

## **Ingegneria del Software (Ingegneria delle Telecomunicazioni)**

*Prof. Domenico Ursino*

Il corso intende fornire gli strumenti teorici e pratici per la gestione di progetti software di notevoli dimensioni. In particolare, verranno trattate le tematiche relative al ciclo di vita dei progetti software, alla raccolta e alla rappresentazione dei requisiti, alla loro analisi, alla progettazione dei sistemi software e, in parte, alla loro implementazione. Nell'ambito della progettazione dei sistemi software un'enfasi particolare verrà data ai pattern. Il corso si concluderà con una parte riguardante il controllo della qualità dei progetti software, la pianificazione dei progetti stessi anche dal punto di vista economico e delle risorse umane e, infine, il testing e la validazione dei sistemi realizzati. Il linguaggio di modellazione di riferimento è UML mentre la metodologia di gestione dei progetti di riferimento è lo Unified Process.

### **Programma del Corso**

#### **Introduzione (0.5 crediti)**

##### ***Unified Modelling Language***

Che cos'è UML, una breve storia di UML, caratteristiche generali di UML, la Model Driven architecture, la Struttura di UML.

##### ***Unified Process***

Introduzione a UP, storia di UP, UP e RUP, istanziazione di UP su un progetto, assiomi di UP, iterazioni e flussi di lavoro, struttura di UP, le fasi di UP.

#### **Gestione dei Requisiti (0.5 crediti)**

##### ***Il flusso di lavoro dei requisiti***

Introduzione, requisiti del software: meta-modello, definizione dei requisiti, individuazione dei requisiti.

##### ***Modellazione dei casi d'uso***

Introduzione, individuazione degli attori e dei casi d'uso, descrizione di un caso d'uso, mapping dei requisiti, applicazione della modellazione dei casi d'uso, generalizzazione tra attori, generalizzazione tra casi d'uso, estensione e inclusione tra casi d'uso.

#### **Analisi (1.5 crediti)**

##### ***Il flusso di lavoro dell'analisi***

Introduzione, il modello di analisi.

##### ***Classi e oggetti***

Cosa sono gli oggetti, notazione UML per gli oggetti, cosa sono le classi, notazione UML per le classi, ambito, creazione e distruzione degli oggetti.

##### ***Individuazione delle classi di analisi***

Analisi di un caso d'uso, cosa sono le classi di analisi, individuazione delle classi, costruzione di una prima bozza di modello di analisi.

##### ***Relazioni***

Cos'è una relazione, cos'è un collegamento, cos'è un'associazione.

##### ***Ereditarietà e polimorfismo***

Generalizzazione, ereditarietà di classe, polimorfismo, esempi di polimorfismo, generalizzazione avanzata.

### ***Package di analisi***

Che cos'è un package, package e spazi dei nomi, package annidati, dipendenze tra package, generalizzazione tra package, analisi dell'architettura.

### ***Realizzazione dei casi d'uso***

Analisi di un caso d'uso, realizzazione dei casi d'uso, interazioni, linee di vita, messaggi, diagrammi di interazione, diagrammi di sequenza, diagrammi di comunicazione.

### ***Diagramma di attività***

Che cosa sono i diagrammi di attività, i diagrammi di attività e UP, attività, semantica delle attività, partizioni delle attività, nodi azione, nodi controllo, nodi oggetto, pin.

## **Progettazione (2 crediti)**

### ***Il flusso di lavoro della progettazione***

Introduzione, manufatti della progettazione: il meta-modello, progettazione dell'architettura.

### ***Classi di progettazione***

Progettazione di una classe, cosa sono le classi di progettazione, anatomia di una classe di progettazione, classi di progettazione ben formate, ereditarietà, template di classe, classi annidate.

### ***Raffinare le relazioni di analisi***

Relazioni di progettazione, aggregazione e composizione, semantica dell'aggregazione, semantica della composizione, raffinamento delle relazioni di analisi, relazioni reificate, esaminare la composizione con le classi strutturate.

