12. Evoluzione del Software

Andrea Polini

Ingegneria del Software
Corso di Laurea in Informatica

Evoluzione del Software - generalità

Cosa, quando, come, perchè?

Note salienti:

- Inevitabilità del cambiamento di un sistema software
- Invesitmenti 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)

Leggi di Lehman

Studio statistico condotto su molti sistemi di grosse dimensioni hanno portato alla formulazione di alcune “leggi” sulla dinamica evolutiva del software:

1. Cambiamento continuo - sempre nuove richiste di miglioramento/estensione
2. Complessità crescente - la struttura tende a essere più compressa. Richiede attenta pianificazione e ristrutturazione
3. Evoluzione autoregolata - progammi di grosse dimensioni hanno caratteristiche di sviluppo intrinseche

Leggi di Lehman

...continua

1. Stabilità organizzativa - organizzazioni lavorano spesso in situazioni di saturazione dunque modifiche allo stuff allocato non producono differenze sostanziali allo sviluppo
2. Conservazione della familiarità - quantità di interventi rimane in generale costante
3. Continuo incremento delle funzionalità fornite
4. Riduzione della qualità
5. Processo con feedback

Mantenimento del software

Tre differenti tipi di interventi sul software:

- Correzione - codifica o design
- Adattamento - nuovi linguaggi, piattaforme
- Perfettivo - nuovi requisiti

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

Giustificazioni sull'alto costo dell'evoluzione

Le fasi di mantenimento del software sono particolarmente costose a causa di alcune "non buone" pratiche di gestione:

- Stabilità dei team di sviluppo
- Responsabilità contrattuali
- Esperienza del team di manutenzione
- "Età del software e struttura"

Come è possibile fare predizioni?

Costo alto rende desiderabili lo sviluppo di "meccanismi" di predizione dei possibili costi e della complessità. Questo permette di giudicare prima che le modifiche vengano effettuate.

La possibilità di predirre quanto un sistema tenderà a richiedere fasi di evoluzioni è certamente correlata alla comprensione delle relazioni tra sistema ed ambiente. Questa relazione è influenzata da:

- Numero e complessità delle interfacce
- Numero dei requisiti di sistema da considerarsi volatili
- I processi di business in cui il sistema è utilizzato

Predirre influenze e costi di un passo di evoluzione

Esperimenti hanno mostrato che più il software è complesso più costa mantenerlo - Wow!!

Esistono molte misure di complessità del software (e.g. ciclomatica). Più sono i componenti del sistema influenzati da una modifica e maggiore è la loro complessità maggiori saranno i costi di evoluzione.

Meno ovvio è che sperimentalmente si è notato che la complessità tende ad accumularsi in pochi componenti di un sistema software. Saranno questi a richiedere i maggiori costi di gestione (ancora una volta Pareto Law).

Ridurre complessità dei componenti

Misure in itinere

Esistono misure che permettono predizione di fasi di evoluzione successive a partire dallo “storico” di quelle precedenti

- Numero di richieste per interventi correttivi
- Tempo medio per l’analisi di impatto
- Tempo medio per implementare una richiesta di modifica
- Andamento del numero delle richieste di modifica pendenti

Il processo di evoluzione

Come ogni fase dello sviluppo si deve prevedere un processo per poter mettere in atto le differenti azioni.
Si sono identificate le seguenti attività in generale organizzate in un processo ciclico attivato da un richiesta di modifica:

- Analisi dell'impatto della modifica richiesta
- Pianificazione della release
- Implementazione della modifica
- Release del sistema

Interventi urgenti

Spesso gli interventi sul software non subiscono una pianificazione così dettagliata. Molto più spesso gli interventi sono fatti in condizioni di urgenza:

- Guasto particolarmente pericoloso da impedire uso normale
- modifiche all'ambiente che provocano l'impossibilità nell'uso
- nuovi competitori o modifica a sistema legislativo

Interventi in urgenza saltano fasi di pianificazione ed analisi ma cercano di arrivare allo sviluppo di una "patch" nel più breve tempo possibile.

Interventi urgenti

...continua

Conseguenze:

- Software e documentazione disallineate
- Ridotta strutturazione del software
- costi successivi di gestione crescenti

Visione idealizzata richiede di applicare successivamente i passi standard del processo evolutivo per lo sviluppo di una patch “pianificata”. Elimina problemi suddetti ma approccio raramente applicato vista generale emersione di nuove modifiche più importanti.

Re-ingegnerizzazione del software

Evoluzione rende software complesso e sembre piè difficile da mantenere (degrado dell'organizzazione complessiva, disallineamento di specifiche e codice)

Re-ingegnerizzazione è un possibile approccio volto a ridurre problemi di manutenzione

Di cosa si tratta?
Si lavora sul codice esistente al fine di migliorarne l'organizzazione, la documentazione, la struttura dei dati. Oppure si procede alla codifica con diverso linguaggio o su differente tecnologia/piattaforma.
Tipicamente **non cambia** l'architettura e le funzionalità offerte

Vantaggi della reingegnerizzazione

Rispetto ad altri approcci all'evoluzione del software:

- Rishio ridotto - procedere ad una nuova fase di sviluppo è molto costoso e può portare con maggiore facilità a mancare gli obiettivi
- Costi ridotti - in generale il costo dello sviluppo ex-novo è estremamente più alto

Re-ingegnerizzare....come

Distinzione fondamentale tra re-ingegnerizzazione e sviluppo è il punto di partenza e definizione della specifica.

Passi di un ipotetico processo di re-ingegnerizzazione:

- Traduzione del codice
- Reverse engineering
- Miglioramento della struttura del programma
- Modularizzazione
- Riorganizzazione della struttura dei dati

Fattori di costo della re-ingegnerizzazione

- Qualità del software di partenza
- Supporto di tool automatici
- Influenza delle modifiche apportate ad dati
- disponibilità di staff esperto

Sistema reingegnerizzato non è in generale facilmente gestibile come uno sviluppo ex-novo

Evoluzione di sistemi Legacy

Interventi sul software vanno sempre valutati in base alla reale necessità del software per gli scopi dell'organizzazione. Esistono differenti opzioni:

- Dismisssione del sistema
- Continuare con regolare manutenzione
- Re-ingegnerizzare il sistema
- Rimpiazzare parti del sistema con nuove componenti

Classi di sistemi Legacy

Analisi di sistemi legacy porta ad identificare 4 classi di sistemi:

- Bassa qualità e basso valore per l'organizzazione
- Bassa qualità e alto valore per l'organizzazione
- Alta qualità e basso valore per l'organizzazione
- Alta qualità e alto valore per l'organizzazione

In generale identificazione del valore di business può essere condotto tramite interviste agli utilizzatori e cercando di capire:

- Uso del sistema
- Il processo di business che il sistema supporta
- Affidabilità del sistema
- Output del sistema

Valutazione tecnica

Fattori per la valutazione tecnica appartengono a due classi principali. Valutazione dell'ambiente operativo e valutazione dell'applicazione

Ambiente operativo:

- Stabilità del fornitore
- Numero di fallimenti
- Età
- Performance
- Requisiti del supporto
- Interoperabilità

Valutazione tecnica

Applicazione:

- Comprensibilità
- Documentazione
- Data
- Performance
- Linguaggio di programmazione usato
- Gestione della configurazione
- Dati di test
- Conoscenze del personale per la fase di manutenzione