Sviluppo di Software Applicativo

Metriche del software

Come misurare le diverse caratteristiche del software: dimensioni, qualità, impegno richiesto per lo sviluppo, ecc.

Ercole Colonese
IBM Global Services
Consulting Services

Ottobre 1999

Indice

Note sull'autore

Ercole Colonese è *Senior Consultant* in IBM e svolge l'attività nell'area dello sviluppo del software applicativo, compreso temi di qualità ed usabilità. La sua esperienza in questo campo è maturata in moltissimi anni di lavoro presso i laboratori di sviluppo software nazionali ed internazionali dell'IBM dove egli ha ricoperto ruoli tecnici e manageriali. Ha realizzato diversi sistemi di qualità aziendali certificati ISO9000 presso l'IBM e presso grandi clienti. Ha partecipato a molti progetti di reingegnerizzazione dei processi di sviluppo software ed alla costruzione di gruppi di lavoro in ottica di produttività, qualità e motivazione. Come docente, tiene regolarmente corsi sulla qualità del software ed i loro processi di sviluppo presso il Learning Center IBM.

email: ercole@coloneseit

Altri documenti dello stesso autore

Sono disponibili o in fase di preparazione altri documenti sul tema "Sviluppo di Software Applicativo", come *L'usabilità del software, dei siti Web, Un approccio integrato allo sviluppo del software, Il ciclo di vita del software, Le metriche del software, La misurazione dei costi del software, L'analisi dei requisiti, Il Capability Maturità Model (CMM), Standard di sviluppo del software, Il test efficace, La comunicazione efficace, Tecniche di revisione ed ispezioni del software, I prototipi leggeri, I gruppi di lavoro.*

Metriche del software

Come misurare le diverse caratteristiche del software: dimensione, qualità, impegno richiesto per lo sviluppo, ecc.

Le metriche sono unità di misura utilizzate per quantificare, misurare e valutare i diversi aspetti dello sviluppo del software. Esistono molte metriche. Qui sono presentate quelle comuni e più utili divise in categorie:

- Metriche relative ai prodotti software (pacchetti o applicazioni specifiche)
- Metriche relative ai processi per lo sviluppo del software
- Metriche relative alla qualità del software
- Metriche object-oriented

Alcune metriche sono dette di “base” ed altre “derivate” in quanto utilizzano, a loro volta, quelle di base.

Metriche relative al prodotto software

Dimensioni del prodotto software

Rappresenta le dimensioni del prodotto software. Tradizionalmente si è misurata in termini di migliaia di linee di codice sorgente (KLOCs). Da alcuni anni è stata introdotta una nuova misura, legata al numero di funzionalità offerte e quindi del valore che esso rappresenta per l'utente, svincolata dalle dimensioni del codice sviluppato. Questa seconda misurazione è espressa come numero di punti funzione (Function Points – FPs). E' utilizzato per misurare il valore del prodotto software e lo sforzo per realizzarlo.

Dimensione dell'applicazione = Numero di linee di codice, in migliaia (KLOCs)
Dimensione dell'applicazione = Numero di punti funzione (FPs)

Codice sorgente

Rappresenta le dimensioni del codice sorgente (di tutto il prodotto software oppure di quello sviluppato in una versione specifica). E' espresso in termini di numero di linee di codice, in migliaia (KLOCs) ed è utilizzato per dimensionare la produttività delle persone e da questa lo sforzo richiesto per sviluppare il prodotto.

Dimensione del codice sorgente = Numero di linee di codice (LOCs)

Codice eseguibile

Rappresenta le dimensioni del prodotto software compilato e pronto per essere eseguito. E' utilizzato per dimensionare l'occupazione di memoria (su disco e/o centrale) necessario ad ospitare il prodotto software. E' misurata in numero di bytes, in migliaia o milioni (KBytes, MBytes).

