

## Scelta del processo di sviluppo

Il processo di sviluppo da utilizzare viene scelto dopo un'analisi preliminare dei requisiti del software.

Nello specifico la nostra analisi preliminare ha considerato le richieste del committente disponibili su (<http://courses.web.cs.unibo.it/cgi-bin/twiki/bin/viewauth/ArchitettureSoftware/ProgettoDelCorso> v0,1) ed ha concluso che i requisiti del software saranno molto stabili, e quasi sicuramente non cambieranno le richieste del committente durante le fasi di sviluppo.

I processi di sviluppo sono stati elencati, e per ognuno vengono effettuate le opportune considerazioni.

- **Processo a cascata**

Un processo che prevede, analisi dei requisiti, progettazione, codifica, e testing tutti in successione.

- Pro: semplice da seguire ed effettuare.
- Contro: Molto dispendioso correggere errori rilevati nelle prime fasi di sviluppo.

Non utilizzeremo questo modello poiché prevediamo di inserire involontariamente errori nella fase di progettazione, e risulterebbe poi troppo costoso in termini temporali rimediare a questi.

- **Processo iterativo-incrementale**

Processo che consiste di varie iterazioni con fase di analisi, progettazione, implementazione e testing. Ogni iterazione ha come risultato finale un prodotto testato con un numero di funzionalità aumentato rispetto all'iterazione precedente.

- Pro: è più semplice risolvere errori introdotti nelle fasi iniziali del processo; il testing è introdotto in fasi precedenti rispetto al modello a cascata.
- Contro: più complesso pianificare il lavoro.

Decidiamo di non adottare questo processo poiché dall'analisi preliminare di evinchi che il nostro software abbia requisiti molto stabili, e risulterebbe inutilmente oneroso spezzettare le attività di analisi dei requisiti e design.

- **Processo a Spirale**

Processo di tipo iterativo dove ogni iterazione comprende analisi, design, sviluppo e testing; inoltre utilizza molto la pianificazione e l'analisi del rischio per tenere sotto controllo il processo.

Scegliamo di non adottare questo processo poiché l'analisi preliminare del software, ritiene non rilevante un'adeguata pianificazione ed un'attenta analisi del rischio ad ogni iterazione.

- **Processo a consegne programmate**

Processo con fasi iniziali di Analisi dei requisiti e progettazione, seguito da varie iterazioni di implementazione e testing. Viene programmato in anticipo il lavoro da svolgere in ogni iterazione.

- Pro: il testing ad ogni iterazione permette di rilevare in anticipo eventuali errori. L'analisi ed il design non sono ripetuti in più fasi.

Non adotteremo questo processo perché non ci riteniamo in grado di programmare adeguatamente il lavoro da svolgere nelle varie iterazioni.

- **Processi Agili**

Scegliamo di non utilizzare questo processo poiché è pensato principalmente per adattarsi ai cambiamenti dei requisiti; avendo previsto che il nostro software abbia requisiti molto stabili risulterebbe troppo costoso adottare un processo agile.

- **V-Model**

Processo a cascata, in cui a lavoro ultimato viene testato il risultato di ogni fase in ordine inverso.

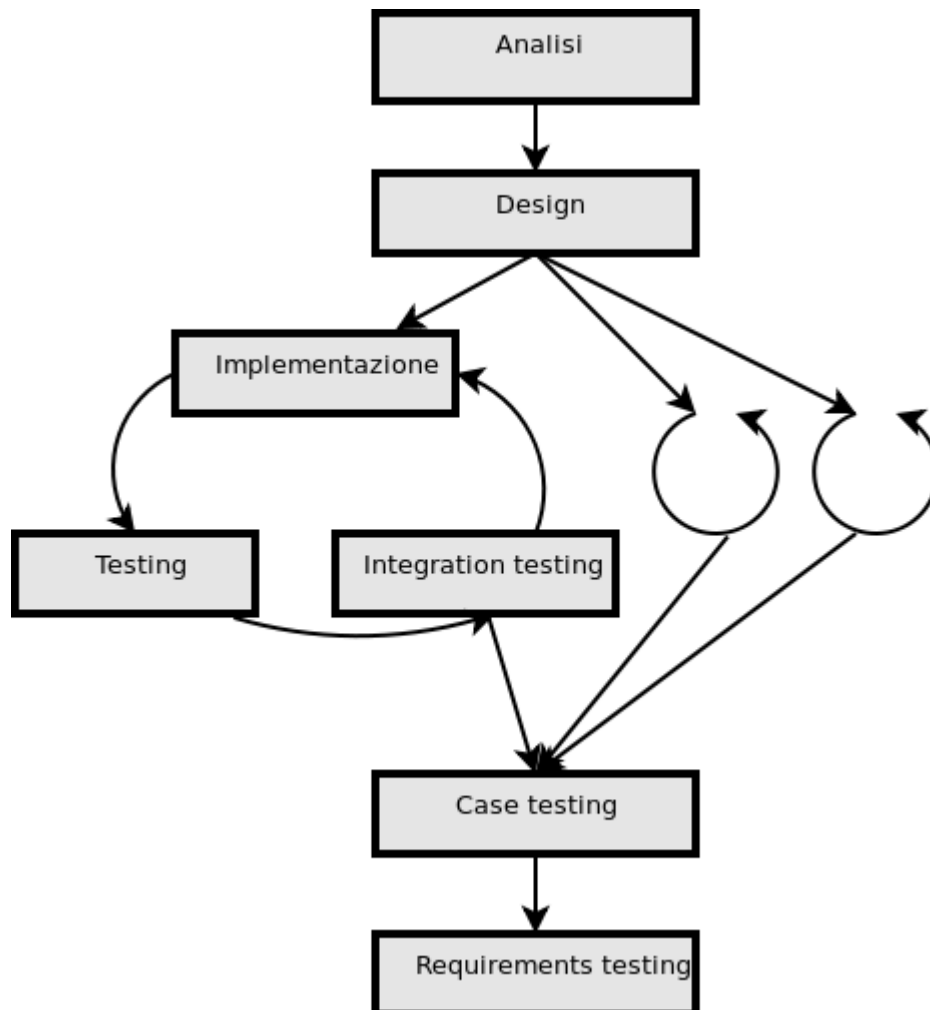
Non adottiamo questo processo poiché rileva gli errori introdotti nella fase di analisi e progettazione solo alla fine del processo, quando risulterebbe troppo oneroso correggerli.

