Metriche del Software

- Il controllo di qualità del software deve classificare i programmi a riguardo di caratteristiche quali leggibilità, modificabilità, affidabilità etc.

- Obiettivi:

  - Confrontare vari programmi che realizzano le medesime specifiche, in modo tale da scegliere il migliore

  - Confrontare programmi che si riferiscono a specifiche diverse (***valutazione in termini assoluti*** dei programmi)

  - Modelli concettuali, metodi realizzativi dei modelli e tecniche risolutive dei metodi con l'intento di definire e valutare un insieme di ***indicatori delle qualità*** di ogni programma

Cosa misurare?

- Le entita` considerate nella misura del software sono solitamente:

  1. *processi*, attivita` correlate allo sviluppo del software

  2. *prodotto*, qualunque semilavorato o sistema prodotto durante lo sviluppo del software

  3. *risorse*, personale, risorse hardware o software necessarie ai processi

- Gli attributi delle entita` sono classificati in:

  interni, possono essere misurati solo direttamente (ad esempio, la dimensione del software)

  esterni, misurati indirettamente (ad esempio, produttivita` del personale)

Esempio

Entity	Internal Attributes	External Attributes
Product		
Requirements	Size, Reuse, Modularity, Redundancy, Functionality	Understandability, Stability
Specification	Size, Reuse, Modularity, Redundancy, Functionality	Understandability, Maintainability
Design	Size, Reuse, Modularity, Coupling, Cohesion	Comprehensibility, Maintainability, Quality
Code	Size, Reuse, Modularity, Coupling, Cohesion, Control Flow Complexity	Reliability, Usability, Reusability, Maintainability
Test set	Size, Coverage level	Quality
Process		
Requirements analysis	Time, Effort	Cost effectiveness
Specification	Time, Effort, Number of requirements changes	Cost effectiveness
Design	Time, Effort, Number of specification changes	Cost effectiveness
Coding	Time, Effort, Number of design changes	Cost effectiveness
Testing	Time, Effort, Number of code changes	Cost effectiveness
Resource		
Personnel	Age, Cost	Productivity, Experience
Team	Size, Communication Level, Structure	Productivity
Software	Size, Price	Usability, Reliability
Hardware	Price, Speed, Memory size	Usability, Reliability

- La misura e` importante in tutte le fasi di sviluppo del software

- Valutazione della bontà intrinseca di un programma (***metodi statici***, che derivano indicatori nell'àmbito delle ***metriche del software***)

- Valutazione del comportamento durante l'esecuzione (***metodi dinamici***, tramite l'esecuzione derivano indicatori sul comportamento; riguardano essenzialmente l'***affidabilità del software***)

- Per la gestione di un progetto software è di fondamentale importanza poter avere una valutazione delle caratteristiche di un programma *il più presto possibile*

- Valutazioni quantitative ed oggettive nelle prime fasi di codifica, o in fase di specifica

- Le misure effettuate dovrebbero permettere durante l'evoluzione del progetto di tenere sotto controllo la produzione del software e la sua qualità

- Valutazione del processo produttivo stesso (*performance*)

- Metodi finora sviluppati presentano un notevole grado di immaturità

Linee di codice

- L'indicatore che viene utilizzato per misurare le dimensioni di un programma è dato dal numero delle sue ***linee di codice*** (Lines of Code, LOC)

- Utilizzato anche per derivare indicazioni su numero di errori presenti e il tempo necessario per il suo sviluppo (correlazione)

- Indicatore abbastanza criticabile (differenze tra linguaggio, tra realizzazioni delle stesse fuinzionalita`, etc.)

- Non misura la dimensione dei dati (i commenti?)

- Il numero di linee di codice deve essere integrato da altre informazioni per giungere ad una buona valutazione delle qualità del software

Numero di istruzioni

- Indicatore molto simile al LOC
- Sorgono difficoltà e ambiguità di interpretazione, specie nei linguaggi dotati di una struttura a blocchi

if Condizione **then** S1 **else** S2

- Non è sempre chiaro se si debbano considerare le sole istruzioni eseguibili di un programma, oppure anche le dichiarazioni dei dati e i commenti

