UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

COURSE SYLLABUS

Software Engineering

2223-3-E3101Q119

Obiettivi

Acquisire conoscenze più avanzate di sviluppo del software rispetto a quelle acquisite durante il corso del II anno di analisi e progettazione del software. Conoscere ed applicare i pattern architetturali durante lo sviluppo del software. Identificare e rimuovere violazioni del codice attraverso l'utilizzo ed il supporto fornito da alcuni tool come SonarQube e SonarCloud Introduzione a DevOps e alla Continuous Integration: utilizzo di Travis (distributed continuous integration service).

Contenuti sintetici

Principi, tecniche e strumenti per lo sviluppo del software. Pattern architetturali ed esempi della loro applicazione nello sviluppo del software. Best practices in Java. Valutazione della qualità del codice attraverso SonarQube e altri strumenti. Esempi di progetti software complessi e valutazione della loro qualità e delle principali problematiche che possono emergere.

Programma esteso

1 Presentazione del corso. Obiettivi e contenuti.
Ingegneria del software: introduzione alla model-driven software engineering, component-based software engineering, service-oriented software engineering, distributed software engineering.

2 Applicazione dei design pattern nello sviluppo del software. Modellazione di un framework di persistenza con i design pattern. Sviluppo di un sistema di gestione di risorse con i design pattern.

3 Architetture software. Progettazione dell'architetture software. Pattern architetturali per le applicazioni enterprise.

Diagramma dei package, dei componenti e di deployment.

4 Best practices in Java. Reflessivitá in Java. Sviluppo di strumenti come plug-in di Eclipse (Lab).

5 Sistemi self-managed e self-adaptive: concetti fondamentali, domini applicativi, e case study. Model-driven engineering.

6 Service-oreinted software engineering: concetti fondamentali. Migrazione verso micro servizi.

7 Valutazione della qualita' del software. Metriche per la valutazione del software. Utilizzo del tool Understand.

8. Identificazione e rimozione di violazioni nel codice attraverso SonarQube (Lab).
9. Utilizzo di Git e GitHub, cooperazione in team durante lo sviluppo di un progetto (Lab).
10. Gestione dei progetti software: concetti base. Pianificazione dei progetti: diagrammi Gantt e PERT. Gestione dei rischi, gestione della qualitá.
11. Utilizzo di SonarCloud (Lab)
12. Introduzione al DevOps
13. Continuous Integration: utilizzo di Travis (distributed continuous integration service)

Prerequisiti

Analisi e Progettazione orientata agli oggetti.

Programmazione in Java.

Modalità didattica

Le lezioni si dovrebbero svolgere in presenza.

Lezioni ed esercitazioni in aula. Il materiale didattico sarà disponibile nella piattaforma dmoodle. Attività ed esercizi di sviluppo software in laboratorio. Uso di Git, GitHub e SonaQube in laboratorio.

Il corso è erogato in lingua italiana.

Materiale didattico

Sommerville, Ingegneria del Software, Pearson, 8° ed, 2007.

C. Larman, Applicare UML e i Pattern – analisi e progettazione orientata agli oggetti, Pearson, 3° ed, 2005. (capitoli non visti nell'insegnamento del II anno di Analisi e Progettazone).

Slide, articoli , capitoli di libri e tutorial online di approfondimento sui vari argomenti del corso disponibili su moodle.

La maggior parte del materiale su cui dovrete studiare sarà disponibile on line.

Periodo di erogazione dell'insegnamento

I semestre

Modalità di verifica del profitto e valutazione

La verifica dell'apprendimento comprende lo sviluppo di un progetto software in gruppo o anche individualmente ed un orale con domande e discussione sul progetto e su alcuni argomenti del corso. Valutazione nel range 0-20.

Valutazione dell'attività di laboratorio, molto importante in quanto gli studenti potranno imparare ad usare diversi strumenti e tool. Valutazione nel range 0-5

Task individule in aula. Valutazione nel range 0-2

Orale. Valutazione nel range 0-5

Orario di ricevimento

su appuntamento via email

Sustainable Development Goals