

Definizioni

- Manutenzione
 - Il processo di introduzione di modifiche ad un prodotto software dopo la sua consegna al cliente
- Evoluzione
 - Spesso usato con lo stesso significato di manutenzione, oppure
 - Singolo passo di un processo di manutenzione che prevede: evoluzione, rilascio di patch, rimozione sistema

(Ing. E. Tramontana - Evoluzione, metriche - 9 -Giu-06) 1

Dati statistici

- I costi di manutenzione rappresentano il 67-80% dei costi del software
- I cambiamenti si possono raggruppare in 4 categorie
 - **Correttivi** - rimozione errori (17%)
 - **Adattativi** - aggiustamenti per un nuovo ambiente (18%)
 - **Perfettivi** - miglioramento e aggiunta di funzionalità (60%)
 - **Preventivi** - modifiche interne per prevenire problemi (5%)
- Incorporare nuove funzionalità è la porzione maggiore di modifiche
 - E' buona pratica anticipare i cambiamenti a design time (tramite parametrizzazione, incapsulamento, etc.)

(Ing. E. Tramontana - Evoluzione, metriche - 9 -Giu-06) 2

Dinamiche di evoluzione

- Sono i processi di cambiamento di un sistema
- Si basano su studi empirici
 - Effettuati da Lehman e Belady (dal 1968 e confermati da studi recenti)
 - Su sistemi software di grandi dimensioni sviluppati da grandi aziende

(Ing. E. Tramontana - Evoluzione, metriche - 9 -Giu-06) 3

Leggi di Lehman

- Cambiamento continuo [1974]
 - I sistemi hanno bisogno di essere continuamente adattati altrimenti diventano progressivamente meno soddisfacenti
- Aumento della complessità [1974]
 - Quando un sistema evolve, la sua struttura aumenta di complessità, a meno che del lavoro viene fatto per preservare o semplificare la sua struttura
- Auto-regolazione [1974]
 - Attributi come dimensione, intervallo tra release e numero di errori trovati in ciascuna release sono approssimativamente invariati
- Stabilità organizzativa [1978]
 - Durante la vita di un sistema il suo tasso di sviluppo è circa costante e indipendente dalle risorse impiegate per lo sviluppo

(Ing. E. Tramontana - Evoluzione, metriche - 9 -Giu-06) 4

Leggi di Lehman

- Conservazione di familiarità [1978]
 - In media, l'incremento di crescita di un sistema tende a rimanere costante o a diminuire
- Continua crescita [1978]
 - Il contenuto di funzioni di un sistema deve continuamente essere incrementato per mantenere la soddisfazione dell'utente
- Diminuzione della qualità [1994]
 - La qualità di un sistema diminuisce se non viene rigorosamente gestita ed adattata durante i cambiamenti

Applicabilità delle leggi

- Sono in generale applicabili a grandi sistemi sviluppati da grandi organizzazioni
- Non è chiaro come si adattano a
 - Piccoli prodotti
 - Prodotti che incorporano un certo numero di COTS
 - Piccole organizzazioni

Costo di manutenzione

- Fattori che contribuiscono al costo
 - Il costo è ridotto se il team di sviluppo è coinvolto nella manutenzione
 - Gli sviluppatori potrebbero non avere responsabilità contrattuali per la manutenzione, quindi non hanno incentivi a fare un design che può essere cambiato in futuro
 - La struttura del programma si degrada mano a mano che si introducono cambiamenti

Modelli di manutenzione

- Modelli di manutenzione
 - Quick-fix
 - Cambiamenti a livello del codice
 - Miglioramento iterativo
 - Cambiamenti fatti in base ad un'analisi del sistema esistente
 - Controllo della complessità e mantenimento del design
 - Riuso
 - Stabilire i requisiti per il nuovo sistema, riusingo il più possibile

Tipi di modifiche

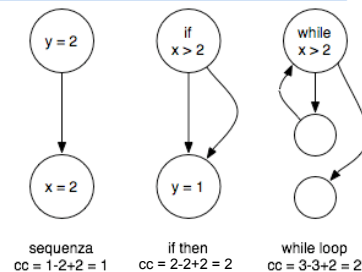
- Re-factoring o re-structuring
 - Processo di cambiamento del software che non altera il comportamento del codice ma migliora la struttura interna
 - Ovvero: prendere un sistema fatto male e modificarlo per ottenere una struttura ben fatta
- Reverse engineering
 - Analizzare un sistema per estrarre informazioni sul suo comportamento o sulla sua struttura
- Re-engineering
 - Alterare un sistema per ricostituirlo in un'altra forma

Metriche

- Una metrica definisce un set di misure per un sistema software
- Obiettivi dell'adozione di metriche
 - Monitorare il prodotto mentre si costruisce
 - Identificare i livelli per ciascuna metrica
 - Rimediare in caso i livelli non sono soddisfacenti
- Solo gli attributi interni possono essere misurati direttamente (es. dimensione), quelli esterni sono ricavati indirettamente

Metriche tradizionali

- Complessità ciclomatica (cc)
 - Valuta la complessità di un algoritmo
 - cc è il numero di test necessari per valutare esaurientemente l'algoritmo
 - Per una sequenza -> un solo test
 - Per una condizione -> due test
 - $cc = \text{numero di test} = \text{archi} - \text{nodi} + 2$
- Dimensione
 - LOC (lines of code), oppure NCNB (non comment non blank)
- Comment Percentage
 - Percentuale di commenti rispetto a LOC (consigliato 30%)



Metriche OO

- Suite di Chidamber & Kemerer o metriche CK
 - WMC (Weighted Methods per Class)
 - La somma delle complessità dei metodi per una classe
 - Se i metodi hanno pari complessità, WMC è il numero di metodi della classe
 - DIT (Depth of Inheritance Tree)
 - Massimo numero di livelli dalla classe alla radice della gerarchia (radice=0)
 - NOC (Number of Children of a Class)
 - Numero di sottoclassi di una classe della gerarchia
 - CBO (Coupling Between Object Classes)
 - Numero di classi con cui una classe interagisce
 - RFC (Response for a Class)
 - Numero di metodi eseguiti al ricevimento di un messaggio, cioè numero di metodi di una classe + numero di metodi invocati da ciascuno di essi
 - LCOM (Lack of Cohesion of Methods)
 - Per ogni campo della classe, si calcola la percentuale di metodi che usano tale campo; si mediano le percentuali e si sottrae dal 100%

Interpretazione metriche CK

- WMC
 - Alto valore di WMC indica grande lavoro di manutenzione, grande specificità della classe e quindi poca possibilità di riuso
- DIT
 - Più alto è DIT per una classe, più difficile è capirne il comportamento, data la grande quantità di metodi ereditati
 - Alto DIT indica maggiore complessità dell'intero sistema, ma anche maggiore riuso
- NOC
 - Indica l'influenza che una classe ha sul sistema
 - Maggiore è NOC, maggiore è il riuso

Interpretazione metriche CK

- CBO
 - Maggiore è CBO maggiore è la dipendenza di una classe da altre, quindi minore possibilità di riuso, maggiore difficoltà a comprendere, modificare e correggere la classe
- RFC
 - Alto RFC indica grande complessità, quindi difficoltà di comprensione e di testing
- LCOM
 - Alta coesione (basso LCOM) indica semplicità, elevata possibilità di riuso