Software Science

- Metodo introdotto da M. Halstead nel 1977
- Teoria completa che a partire da pochi elementi e semplici fosse in grado di rendere ragione delle caratteristiche misurabili dei programmi
- La Software Science parte dall'assunto che ogni programma sia composto unicamente di ***operatori*** ed ***operandi***
- Indicatori atti a misurare le varie qualità del software determinati dal numero e del numero di volte in cui operatori e operandi si presentano in un programma
- Le ***equazioni*** che definiscono gli indicatori non fanno uso di tecniche statistiche
- Indicatori definiti con lo scopo di misurare le caratteristiche di programmi *a posteriori*
- Indicatori con intento *previsionale*

Software Science: definizioni

- ***Operando***: variabile o costante presente nel programma
- ***Operatore***: simbolo o combinazione di simboli che influenza il valore o l'ordinamento di un operando
- Per ogni linguaggio di programmazione va effettuata una separazione tra operatori e operandi di tutte le entità che possono presentarsi

Software Science: grandezze

- Grandezze fondamentali costituiscono gli ingressi che vengono forniti al modello
- Gli ndicatori delle altre quantità vengono derivati tramite calcoli effettuati su tali grandezze
- In particolare sono considerati fondamentali
  - η_1 numero di ***operatori distinti***
  - η_2 numero di ***operandi distinti***
  - N_1 ***numero*** totale di ***volte*** in cui gli ***operatori*** appaiono nel programma
  - N_2 ***numero*** totale di ***volte*** in cui gli ***operandi*** appaiono nel programma
  - η_2^* numero di ***operandi concettuali*** di ingresso/uscita distinti
- Se il programma considerato consta di più moduli, i valori di η_1, η_2, N_1 ed N_2 si ottengono come somma di quelli che si riferiscono ad ogni modulo

Esempio

```
begin
      max := 0;
      read(x);
      while x<>0 do
           begin
                if x > max
                then max := x;
                read(x)
           end;
      write(max)
end;
```

Operatori		Operandi	
begin … end	2	max	4
:=	2	0	2
;	5	x	5
read	2		
(…)	3		
while…do	1		
<>	1		
if…then	1		
>	1		
write	1		

- $\eta_1 = 10$
- $\eta_2 = 3$
- $N_1 = 19$
- $N_2 = 11$
- $\eta_2^* = 3$ (2 ingressi, 1 uscita)

Le equazioni

- Indicatore *lunghezza* N di un programma, dato da

$$N = N_1 + N_2$$

- Ottenibile solo a posteriori, dopo che il programma è stato ultimato

- A scopi previsionali viene introdotto un ulteriore indicatore, detto lo *stimatore della lunghezza*, calcolato in funzione solo di η_1 ed η_2, e dato da

$$\hat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$$

(Per l'esempio il valore di tale stimatore è circa pari a 38)

- Utile se si hanno buone approssimazioni per η_1 ed η_2 fin dalle prime fasi di codifica, sulla base della tipologia del problema che si sta risolvendo

- Indicatore *volume* V di un programma, inteso come il numero di bit necessario per rappresentare un programma

- Vocabolario η di un programma come

$$\eta = \eta_1 + \eta_2$$

- Numero di bit necessario per codificare ogni elemento del programma: $\log_2 \eta$

$$V = N \log_2 \eta$$

- *Volume potenziale* V^*, volume di un programma in un linguaggio di programmazione in cui la funzionalità che il programma realizza si trova già tra le funzionalità predefinite (numero di operandi uguale a η_2^*, due operatori dati dalla procedura e un operatore di raggruppamento dei suoi argomenti):

$$V^* = \left(2 + \eta_2^*\right) \log_2 \left(2 + \eta_2^*\right)$$

- Indicatore L del *livello del programma* come rapporto tra il volume della forma potenziale ed il volume della realizzazione effettiva:

$$L = \frac{V^*}{V}$$

(Per l'esempio in esame risulta circa valere 0,1045)

- Valutazione attendibile ed univoca di η_2^* difficile, stimatore per L definito da:

$$\hat{L} = \frac{2\,\eta_2}{\eta_1 N_2}$$

(Per l'esempio fornisce il valore 0,05454)

- Indicatore della *difficoltà del programma* come inverso di tale stimatore

- La difficoltà viene utilizzata nella definizione dell'indicatore dello *sforzo* E richiesto per scrivere un programma, inteso come il numero di discriminazioni mentali richieste per scriverlo

- Si suppone che ciascuna delle N operazioni di scelta tra operandi ed operatori nel vocabolario η avvenga con un algoritmo di selezione la cui efficienza è pari a quella della ricerca binaria: il numero totale delle operazioni di scelta è $N \log_2 \eta$, ossia il volume V