Dimensione del codice eseguibile = Numero di bytes (KBytes o MBytes)

Documentazione

Rappresenta le dimensioni della documentazione prodotta per il cliente (manuali per l'utente, altri manuali). E' misurata in termini di numero di pagine prodotte. E' utilizzata per valutare la produttività in fase di stesura dei manuali e per valutare lo sforzo richiesto.

Dimensione dei manuali per gli utenti = Numero totale di pagine (Pagine)

Metriche relative al processo di sviluppo

Durata del progetto

Rappresenta la durata del progetto espresso in numero di mesi o, per progetti di dimensioni contenute, in settimane. E' utilizzata per valutare l'effettiva disponibilità del prodotto realizzato.

Durata del progetto = Numero di mesi (o di settimane o di giorni)

Produttività

Rappresenta la produttività media delle risorse impiegate, cioè delle persone coinvolte, nelle diverse fasi del progetto. Si parla quindi di produttività media del progetto, oppure di produttività di analisi e disegno, oppure ancora di produttività di programmazione, ecc. E' misurata in termini di numero di linee di codice o di punti funzione sviluppati da una persona nell'unità di tempo stabilita (mese, settimana, giorno, ora). La produttività di una fase specifica è misurata, per esempio, in numero di linee di codice sviluppate per mese-uomo, oppure numero di casi di test eseguiti per giorno-uomo, oppure numero di pagine scritte per giorno-uomo. E'utilizzata per valutare lo sforzo richiesto per lo sviluppo del progetto a fronte delle sue dimensioni.

Produttività media = KLOCs / Mese uomo (oppure settimana uomo, o giorno uomo, o ora uomo)
Produttività media = FPs / Mese uomo (oppure settimana uomo, giorno uomo, ecc.)

Per le fasi specifiche di sviluppo:

Produttività di codifica = KLOCs / Mese uomo (oppure settimana uomo, o giorno uomo, ecc.)
Produttività di test = Casi di test / Mese uomo (oppure settimana uomo, o giorno uomo, ecc.)
Produttività di documentazione = Pagine / Mese uomo (oppure settimana uomo, ecc.)

Impegno richiesto per realizzare il progetto

Rappresenta l'impegno necessario per sviluppare il progetto tenendo conto delle sue dimensioni e della produttività media delle persone coinvolte. E' quindi calcolato come rapporto tra le dimensioni del progetto e produttività media. E' utilizzato per valutare il costo, in termini di risorse, per sviluppare il progetto.

Impegno = Dimensione / Produttività = Totale mese uomo (oppure settimane uomo, o giorni uomo, o ore uomo) richiesti per lo sviluppo del progetto

Numero di cambiamenti apportati

Rappresenta il numero di modifiche apportate al progetto in corso d'opera. Le modifiche possono riguardare i requisiti, le funzionalità, il disegno, il codice, i manuali. Sono generati dalla necessità di aggiungere un requisito nuovo oppure di modificare o cancellarne uno esistente. La necessità può derivare da una richiesta esplicita del committente oppure da una errata o incompleta interpretazione del fornitore. Un requisito nuovo (o modificato o cancellato) provoca, quasi sicuramente, una o più modifiche alle funzionalità, al disegno, al codice ed alla documentazione per l'utente. E' importante, quindi, misurare il numero di modifiche apportare in corso d'opera per valutare gli impatti sui tempi di realizzazione e sui costi del progetto.

Modifiche = Numero di modifiche ai requisiti (o funzionalità, o disegno, o codice, o manuali)

Numero di errori rilevati

Rappresenta il numero di errori rilevati nel prodotto durante le diverse fasi di sviluppo. Così si parla di numero di errori rilevati nel disegno, nel codice, nella documentazione, oppure nei casi di test, ecc. Gli errori sono rilevati sulla documentazione si riferiscono alle attività di revisione e ispezione tecnica, quelli di codice alle attività di test. Sono importanti in quanto permettono di calcolare l'efficacia del processo di revisione e di test e, da questa, la curva di rimozione degli errori durante lo sviluppo. Dalla curva di rimozione degli errori è possibile prevedere, con tecniche più o meno sofisticate, il tasso di errori residuo, ovvero il numero di errori che gli utenti troveranno in fase di esercizio del prodotto. Il numero di errori rilevati è normalizzato rispetto alle dimensioni del progetto.

