Linee guida per lo sviluppo del progetto per il corso di Ingegneria del Software I

Scopo

Il progetto serve a dimostrare che lo studente si sia esercitato con un processo per lo sviluppo di software complesso. Si raccomanda l'uso di un processo di sviluppo software iterativo (s raccomanda il lo Unified Process – UP) che permetta di prendere confidenza e esercitarsi con tutte le principale fasi dello sviluppo senza dover produrre un software pienamente funzionante. È importante notare che primo fattore di valutazione non sarà la qualità del codice finale implementato. Lo studente dovrà invece dimostrare di aver compreso le caratteristiche del processo UP e dei corrispondenti documenti e modelli. In particolare sono particolarmente importanti le definizioni dei diagrammi d'uso, il modello concettuale, il diagramma delle classi ed eventuali diagrammi di interazione. Sarà altresì rilevante la spiegazione delle scelte prese in fase di progetto.
Per riferimento ricordo che l'obiettivo è principalmente rivolto alla comprensione delle azioni da intraprendere durante le prime iterazioni di un processo iterativo. In particolare con riferimento allo Unified Process si può dire che viene richiesto di procedere con la fase di Inception e con una iterazione della fase di Elaboration.

Svolgimento

Primo passo nello sviluppo del progetto sarà uno studio iniziale di fattibilità e la definizione di un numero ragionevole di Use Cases che possano essere considerati una sorta di nucleo centrale per il sistema oggetto dell'analisi. L'identificazione degli Use Case dovrà essere principalmente derivata dal testo del progetto assegnato al gruppo cercando di immedesimarsi in un possibile richiedente del corrispondente sistema. Riguardo il formato degli UC si consiglia caldamente di utilizzare quello visto a lezione principalmente derivato dal modello proposto da Alistair Cockbourn.

Successivamente alla fase di identificazione degli UC il gruppo dovrà definire il modello concettuale. Ricordo che questo modello è certamente tra i più importanti tra quelli necessari allo sviluppo di un buon sistema software. Le techniche da utilizzare nella definizione del modello concettuale ed identificazione delle classi concettuali sono quelle viste a lezione (i.e. category list e textual analysis). Ricordo che una classe concettuale non contiene né definizioni di metodi né di attributi.

Definito il modello concettuale si dovrà procedere alla identificazione delle classi e del modello di design del sistema. Nello svolgimento di questa fase si dovranno identificare e mappare alcune classi del modello concettuale in classi di design altre classi del modello concettuale potranno invece divenire attributi di altre classi. Infine alcune classi del modello concettuale non saranno considerate come rilevanti per la specifica implementazione. Successivamente all'identificazione delle classi del modello di design si dovrà procedere all'assegnamento delle responsabilità alle varie classi. Assegnare le responsabilità vuol in un certo senso dire identificare i metodi che faranno parte della definizione delle varie classi. Tale assegnamento sarà agevolato cercando di riprodurre l'esecuzione del task identificato da uno UC attraverso la collaborazione delle classi di design. In generale tale interazione può essere rappresentata attraverso un diagramma di interazione UML quale ad esempio un Sequence Diagram (SD). È evidente che uno stesso UC identifica più di una interazione possibile oltre a quella in particolare definita dal principale scenario di successo.

Definito il modello di design di basso livello si procederà allo sviluppo delle classi identificate in accordo a quanto

specificato in termini di attributi e metodi per ognuna delle classi. È possibile per le varie classi e metodi definire ulteriori modelli quali ad esempio Activity e State diagram, ma questo non è richiesto. Eventuali modellazioni di questo tipo saranno considerate come un extra.

Nella definizione del design si suggerisce di considerare possibili usi di uno dei Design Pattern visti a lezione. Questo potrebbe semplificare la fase di codifica e nel caso potrebbe aiutare nell'identificazione di ulteriori classi concettuali. Il non uso dei Design Patterns non sarà comunque, ed in nessun caso, considerato come un fattore negativo per il progetto.

La fase di codifica dovrà essere svolta preferibilmente utilizzando il linguaggio di programmazione Java.

Si consiglia di associare alla fase di sviluppo del codice una fase di definizione di casi di test di unità (dove per unità si intende metodi o al più classi), definiti utilizzando il framework per il testing JUnit (la versione 3.8 è più facile da utilizzare della 4.*, ultima scaricabile dal sito www.junit.org, soprattutto per chi non fosse esperto di Java e delle nuove caratteristiche del linguaggio quali ad esempio metadata e generics). La presenza di casi di test nel documento di progettom, codificati secondo il formato specificato da JUnit, sarà considerata estremamente positiva.

Lo studente, al fine di documentare il proprio lavoro anche se non previsto da una documentazione classica per la descrizione di un processo UP, è assolutamente libero di riportare spiegazioni ulteriori alle decisioni prese durante il progetto, descrivendo ad esempio perchè si ritengono le scelte corrette o descrivendo come i concetti visti a lezione sono stati applicati nei vari momenti dello sviluppo.

Buon lavoro!

Per eventuali ulteriori chiarimenti: andrea.polini@isti.cnr.it
oppure scrivete ed accedete al forum!!

JUnit in Eclipse

Le due figure seguenti vi mostrano dove trovate il supporto a JUnit dentro Eclipse. Attenzione ricordatevi di aggiungere la libreria di JUnit (3.8.* o 4.*) al build path del progetto.