- Si suppone poi che il numero di discriminazioni mentali da effettuare per ognuna di tali operazioni sia proporzionale alla difficoltà del programma

- Si ricava che:

$$E = \frac{V}{L} = \frac{V^2}{V^*}$$

- Indicatore del *tempo* occorrente per scrivere un programma (dato un numero S di discriminazioni mentali, compreso per le attività umane tra 5 e 20, per la programmazione circa pari a 18):

$$\hat{T} = \frac{E}{S}$$

nella quale si utilizza per il calcolo di E il valore dello stimatore di L al posto del valore vero

Esempio

N	30
$\widehat{N}$	38
V	112
V^*	11,61
L	0.1037
$\widehat{L}$	0.0545
E	1568
$\hat{T}$	87.1

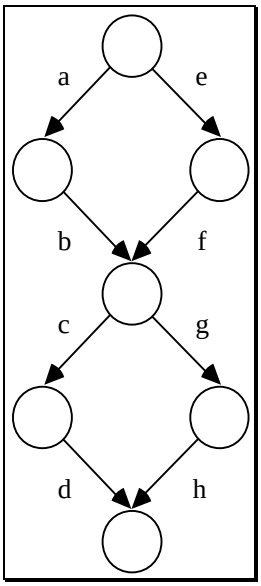
Discussione

- Il modello di Halstead si basa sull'assunzione che tutte le entità che possono comparire in un programma si possano suddividere tra operatori ed operandi
- In molti casi le misure degli indicatori derivati secondo la Software Science risultano piuttosto inattendibili
- La validità degli indicatori si ha non nei singoli casi, ma piuttosto ***in media***, ossia nei riguardi di un campione di programmi reali
- Purtroppo, gli indicatori non vengono costruiti con i metodi impiegati dalla statistica, ma su uno specifico programma
- Gli indicatori derivati risultano più efficaci se applicati a programmi di dimensioni medie
- Alcune ipotesi senza motivazione né teorica né sperimentale

Il numero ciclomatico

- Modello di metrica del software proposto da McCabe nel 1976
- Misura della sola ***complessità*** del software intesa in riferimento alla sua produzione, comprensione e modifica
- Viene preso in considerazione il solo ***flusso di controllo***, senza alcun riferimento alla complessità dei dati (grafo del flusso di controllo)
- Metrica svincolata dalle particolarità di un linguaggio
- Il ***numero ciclomatico*** e` una definizione operativa di complessità del flusso di controllo di un programma
- Identificazione di tutti i cammini che permottano di raggiungere una copertura accettabile del programma

- ***Cammini di base***: cammini completi (a partire dal nodo iniziale raggiungono il nodo terminale del grafo del flusso di controllo), le cui combinazioni lineari esauriscono lo spazio di tutti i cammini completi di un programma



- Per l'esempio, il numero dei cammini di base è 3

- In un grafo fortemente connesso il numero dei cammini linearmente indipendenti si calcola come:

$$e - n + 1$$

dove *e* è il numero degli archi ed *n* è il numero dei nodi

- Il grafo del flusso di controllo di programma non risulta fortemente connesso

- Se il programma è ben formato esiste sempre almeno un cammino che permette di collegare il nodo iniziale con uno qualunque degli altri nodi ed esiste sempre almeno un cammino che permette di collegare uno qualunque dei nodi con il nodo terminale

- Si rende fortemente connesso il grafo del flusso di controllo di un programma aggiungendo un arco orientato che va dal nodo terminale al nodo iniziale (+ 1 arco)

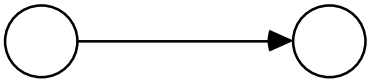
- Il ***numero ciclomatico*** e` il numero dei cammini linearmente indipendenti del grafo G modificato, v(G):

$$v(G) = e - n + 2$$

- Assunto come misura della complessità del flusso di controllo del programma

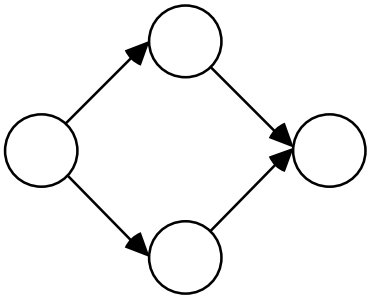
Esempi

Sequenza



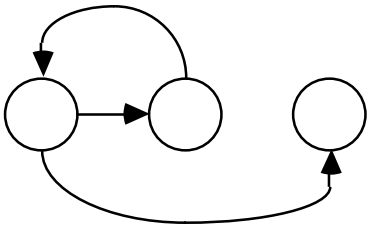
v(G) = 1

Test a due uscite
(if … then … else)