Numero di errori = Numero di errori rilevati / Dimensioni del progetto(KLOCs o FPs)

Copertura dei test

Rappresenta il livello di copertura che i test eseguiti forniscono rispetto alle funzionalità offerte dal prodotto. E' misurato come rapporto tra le funzionalità effettivamente verificate dai casi di test eseguiti ed il numero totale delle funzioni disponibili nel prodotto ed è espresso in punti percentuale. Una buona copertura è pari al 100%. Può essere calcolata anche come rapporto tra il numero totale di casi di test eseguiti ed il numero totale delle funzionalità offerte dal prodotto. In questo caso il valore può superare il 100%. Questa seconda misurazione evidenzia che alcune funzioni sono testate con più di un caso di test, e deve essere sempre accompagnata dalla prima (copertura) che dimostra quante funzioni sono state testate.

Copertura del test (1) = Numero di funzioni testate / Numero totale di funzioni disponibili
Copertura del test (2) = Numero di casi di test eseguiti / Numero totale di funzioni disponibili

Efficacia dei test

Rappresenta la capacità di rilevare errori nel prodotto tramite le attività di test. E' misurata come rapporto tra il numero di errori rilevati ed il numero di casi di test eseguiti. Essa è comunque una misurazione qualitativa e non quantitativa. Infatti, il numero di errori rilevati tiene conto solo degli errori effettivamente riconosciuti come tali e di cui si fornisce una correzione del codice. Sono perciò esclusi dal conteggio gli errori duplicati di altri errori, quelli dovuti ad una operazione sbagliata del testatore, quelli dovuti ad una errata impostazione del sistema (ambiente, base dati, ecc.), ecc. Per riferirsi a questa misurazione si usa il termine di "errori validi". Si In ambienti di test più evoluti si calcola anche il rapporto tra il numero di errori "validi" ed il numero totale di errori rilevati. Un'altra misurazione dell'efficacia del test è rappresentata dal rapporto tra il numero di errori validi rilevati in fase di test e le dimensioni del prodotto.

Efficacia del test (1) = Numero di errori validi / Numero di casi di test eseguiti
Efficacia del test (2) = Numero di errori validi / Numero totale di errori rilevati
Efficacia del test (3) = Numero di errori validi / Dimensioni del prodotto

Efficacia delle revisioni

Rappresenta la capacità di rilevare errori nell'analisi e nel disegno del prodotto tramite le attività di revisione ed ispezioni tecniche. E' misurata come rapporto tra il numero di errori rilevati durante la revisione e le dimensioni del prodotto. Ovviamente le dimensioni del prodotto in fase di analisi e disegno rappresenta la stima iniziale, oppure di pagine revisionate.

Efficacia delle revisioni (1) = Numero di errori / Dimensioni del prodotto
Efficacia delle revisioni (2) = Numero di errori / Numero di pagine ispezionate

Metriche relative alla qualità del software

Complessità ciclomatica

Rappresenta il livello di complessità di un modulo (programma) calcolato rispetto ad un modello strutturato, cioè secondo le regole della programmazione strutturata che tutti i programmatori conoscono. E' calcolata sul grafo che rappresenta la logica interna del modulo dove si prendono in considerazione i cammini logici percorsi dal software all'interno del modulo ed i nodi presenti, cioè i punti decisionali del programma. Il numero ciclomatico è stato introdotto da Tom McCabe e perfezionato da altri. Detto) presenti del modulo ed ia capacità di rilevare errori nell'analisi e nel disegno del prodotto tramite le attività di revisione ed ispezioni tecniche. Dal numero ciclomatico puro è stata derivata una seconda misurazione pratica che è quella effettivamente usata: la complessità ciclomatica "essenziale" di un modulo che è la complessità ciclomatica base calcolata sul grafo dopo aver tutti i cammini strutturati. Mentre per la complessità ciclomatica base Tom McCabe raccomandava valori inferiori a 10, per la complessità ciclomatica essenziale oggi si raccomandano mediamente valori inferiori a 4.