### ***Interfacce e componenti***

Progettazione di un sottosistema, cos'è un'interfaccia, interfacce richieste e fornite, realizzazione di interfacce ed ereditarietà, porte, interfacce e sviluppo basato sui componenti, cos'è un componente, stereotipi dei componenti, sottosistemi, individuare le interfacce, progettare usando le interfacce, vantaggi e svantaggi delle interfacce.

### ***I Pattern***

Introduzione ai pattern, pattern architetturali, design pattern e idiomi, il significato dei design pattern, la storia dei design pattern, elementi caratterizzanti dei design pattern, il cluster dei pattern GoF, il pattern architetturale di layering, il design pattern Model-View-Controller, il pattern DAO, il pattern Abstract Factory, il pattern Builder, il pattern Factory Method, il pattern Singleton, il pattern Adapter, il pattern Composite, il pattern Façade, il pattern Proxy, il pattern Command, il pattern Observer, il pattern Strategy, il pattern Template Method.

### ***Progettazione della realizzazione dei casi d'uso***

Progettazione di un caso d'uso, progettazione della realizzazione di un caso d'uso, diagrammi di interazione di progettazione, modellazione della concorrenza, interazioni a livello di sottosistema, diagrammi di temporizzazione.

### ***Macchine a stati***

Introduzione alle macchine a stati, le macchine a stati e UP, diagrammi delle macchine a stati, stati, transizioni, eventi, stati compositi, stati della sotto-macchina, comunicazioni tra sotto-macchine, stati con memoria.

## **Implementazione (0.5 crediti)**

### ***Il flusso di lavoro dell'implementazione***

Introduzione, manufatti dell'implementazione: meta-modello, descrizione del flusso di lavoro dell'implementazione.

### ***Deployment***

Implementazione dell'architettura, il diagramma di deployment, nodi, manufatti, deployment.

### **Nozioni di Project Management (0.5 crediti)**

#### ***Ciclo di vita di un progetto e metodologie di processo***

La nozione di ciclo di vita, varie tipologie di cicli di vita, valutazione dei rischi, metodologie di processo.

#### ***Management, lavoro di squadra e controllo della qualità***

Management, lavoro di squadra, leadership, controllo della qualità.

### ***Il software come prodotto***

Premessa, rapporti tra committente e sviluppatore, il mese-uomo, pianificazione delle risorse, costo del software, metriche software, tecniche di stima dei costi del software, diagrammi di Gantt, configuration management.

### ***Verifica, validazione e test***

Premessa, revisione di qualità, verifica, validazione, testing, revisioni e ispezione del codice.

### **Introduzione ad Enterprise Architect (0.5 crediti)**

#### ***Enterprise Architect***

Principi e filosofia di funzionamento, costruzione dei progetti, l'interfaccia grafica, stesura dei vari diagrammi UML.

### **Modalità dell'esame**

L'esame consiste in una prova progettuale e in una prova orale. La prova progettuale è volta a mettere in pratica tutte le nozioni via via acquisite durante il corso; in particolare, sarà richiesto allo studente di portare avanti la specifica e l'analisi dei requisiti, la progettazione e il deployment di un progetto software relativo ad una realtà di interesse. A tal fine verrà utilizzato anche Enterprise Architect. La tesina simulerà, pertanto, un vero e proprio progetto aziendale da condurre in team attraverso i comuni strumenti software utilizzati a tale scopo. La prova orale ha lo scopo di discutere la prova progettuale e di verificare la comprensione della rimanente parte del programma.

### **Testi Consigliati**

- J. Arlow, I. Neustadt, "UML 2 e Unified Process – Analisi e progettazione Object-Oriented", Seconda Edizione, McGraw-Hill, 2006.
- P. Stevens, R. Pooley, "Usare UML", Seconda Edizione, Pearson Education, 2008
- C. Larman, "Applicare UML e i pattern", Pearson Education, 2005
- E. Damiani, M. Madravio, A. Bohm, "UML Pratico", Seconda Edizione, Pearson Education, 2007.
- M. Fowler, "UML distilled", Terza Edizione, Pearson Education, 2004.
- L. Baresi, L. Lavazza, M. Pianciamore, "Dall'idea al codice con UML 2", Pearson Education, 2006.