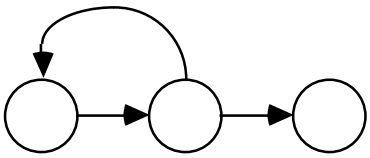
v(G) = 2

Ciclo a condizione iniziale
(while)



v(G) = 2

Ciclo a condizione finale
(repeat)



v(G) = 2

- Il numero ciclomatico viene calcolato anche a partire dal numero dei punti di decisione del programma (teorema di Mills):

$$v(G) = d + 1$$

Si assume però che un punto di decisione a k uscite contribuisca come k-1 punti di decisione a due uscite

- Se il programma ha ***procedure*** al suo interno, il numero ciclomatico dell'intero grafo e` allora dato dalla somma dei numeri ciclomatici dei singoli grafi indipendenti:

$$v(G) = e - n + 2p$$

e ed n sono rispettivamente archi e nodi del grafo nel suo insieme, p il numero di grafi (procedure) indipendenti

Discussione

- Teoria limitata all'analisi della complessità del flusso di controllo
- Meglio fondata matematicamente rispetto a quella di Halstead
- Il numero ciclomatico cattura almeno in parte ciò che intuitivamente è la complessità del flusso di controllo
- Esistono conferme sperimentali che indicano che si ha un buon grado di correlazione tra il numero ciclomatico e grandezze sicuramente influenzate notevolmente dalla complessità del flusso di controllo, come ad esempio il numero degli errori riscontrati
- La complessità ciclomatica di un modulo non deve superare il valore 10 (raccomandazione)

COnstructive COst MOdel

- Metrica definita da B. Boehm con l'obiettivo di descrivere il ***costo di produzione***

- La teoria fornisce una serie di tre modelli:

  Basic COCOMO, questo modello calcola lo sforzo e il costo per lo sviluppo di un software come funzione della dimensione del programma espressa in linee di codice stimate

  Intermediate COCOMO, oltre alla dimensione del programma si introducono nella funzione degli indicatori che includono valutazioni del prodotto, dell'hardware disponibile, del personale e degli attributi del progetto

  Advanced COCOMO, incorpora le caratteristiche della precedente descrizione separando gli indicatori per ogni fase del processo di sviuppo (analisi, schematizzazione, ...)

Basic COCOMO

- Boehm distingue i *progetti software* in tre classi :

  organici, progetti semplici e relativamente piccoli nei quali un gruppo ristretto di programmatori con buona esperienza applicativa lavora con un insieme di specifiche non rigide

  semi isolati**,** progetti di dimensione e complessita` media nel quale un gruppo di programmatori con diverso livello di esperienza deve sddisfare un insieme di specifiche piu` o meno rigide

  comprensivi**,** progetti che devono essere sviluppati rispettando precisi vincoli software, hardware ed operazionali

- La classe del progetto ci dice quali coefficienti ($\underline{a}_b$, $\underline{b}_b$, $\underline{c}_b$, $\underline{d}_b$) introdurre nelle equazioni del modello

Equazioni

- Indicatore E indica lo sforzo (*effort*) applicato in mesi-uomo:

$$E = a_b * KLOC^{\,b_b}$$

- Indicatore D (*development*)per il tempo di sviluppo in mesi

$$D = c_b * E^{\,d_b}$$

(KLOC sono le migliaia di linee di codice stimate)

- Il rapporto E / D fornisce il numero di persone raccomandate per il progetto

- Il risultato delle equazioni e` tanto piu` preciso tanto piu` precisi sono i dati in input ed in particolare, oltre alla classe del progetto, la stima delle linee di codice

- Verifica su piu` di sessanta progetti, precisione dei risultati del ± 20 % nel 70 % dei casi

- Il numero delle linee di codice e` spesso ricavato da una analisi con il metodo dei Function Point

Coefficienti

Progetto Software	a_b	b_b	c_b	d_b
Organico	2.4	1.05	2.5	0.38
Semi isolato	3	1.12	2.5	0.35
Comprensivo	3.6	1.2	2.5	0.32