Complessità ciclomatica di un modulo v(G) = e – n + 2 (dove e = cammino, n = nodo)

Livello di accoppiamento

Rappresenta il grado di conoscenza che un modulo/programma ha su di un altro modulo/programma chiamante o chiamato. Esso definisce, cioè, se la logica di un programma dipende dalla logica di un altro programma. L'accoppiamento è bene che assuma, quindi, un valore basso. Esiste una scala di valori di accoppiamento a 6 livelli così definiti:

1. *Per dato*
2. *Per strutture dati*
3. *Per controllo*
4. *Per elementi esterni*
5. *Per aree dati comuni*
6. *Per contenuto*

Livello di coesione

Rappresenta il grado di dipendenza funzionale tra le parti che compongono un modulo/programma, cioè il grado di attinenza delle sue varie parti. Occorre, quindi, che i programmi abbiano un alto valore di coesione. Esistono 7 livelli di coesione, in ordine crescente:

1. *Casuale*
2. *Associazione logica*
3. *Temporale*
4. *Procedurale*
5. *Comunicazione*
6. *Sequenziale*
7. *Funzionale*

Difettosità residua

Rappresenta il numero di errori residui che si suppone abbia il software una volta rilasciato in esercizio. Il tasso di difettosità residuo è ovviamente una stima e potrà essere realmente misurata solo a consuntivo contando il numero di anomalie rilevate dagli utenti in un periodo di tempo stabilito (per esempio nei primi sei mesi di utilizzo oppure nel primo anno di esercizio). Al momento del rilascio del prodotto la stima degli errori residui può essere fatta con tecniche diverse più o meno sofisticate che qui non descriviamo. Il numero a consuntivo, invece, è normalizzato con le dimensioni del prodotto (numero di KLOCs o di FPs).

Difettosità residua = Numero di anomalie rilevate dagli utenti nel periodo considerato / Dimensioni del prodotto

Usabilità

Rappresenta la facilità di utilizzo del prodotto misurata in termini di:

- Facilità di comprensione,
- Facilità di apprendimento,
- Facilità d'uso.

E' un giudizio soggettivo espresso da uno o più utenti sulla propria percezione della facilità d'uso delle funzionalità sperimentate. Per ridurre la soggettività della misura si calcola la media del giudizio espresso da almeno cinque utenti diversi. Il giudizio è espresso utilizzando una scala di valori predefinita (per esempio, su una scala a tre valori: 1 = poco usabile, 2 = mediamente usabile, 3 = molto usabile, oppure 1= bassa usabilità, 2 = media usabilità, 3= alta usabilità, ecc.). Il giudizio espresso dagli utenti coinvolti dipende da fattori personali (es.: cultura, coinvolgimento, conoscenza, ecc.) e dalle attività svolte per esprimere il giudizio (es.: test di usabilità, oppure revisione di un prototipo). Si raccomanda, quindi di scegliere gli utenti da

coinvolgere in numero e rappresentatività adeguata e di eseguire i test simulando quanto più possibile una situazione reale di utilizzo del prodotto (es.: caso d'uso reale e concreto).

Usabilità = Valore medio della valutazione espressa dagli utenti coinvolti su una scala predefinita di valori di usabilità (per esempio su una scala a tre valori di usabilità: bassa, media, alta)

Efficienza

Rappresenta le prestazioni del prodotto, cioè la sua capacità di reazione alle richieste dell'utente. Sono conosciute più propriamente come "performance" del prodotto. E' una misura oggettiva e si riferisce, generalmente ai tempi di risposta di una transazione oppure all'utilizzo di risorse di sistema da parte dell'applicazione per eseguire le funzionalità richieste.