- **RUP**

- Pro: enfasi su una documentazione accurata; risolve in modo pro-attivo i problemi legati al rischio di progetto; alto riuso di componenti,
- Contro: prevede che il team sia competente nel dominio del software.

Non adottiamo questo processo poiché presuppone che il team sia esperto nel dominio del software da realizzare.

In seguito alle considerazioni precedenti scegliamo di adottare un nostro processo di sviluppo che descriviamo qui di seguito.



Analizziamo le fasi singolarmente:

- **Analisi**

Durante questa fase saranno specificati tutti i requisiti che il software deve soddisfare. Questa fase produrrà un documento Specifica dei Requisiti Software visibile come allegato nel file SRS.pdf

- **Design**

In questa fase sarà definita l'architettura del sistema. Successivamente verrà definito un design ad un livello di dettaglio tale da permettere lo sviluppo nelle fasi successive. In particolare saranno definite le interfacce tra i componenti del software.

Tutto il gruppo deve partecipare attivamente in questa fase.

Se in questa fase emergeranno errori relativi all'analisi dei requisiti, modificheremo immediatamente i requisiti per poi modificare di conseguenza il design. Questa fase produrrà un documento Architettura e Design visibile come allegato nel file Architettura\_e\_Design.pdf

- **Iterazione di sviluppo**

Il software verrà partizionato in 3 parti ognuna della quali sarà sviluppata da un diverso team di sviluppo; nel nostro caso ogni team è composto da una sola persona. Ogni team effettuerà delle iterazioni fino a raggiungere la completa implementazione delle funzionalità che lo riguardano. Le iterazioni non sono pianificate né in senso temporale, né come lavoro da svolgere.

Ad ogni iterazione lo sviluppatore esegue tre fasi:

- **Implementazione**

Lo sviluppatore implementa un sottoinsieme della funzionalità che deve sviluppare. Esso è libero di effettuare scelte implementative non vincolate dalla fase di design.

- **Testing**

Lo sviluppatore testa la parte di funzionalità prodotta nell'iterazione corrente; esso non ha vincoli su come effettuare questo test.

- **Integration Testing**

Lo sviluppatore integra ciò che ha creato nell'iterazione corrente con la parte di software che è già stata testata nelle iterazioni precedenti da lui e dagli altri team di sviluppo. Inoltre effettua un test per verificare che le interazioni fra i componenti dopo l'integrazione siano corrette. La scelta di questo tipo di test è lasciata agli sviluppatori.

La documentazione da produrre in questa fase è lasciata al libero arbitrio degli sviluppatori.

Quando durante un iterazione un team identifica errori o mancanze derivanti dalle fasi di analisi o design, tutto il gruppo si riunisce per risolvere il problema; ove occorre saranno immediatamente modificati la Specifica dei Requisiti Software, l'architettura, ed il design. La correzione del problema sarà poi affidata ad uno dei tre team di sviluppo.

- **Case testing**

Dopo che tutti i team di sviluppo hanno terminato l'implementazione, vengono descritti ed eseguiti i casi di test che il software deve superare per passare alla fase successiva. Quando saranno rilevati errori sarà responsabilità del team che ha sviluppato la parte interessata risolverli.

- **Requirements Testing**

Dopo che tutti i casi di test sono stati superati correttamente, si verificherà che il software rispetti tutte le caratteristiche descritti nella Specifica dei Requisiti Software, comprese eventuali modifiche ai requisiti introdotti nelle fasi di sviluppo. In caso di problemi questi saranno risolti dal team di

sviluppo designato, ed il software dovrà ripetere le fasi di Case Testing e Requirements Testing. Il software che sia approvato da questa fase, sarà considerato completato.

Vantaggi del processo scelto:

- Tutto il gruppo parteciperà alla fase di design, dunque ognuno di noi sarà dunque consapevole dell'intero design di sistema.
- Ogni membro del gruppo dovrà conoscere l'implementazione solo della parte di software di sua competenza.
- Le iterazioni non pianificate permettono un approccio più adattativo agli imprevisti dei singoli team di sviluppo, inoltre un approccio non pianificato in fasi di sviluppo permette di mitigare gli effetti derivanti dall'individuazione di errori risolvendoli immediatamente.
- Il processo non suppone a priori che i membri del gruppo debbano essere esperti nel particolare dominio del software, né nella sola porzione di software che dovranno implementare.
- I Case Testing sono sviluppati dopo l'implementazione, quando la conoscenza sul software del gruppo è migliore; questo porterà a Test Case più accurati e precisi.
- Le fasi di Analisi e Design sono uniche e non ripetute, e questo è coerente con la stabilità dei requisiti prevista dall'analisi preliminare.

Svantaggi:

- Gli errori introdotti in fasi di Analisi e Design possono essere identificati molto tardi nello sviluppo del software, e la loro correzione richiede la presenza di tutto il gruppo.