Intermediate COCOMO

- Il modello di base viene esteso considerando degli indicatori di costo che possono essere raggruppati in quattro categorie:

1. ***Caratteristiche dell prodotto***
   a) Affidabilita` richiesta
   b) Dimensione del database
   c) Complessita` del software
2. ***Caratteristiche hardware***
   a) Vincoli sulla performance durante l'esecuzione
   b) Vincoli di memoria
   c) Volatilita` della macchina virtuale
   d) Tempo di ciclo specificato
3. ***Caratteristiche del personale***
   a) Capacita` degli analisti
   b) Capacita` degli ingegneri del software
   c) Esperienza applicativa
   d) Conscenza della macchina virtuale
   e) Esperienza nel linguaggio di programmazione
4. ***Caratteristiche del progetto***
   a) Uso di strumenti software
   b) Applicazione dei metodi di ingegneria del software
   c) Uso di strumenti per la pianificazione dello sviluppo

- Ognuno attributo e` valutato su di una scala di importanza che va da 0 (molto basso) a 5 (molto alto)

- Sulla base del punteggio di ogni caratteristica una tabella restituisce un moltiplicatore (con valore da 0.7 a 1.66)

- Un moltiplicatore minore di 1 riduce lo sforzo richiesto, maggiore di 1 lo aumenta

- Il fattore di aggiustamento EAF (*Effort Adjustment Factor*) viene ottenuto facendo il prodotto di tutti i moltiplicatori

- L'equazione per E diventa, in questo caso:

$$E = a_b * KLOC^{b_b} * EAF$$

mentre quella per il calcolo di D resta invariata

Metodo dei Function Point:

EI, EO, EQ e loro punteggio

Definizione input esterni

Un input esterno elabora dati od informazioni di controllo provenienti dall' esterno del confine dell' applicazione. L' input esterno e` un processo elementare. I dati elaborati mantengono o meno uno o piu` ILF. Le informazioni di controllo elaborate possono o meno mantenere un ILF

- Complessita` determinata in termini di numero di ***tipi di file referenziati*** (FTR) e di ***elementi di tipo dati*** (DET)

Definizione FTR

Un tipo di file referenziato e` :

Un file logico interno letto o mantenuto dalla funzione

Un file esterno di interfaccia letto dalla funzione

- Si conti solamente un FTR per ciascun ILF che e` sia mantenuto che letto dall' input esterno

Definizione DET

Un elemento di tipo dati (DET) e` un campo unico riconoscibille dall' utente, non ricorsivo, mantenuto in un ILF da un input esterno

Complessita` EI

Si usano due tabelle, per calcolare la complessita` qualitativa della transazione, e il numero di f.p. non pesati risultanti:

	Complessita` EI			Function point per EI	
	DET #	1 .. 4	5 .. 15	16 ..	Compl.	F. point
FTR						
0, 1		Low	Low	Average	Low	3
2		Low	Average	high	Average	4
3 ..		Average	High	High	High	6

Definizione output esterni

Un output esterno e` un processo elementare che genera dati o informazioni di controllo che vengono inviati all'esterno del confine dell' applicazione

- Complessita` determinata in termini di numero di ***tipi di file referenziati*** (FTR) e di ***elementi di tipo dati*** (DET)

Definizione FTR

Un tipo di file referenziato e` un file ***letto*** durante l' elaborazione dell' output esterno

- Si conta un FTR per ogni file logico interno od esterno che sia letto

Complessita` degli EO

	Complessita` EO			Function point per EO	
	DET #	1 .. 5	6 .. 19	20 ..	Compl.	F. point
FTR						
0, 1		Low	Low	Average	Low	4
2, 3		Low	Average	high	Average	5
4 ..		Average	High	High	High	7

Definizione interrogazioni esterne

Un'interrogazione esterna (External Query) e` un processo elementare composto da una combinazione di input ed output che si risolve in un ***reperimento di dati***. Il lato di output non contiene dati derivati. Nessun file logico interno e` mantenuto durante l'elaborazione

Definizione dati derivati

I dati derivati sono i dati che richiedono un trattamento diverso dal semplice reperimento e presentazione delle informazioni provenienti dai file logici interni od esterni

Nota Bene: l' elaborazione non aggiorna alcun ILF

Definizione FTR

Un tipo di file referenziato e` un file *letto* durante l' elaborazione dell' interrogazione esterna