Performance (1) = Tempo medio di risposta di una transazione (in secondi)
Performance (2) = Numero di spazio richiesto su disco o su memoria centrale
Performance (3) = % di utilizzo delle linee di trasmissione impiegate

Robustezza (o Affidabilità)

Rappresenta la capacità del prodotto di continuare a funzionare senza degrado nelle sue prestazioni anche in condizioni particolari (esempio: uso continuativo 24 ore su 24, 7 giorni su 7), oppure in condizioni limite (esempio: 500 utenti collegati contemporaneamente). Si misurano le prestazioni del prodotto (tempi di risposta ed utilizzo delle risorse) e si verifica che non ci sia degrado nelle prestazione. Valgono le stesse metriche definite per calcolare le prestazioni.

Performance (reattività) = Tempo medio di risposta di una transazione (in secondi)
Performance (utilizzo memoria) = Numero di spazio richiesto su disco o su memoria centrale
Performance (utilizzo linee) = % di utilizzo delle linee di trasmissione impiegate

Recuperabilità

Rappresenta la capacità del prodotto di recuperare una condizione anomala quando questa si presenti (esempio: ripristino dei dati di partenza nel caso di caduta durante l'esecuzione di una transazione). Le capacità di recupero possono essere automatiche, semi-automatiche, manuali, assenti. Ovviamente un prodotto senza capacità di recupero delle situazioni anomale è di bassa qualità e potrebbe non essere accettato. Le metriche sono quindi:

Recuperabilità = Automaticamente, Semiautomatica, Manuale, Assente

Manutenibilità

Rappresenta la facilità di eseguire la manutenzione (correttiva o evolutiva) da parte di personale qualificato ma non necessariamente esperto del prodotto specifico. Nel caso di manutenzione correttiva (quella che interessa questo processo specifico) la manutebilità può essere misurata tramite i seguenti parametri:

- Tempo medio di risoluzione delle anomalie in base alla loro gravità;
- Numero di correzioni di anomalie non andate a buon fine (che richiedono, cioè, un secondo intervento correttivo) rispetto al numero totale di anomalie risolte

Manutenibilità (1) = Tempo risoluzione medio/ Totale anomalie risolte
Manutenibilità (2) = Anomalie risolte con riciclo / Totale anomalie risolte)

Metriche object-oriented:

Rappresentano i diversi aspetti che caratterizzano il software sviluppato con la tecnologia ad oggetti. Gli aspetti rilevanti del software object-oriented sono:

- Dimensione del modulo,
- Produttività nello sviluppare una classe,
- Rapporto tra classi principali e classi di supporto,
- Percentuale di dati pubblici (e protetti) di una classe,
- Numero di accessi ai dati pubblici (protetti),
- Metodi "pesati" per classe,
- Profondità dell'albero dell'ereditarietà,
- Numero di figli di una classe,
- Accoppiamento tra le classi/oggetti,
- Risposta di una classe.

Segue una loro descrizione sintetica.

Dimensione di un metodo = Numero di linee di codice di un metodo (in media 1-3 linee di codice, al massimo 10)
Produttività per sviluppare una classe = Numero di giorni-uomo necessari (in media da 10 a 30 giorni)
Rapporto tra classi principali e classi di supporto = in media 1 a 3 (per applicazioni con interfacce utente importanti il rapporto può essere molto più alto)
Percentuale di dati pubblici (e protetti) di una classe = la percentuale dipende dal tipo di applicazione dove la classe è utilizzata
Numero di accessi ai dati pubblici protetti = Numero totale degli accessi
Metodi "pesati" per classe = Complessità e Numero di metodi per classe

Profondità dell'albero dell'ereditarietà = Numero di strati di ereditarietà che costituiscono una gerarchia di classi
Numero di "figli" di una classe = Numero di figli di una classe specifica; oppure media aritmetica calcolata su tutte le classi
Accoppiamento tra le classi/oggetti = Numero totale delle classi con cui una specifica classe interagisce; oppure Media aritmetica calcolata su tutte le classi
Risposta di una classe = Numero di metodi presenti in una classe specifica; oppure Numero di risposte possibili ad un messaggio relativamente ad una classe specifica