Breve introduzione su Verifica e Validazione, e Manutenzione del Software

Andrea Polini

Ingegneria del Software
Corso di Laurea in Informatica

Verifica e Validazione - generalità

Dunque il problema a questo punto è capire se il sistema risolve effettivamente i problemi per cui è stato concepito e se lo fa "correttamente"

Are we building the right product?

vs.

Are we building the product right?

Verifica: il prodotto che stiamo sviluppando è corretto?

Validazione: stiamo sviluppando il corretto prodotto?

Verifica e Validazione - generalità

Dunque il problema a questo punto è capire se il sistema risolve effettivamente i problemi per cui è stato concepito e se lo fa "correttamente"

Are we building the right product?

vs.

Are we building the product right?

Verifica: il prodotto che stiamo sviluppando è corretto?

Validazione: stiamo sviluppando il corretto prodotto?

V&V

In qualche modo le attività di V&V intendono “rassicurare” l’utilizzatore che il sistema è “adatto allo scopo”

Livello di fiducia che si intende fornire non deve considerarsi un concetto assoluto ma dipende da diversi fattori:

- Funzione del software all’interno dell’organizzazione
- Attese dell’utente
- Politiche di mercato

Approaches to V&V

Approcci statici: analisi statica dei sorgenti e altri documenti di progetto (ispezione e revisione, verifica formale) - utili a verificare conformità e coerenza nelle fasi dello sviluppo (non permettono validazione)

Approcci dinamici: testing - prevede esecuzione del software o prototipi al fine di scoprire difetti (validazione e verifica). Tecnica di gran lunga più utilizzata per la verifica e validazione del software

E poi?

- Debugging: Attività che si occupa della localizzazione del guasto che ha generato un fallimento.
- Rivalidazione e Regression Testing

Testing

Il testing del software prevede l'esecuzione di alcuni "esperimenti" in un ambiente controllato al fine di poter acquisire sufficiente fiducia sul suo funzionamento. Testing riguarda tipicamente proprietà funzionali ma può riguardare anche caratteristiche extra-funzionali.

Due obiettivi differenti:

- Dimostrare che il sistema risponde alle esigenze (almeno un test per ogni requisito)
- Scoprire guasti (cercare di far manifestare tutti i possibili guasti)

Il testing non può dimostrare l'assenza di guasti ma solo la loro presenza

E.W. Dijkstra

Testing

Il testing del software prevede l'esecuzione di alcuni "esperimenti" in un ambiente controllato al fine di poter acquisire sufficiente fiducia sul suo funzionamento. Testing riguarda tipicamente proprietà funzionali ma può riguardare anche caratteristiche extra-funzionali.

Due obiettivi differenti:

- Dimostrare che il sistema risponde alle esigenze (almeno un test per ogni requisito)
- Scoprire guasti (cercare di far manifestare tutti i possibili guasti)

Il testing non può dimostrare l'assenza di guasti ma solo la loro presenza

E.W. Dijkstra

Genesi dei Fallimenti

Errore: si riferisce al processo logico che porta ad una non corrispondenza del software rispetto a quanto necessario come risultato dell'attività compiuta da un progettista/programmatore (il quale può commettere un errore)

Guasto: parte del progetto che in qualche modo contiene la codifica dell'errore

Fallimento: manifestazione del guasto con osservazione di un funzionamento scorretto

Unit testing

Riguarda il testing di elementi software in isolamento, tramite uso di stub e/o mock

Strategie di definizione dei casi di test:

- Flusso dei dati
- Flusso del controllo
- Espressioni condizionali
- Dati di input
- Cicli

Unit testing in ambito OO

Unit testing

Riguarda il testing di elementi software in isolamento, tramite uso di stub e/o mock

Strategie di definizione dei casi di test:

- Flusso dei dati
- Flusso del controllo
- Espressioni condizionali
- Dati di input
- Cicli

Unit testing in ambito OO

Unit testing

Riguarda il testing di elementi software in isolamento, tramite uso di stub e/o mock

Strategie di definizione dei casi di test:

- Flusso dei dati
- Flusso del controllo
- Espressioni condizionali
- Dati di input
- Cicli

Unit testing in ambito OO

Integration testing

Strategie di Integrazione:

- top-down
- bottom-up
- big-bang

Regression Testing

Integration testing in ambito OO

Integration testing

Strategie di Integrazione:

- top-down
- bottom-up
- big-bang

Regression Testing

Integration testing in ambito OO

System Testing

System testing di natura funzionale considera quei requisiti generali del sistema

Caratteristiche extrafunzionali generalmente sperimentabili affidabilmente soltanto a questo livello (proprietà non composizionali):

- usability
- security testing
- stress testing
- performance testing
- ...

System Testing

System testing di natura funzionale considera quei requisiti generali del sistema

Caratteristiche extrafunzionali generalmente sperimentabili affidabilmente soltanto a questo livello (proprietà non composizionali):

- usability
- security testing
- stress testing
- performance testing
- ...

Validation Testing

Test che cercano di valutare la soddisfazione del cliente e l'aderenza alle sue esigenze.

Tecniche principali:

- alfa testing
- beta testing

Debugging

Riguarda le attività di localizzazione e risoluzione dei bug.
Reso difficile da diversi fattori:

- sintomo e causa possono essere “distanti”
- sintomo si può manifestare sporadicamente
- il sintomo potrebbe non esser dovuto a guasti logici nel programma
- il sintomo può essere dovuto a temporizzazione ed interleaving particolari
- difficoltà di riproduzione delle condizioni di errore
- sintomo dovuto a diversi task in esecuzione su diversi processori

Una volta individuato il guasto dovrà essere risolto.
Attività delicata in quanto ogni nuova riga di codice può introdurre nuovi guasti

Evoluzione del Software - generalità

Cosa, quando, come, perchè?

Note salienti:

- Inevitabilità del cambiamento di un sistema software
- Investimenti nel software riguardano per larga parte la gestione di software esistente
- Più nuovi requisiti che riparazione da guasti
- Processo di evoluzione del software - come gestiamo l'evoluzione - e nuove fasi (comprensione del software)

Qualche nota sui costi

In generale il costo dell'evoluzione copre fino al 50% dell'intero costo di un prodotto software. In alcuni casi può addirittura arrivare all'80% dell'intero costo.

Miglioramenti anche lievi ai fini della fase di evoluzione possono portare a risparmi considerevoli

e.g. comprensione del software migliorata con buona documentazione

Testing 50-60% dello sviluppo, Evoluzione 80% dei costi totali!!
Dunque sembrerebbe che le fasi di "specifica, progettazione, implementazione" siano le fasi più facili!
In realtà riduzione di queste percentuali passa per aumento dei costi e migliore gestione delle fasi iniziali dello sviluppo!

Qualche nota sui costi

In generale il costo dell'evoluzione copre fino al 50% dell'intero costo di un prodotto software. In alcuni casi può addirittura arrivare all'80% dell'intero costo.

Miglioramenti anche lievi ai fini della fase di evoluzione possono portare a risparmi considerevoli

e.g. comprensione del software migliorata con buona documentazione

Testing 50-60% dello sviluppo, Evoluzione 80% dei costi totali!!
Dunque sembrerebbe che le fasi di "specifica, progettazione, implementazione" siano le fasi più facili!
In realtà riduzione di queste percentuali passa per aumento dei costi e migliore gestione delle fasi iniziali dello sviluppo!