Documentazione

Versione 1: 22/10/2024

Software Engineering Management (capitolo 2)

DEVE contenere un PROJECT PLAN per il progetto - con i 14 punti come indicato in sezione 2.1 del libro. Il project plan

DEVE essere consegnato (condiviso) almeno 1 mese prima dell'esame.

POTREBBE essere modificato mano a mano e tenuto traccia delle versioni.

Software Life Cycle (capitolo 3)

Abbiamo utilizzato il framework SCRUM per gestire e sviluppare il nostro software.

Abbiamo organizzato il nostro lavoro seguendo i metodi offerti da SCRUM:

- Product Backlog
- Sprint Planning
- Sprint
- Daily Scrum
- Sprint Review
- Sprint Retrospective
- Roles

DEVE indicare il tipo di processo di sviluppo seguito

PUO' formalizzare con diagramma UML una piccola parte del sw development (ad esempio una change request come viene trattata)

DOVREBBE anche usare un approccio MDA o parte di esso (con Yakindu o Papyrus UML come visto in classe)

POTREBBE presentare il progetto come SPL

Documentazione: breve documento (1 GESTIONE DEL PROGETTO) in cui indicare il tipo di processo seguito effettivamente. Eventuali differenze rispetto a quanto previsto nel project plan vanno indicate qui. Anche ad esempio come sono stati organizzati il timebox o gli sprint (ad esempio con le date) vanno indicati qui.

Configuration Management (capitolo 4)

Utilizzo di GitHub, dei relativi comandi (add, commit, push, pull, fetch, clone, merge) e dei tools offerti da GitHub, come Project Board, di tipo kanban, con il quale vengono gestiti gli elementi da implementare e i bug da sistemare

Il link per accedere al repository è il seguente: https://github.com/ZanottiMatteo/BGTransports

Aggiungere statistiche

DEVE usare un sistema per il configuration management (consigliato github) con i relativi comandi (git add, commit, push etc.)

DEVE dimostrare di aver usato issuses, branches, pull requests, e code reviews

PUO' usare la project board (tipo kanban o varianti simili)

Documentazione: aggiungere al documento (1) eventuali strumenti usati. Gli issue e l'uso di github è da vedere sul sito github (non è necessario documentarlo ma si possono portare alcune statistiche come numero di commit, di issues e cose del genere).

People Management and Team Organization (capitolo 5)

Abbiamo utilizzato un approccio agile allo sviluppo del software: SCRUM.

Il team lavora basandosi su una buona collaborazione, pianificazione, comunicazione e flessibilità.

Le persone che lavoro al progetto sono 2:

1. Maffeis Riccardo -> Scrum Master che si occupa di far seguire le pratiche Scrum corette e risolve eventuali problematiche.
2. Zanotti Matteo -> Product Owner che si occupa di definire le funzionalità e le priorità del prodotto

Entrambi si occupano della programmazione del software

DEVE dire le persone e l'organizzazione del lavoro tra le persone (rifacendosi ad uno schema del libro)

Documentazione: aggiungere al documento (1) la documentazione riguardante questo punto.

Softare Quality (capitolo 6)

POTREBBE contenere una lista di qualità che hanno guidato il progetto con la loro spiegazione relativa al contesto

Documentazione: aggiungere al documento (2 – vedi dopo) la documentazione riguardante questo punto.

Requirement Engineering (capitolo 9)

DOVREBBE indicare come i requisiti sono stati elicitati

DEVE contenere la specifica dei requisiti (ad esempio seguendo lo IEEE 830)

DEVE ???

Documentazione: Documentare i requisiti in documento (2 REQUISITI). La descrizione dei requisiti può seguire lo schema visto in classe dei requisiti presentati in https://github.com/foselab/abz2024_casestudy_MLV/blob/main/Mechanical_Lung_Ventilator%201_5.p df Al suo intero il documento dovrebbe riportare i casi d'uso ed eventuali altri diagrammi UML (ad esempio macchine di stato)

Modelling (Libro UML @ Classroom)

DEVE contenere i diagrammi UML visti a lezione, se non già presentati in altre sezioni della documentazione: - use case diagram (si consiglia di metterlo nella sezione dei requisiti) - class diagram (dal quale si

DEVE generare una prima versione del codice in Java usando Papyrus Designer - si consiglia di metterlo nella sezione software design) - ALMENO UNO state machine diagram (NON DEVE essere troppo semplice, si deve cercare di usare la maggior parte degli elementi che costituiscono uno state diagram come visto a lezione) - ALMENO UN sequence diagram (NON DEVE essere troppo semplice, si deve cercare di usare la maggior parte degli elementi che costituiscono un sequence diagram come visto a lezione). - UN diagramma TRA communication diagram e timing diagram - ALMENO UN activity diagram (NON DEVE

essere troppo semplice, si deve cercare di usare la maggior parte degli elementi che costituiscono un activity diagram come visto a lezione) - component diagram (si consiglia di metterlo nella sezione in cui si presenta la software architecture)

Software Architecture (capitolo 11)

DEVE contenere la descrizione dell'architettura con almeno un paio di architectural views (per differenti punti di vista)

DOVREBBE avere almeno una vista con connettori e componenti con la descrizione dello stile architetturale (11.4)

DEVE utilizzare almeno una libreria esterna con maven. Ad esempio l'uso di log4j è molto consigliata.

Software Design (capitolo 12)

DEVE contenere una descrizione del design (mediante i diagrammi UML va bene)

POTREBBE contenere un calcolo di complessità (ad esempio con McCabe) di una piccola parte

DOVREBBE contenere qualche misurazione del codice, (con qualche metrica che abbiamo visto). Alcuni tools che vedremo a lezione: stanide, jdepend, struture101, sonarlint, PMD …

DEVE applicare un paio di design pattern visti a lezione

Documentazione: Documentare l'architettura e il design in documento (3 DESIGN).

Software Testing (capitolo 13)

PUO' avere un documento di plan per l'attività di test

DEVE contenere dei casi di test di unità implementati con la loro descrizione nel documento DOVREBBE avere qualche misura di copertura per i casi di test

Documentazione: Documentare i test in un documento (4 TESTING).

Software Maintenance (capitolo 14)

POTREBBE contenere un di attività di reverse engineering (se si è partiti da codice esistente)

DOVREBBE documentare alcune attività di refactoring che sono state fatte.

Documentazione: Documentare eventuali attività di refatoring in un documento (5 MAINTENANCE)

Funzionalità	Data

Bug	Data

Daily	Data	Modalità	Sprint
1 - 5	21/10 – 25/10 (2024)	Riunioni in presenza giornaliere	1
6-10	28/10 – 1/11 (2024)	Riunioni in presenza e a distanza giornaliere	2
11-15	04/11 – 08/11 (2024)	Riunioni a distanza giornaliere	3
16-20	11/11 – 15/11 (2024)	Riunioni in presenza giornaliere	4
21-25	18/11 – 22/11 (2024)	Riunioni in presenza giornaliere	5