- Si considerino separatamente il lato di input ed il lato di output dell' interrogazione esterna: consideriamo un FTR per il lato di input per ogni file logico i cui campi appaiono nel flusso di dati entrante dall' agente esterno, e un FTR per il lato di output per ogni file

logico i cui campi appaiono nel flusso di dati uscente verso l' agente esterno

Complessita` EQ

- Si considerano separatamente le complessita` dei due lati dell' interrogazione :

  lato input

  lato output

- Per ognuno dei due lati si calcola la complessita` basandosi sui file referenziati FTR e sugli elementi di tipo dati DET

- Nella terza tabella che restituisce i f.p. per ogni EQ si entra con il valore piu` alto di complessita` rilevato fra i due lati dell' interrogazione

Complessita` EQ lato input

DET # / FTR	1 .. 4	5 .. 15	16 ..
0, 1	Low	Low	Average
2	Low	Average	high
3 ..	Average	High	High

Complessita` EQ lato output				Function point per EQ	
DET # / FTR	1 .. 5	6 .. 19	20 ..	Compl.	F. point
0, 1	Low	Low	Average	Low	3
2, 3	Low	Average	high	Average	4
4 ..	Average	High	High	High	6

Metodo dei Function Point: determinazione del fattore di aggiustamento

- Il valore del fattore di aggiustamento introduce nel calcolo l' influenza delle caratteristiche generali del sistema in cui sara` eseguita l'applicazione
- Va applicato al numero di FP per ottenere il numero dei FP pesati
- Introdotto per migliorare i risultati che si ottenevano dall'applicazione del metodo ad alcuni casi pratici

Caratteristiche del sistema

- 14 caratteristiche del sistema prese in considerazione :

  Comunicazione dati

  Distribuzione dell' elaborazione

  Prestazioni

  Utilizzo estensivo della configurazione

  Frequenza delle transazioni

  Inserimento dati interattivo

  Efficienza per l' utente finale

  Aggiornamento interattivo

  Complessita` elaborativa

  Riusabilita`

  Facilita` d' installazione

  Facilita` di gestione operativa

  Molteplicita` di siti

  Facilita` di modifica

- Valutazione della loro influenza sull'applicazione su di una scala di valori 1-5:

  0 Non presente, o di nessuna influenza

  1 Influenza secondaria

  2 Influenza moderata

  3 Influenza media

  4 Influenza significativa

  5 Influenza forte generalizzata

- Il grado di influenza (*Total Degree of Influence* o TDI) e` la somma dei punteggi interi da 0 a 5 attribuiti in vario modo alle 14 caratteristiche che si e` scelto di considerare

- Il fattore di aggiustamento e` legato al grado di influenza da una legge lineare:

$$VAF = 0.01 * TDI + 0.65$$

- ***Comunicazione dati***

- Valutare in quale misura l'applicazione riceve e trasmette dati attraverso sistemi di comunicazione, (si consideri trasmissione dati anche verso il terminale locale dell' unita` di controllo) e quanto cio` abbia influenzato lo sviluppo dell' applicazione, facendo in modo che supportasse piu` protocolli di trasmissione

0 L' applicazione e' una semplice elaborazione batch o risiede su di un calcolatore che opera autonomamente

1 L' applicazione e' batch, ma ha inserimento dati o stampa remoti

2 L' applicazione e' batch, ma ha inserimento dati e stampa remoti

3 L' applicazione utilizza una raccolta dati interattiva o un *front-end* TP (TeleProcessing) verso un processo batch o un sistema di interrogazioni

4 L' applicazione e` piu` di un *front-end* ma supporta solo un tipo di protocollo di comunicazione TP

5 L' applicazione e` piu` di un *front-end*e supporta piu` di un tipo di protocollo di comunicazione

- ***Distribuzione dell' elaborazione***

- Si cerca di indicare come sono distribuiti i dati e le funzioni di elaborazione all' interno dei confini dell'applicazione e se l'applicazione e` interessata dal loro movimento

0 L' applicazione non fornisce ausili al trasferimento dati o funzioni di elaborazione

1 L' applicazione prepara i dati per l' elaborazione da parte dell' utente finale su di un altro componente del sistema come un foglio elettronico o un DBMS

