

1. A. Natali and A. Molesini. *Costruire sistemi software: dai modelli al codice*. Esculapio, 2009.