2 L' applicazione prepara, trasferisce ed elabora i dati su di un altro componente del sistema

3 La distribuzione di dati e funzioni e` interattiva ed unidirezionale

4 La distribuzione di dati e funzioni e` interattiva e bidirezionale

5 Le funzioni di elaborazione sono eseguite dinamicamente sul componente piu` appropriato del sistema

- ***Prestazioni***

- Indica se all' applicazione sono posti vincoli stringenti su tempi di risposta o *troughput* (quantita` di lavoro per unita` di tempo) tali da influenzarne lo sviluppo

0 Nessun particolare requisito prestazionale e` stato espresso dall' utente

1 I requisiti prestazionali espressi non comportano azioni particolari

2 Il tempo di risposta o lo *troughput* sono critici durante le ore di picco. Non e` richiesta alcuna progettazione per l' utilizzo della CPU. L' elaborazione puo` essere completata il successivo giorno lavorativo

3 Il tempo di risposta od il *troughput* sono critici durante tutte le ore lavorative. Non e` richiesta alcuna progettazione per l' utilizzo della CPU. I sistemi interfacciati all' applicazione pongono dei vincoli sul completamento dell' elaborazione

4 In aggiunta i requisiti prestazionali richiedono un passo di analisi delle prestazioni durante la progettazione

5 In aggiunta i requisiti prestazionali richiedono analisi delle prestazioni durante le fasi di progettazione, sviluppo, realizzazione

- ***Utilizzo estensivo della configurazione***

- Notiamo se l' applicazione e` stata progettata in funzione di una particolare configurazione di hardware che si intende usare per farla girare

0 L' applicazione non e` vincolata da particolari configurazioni hardware

1 Le configurazioni hardware particolari pongono dei vincoli gia` naturalmente soddisfatti dall' applicazione

2 Occorre fare considerazioni su sicurezza e tempi

3 Una parte specifica dell' applicazione richiede un processore con particolari requisiti

4 Limitazioni operative esplicite richiedono un elaboratore dedicato per l' applicazione

5 L' applicazione pone dei vincoli sui componenti distribuiti del sistema

- ***Frequenza delle transazioni***

- Osserviamo se una eventuale alta frequenza di transazioni ha influenzato le fasi di progettazione, sviluppo, installazione o gestione dell' applicazione

0 Non e` previsto un periodo di picco delle transazioni

1 E` previsto un periodo di picco delle transazioni

2 E` previsto un picco settimanale di transazioni

3 E` previsto un picco giornaliero di transazioni

4 L' alta frequenza di transazioni dichiarata o gli accordi sui livelli di servizio sono tali da richiedere un passo di analisi delle prestazioni durante la progettazione

5 In aggiunta e` richiesto l' uso di strumenti per l' analisi delle prestazioni durante le fasi di progettazione, sviluppo e/o installazione

- ***Inserimento dati interattivo***

- Esaminiamo se l'applicazione fornisce funzioni per l' inserimento e controllo interattivo di dati

0 Tutte le transazioni sono elaborate in modalita` batch

Le transazioni per l' inserimento interattivo di dati sono comprese :

1 fra 1% e 7%

2 fra 8% e 15%

3 fra 16% e 23%

4 fra 24% e 30%

5 fra 30% e 100%

- ***Efficienza per l' utente finale***

- Punteggio assegnato sulla base della presenza di caratteristiche mirate alla facilita`d'uso per l' utente:

  Aiuti di navigazione
  Menu
  Help e dcumentazione in linea
  Spostamento automatico cursore
  Scorrimento
  Stampe remote per mezzo di transazioni interattive
  Tasti funzionali predefiniti
  Richiesta di attivazione di job batch attraverso transazioni interattive
  Selezione con cursore dei dati a video
  Uso di strumenti grafici per arricchire il contenuto informativo (colori, reverse video, ...)
  Documentazione mediante hard copy delle transazioni interattive
  Supporto mouse
  Pop up
  Minimizzazione schermate per completare una sessione interattiva
  Supporto per due lingue (valore 4)
  Supporto multilingue (valore 6)

- Punteggio assegnato:

  0 Nessuna caratteristica

  1 Da 1 a 3

  2 4 o 5

  3 Oltre 6

  4 In aggiunta ci sono delle specifiche che impongono una progettazione orientata ai fattori umani, ad esempio uso di valori predefiniti, di modelli, etc

5 In aggiunta occorre l' uso di strumenti e procedure per verificare il raggiungimento dell' efficienza

- ***Aggiornamento interattivo***

- L' applicazione fornisce l' aggiornamento interattivo dei suoi file interni logici

0 Nessuno

1 Si possono aggiornare in modo interattivoda uno a tre file logici, il volume di aggiornamenti e` basso e le operazioni sui file semplici

2 Si possono aggiornare da quattro o piu` il volume di aggiornamenti e` basso e le operazioni sui file semplici

3 E` possibile aggiornare la maggior parte di ILF

4 In aggiunta il sistema e` stato progettato con protezione contro la perdita di dati

5 In aggiunta elevati volumi di aggiornamento portano a dover considerare i costi di ripristino. Sono presenti procedure di ripristinno automatizzate che richiedono il minimo intervento dell' operatore

- ***Complessita` elaborativa***

- Per valutare l' influenza della capacita` elaborativa vediamo quante fra le seguenti voci appaiono nell' applicazione in esame, il loro numero restituisce il grado di influenza. (0 – 5)

  Controlli dedicati e/o particolari elaborazioni di sicurezza (per esempio speciali elaborazioni di verifica)

  Notevole elaborazione logica

  Notevole elaborazione matematica

  Molte eccezioni che impediscono l' andamento a buon fine di transazioni che devono poi essere rifatte od annullate

  Elaborazione complessa che gestisce piu` possibiita` di input output

- ***Riusabilita`***

- Quanta parte di codice e` sfruttabile od e` stata sfruttata da altre applicazioni

0 Nessuna parte di codice riusabile

1 Il codice e` riusabile all' interno dell' applicazione stessa

2 Meno del 10% dell' applicazione e` sfruttabile dall' utente per altre necessita` non previste dal progetto

3 Il 10% o piu` dell' applicazione e` sfruttabile dall' utente per altre necessita` non previste dal progetto

4 L' applicazione e` stata specificatamente progettata e/o documentata per un facile riuso, essa e` personalizzabile a livello di codice

5 Come sopra tranne che si puo` personalizzare l' applicazione mediante la modifica di parametri utente

- ***Facilita` d' installazione***

0 Non ci sono specifiche e l' installazione non richiede particolari inizializzazioni

1 Non ci sono specifiche ma l' installazione richiede particolari inizializzazioni

2 Ci sono specifiche per la conversione e l' installazione, sono state fornite guide, ma l' impatto sul progetto e` trascurabile

3 Ci sono specifiche per la conversione e l' installazione, sono state fornite guide, l' impatto della conversione sul progetto e` rilevante

4 In aggiunta a punto 2 sono stati forniti strumenti automatici per la conversione e l' installazione

5 In aggiunta a punto 3 sono stati forniti strumenti automatici per la conversione e l' installazione

Nota : con conversione si intende la modifica della struttura di dati per poterli passare da una

applicazione, che li richede con un certo formato, ad un' altra, il cui formato proprietario e` diverso

- ***Facilita` di gestione operativa***

- Si intende vedere se l' applicazione minimizza la necessita` di attivita` manuali quando richiesto dall' utente

0 Non ci sono specifiche eccetto le normali procedure di salvataggio

1 .. 4 Vedere quante delle seguenti voci appaiono

Sono fornite efficaci procedure di avviamento salvataggio e ripristino ma e` richiesto l' intervento dell' operatore

Sono fornite efficaci procedure di avviamento salvataggio e ripristino e non e` richiesto l' intervento dell' operatore (vale due voci)

L' applicazione minimizza la necessita` di montaggio di nastri

L' applicazione minimizza la necessita` di gestione della carta

2 L' applicazione deve svolgere operazioni non presidiate (nessun intervento di operatore ad eccezione di avviamento o chiusura), occorre il recupero automatico di eventuali errori

- ***Molteplicita` di siti***

- L' applicazione e` nata considerando una sua eventuale diffusione in varie sedi con esigenze simili, e per sfruttare questa diffusione

0 Non e necessario considerare piu` l'installazione dell' applicazioni in piu` sedi (o per piu` utenti)

1 Si e` considerata l' installazione in piu` siti con hardware e software identici

2 Si e` considerata l' installazione in piu` siti con hardware e/o software simili

3 Si e` considerata l' installazione in piu` siti con hardware e/o software diversi

4 L' applicazione e` descritta dal punto 1 o 2 inoltre e` fornita documentazione e piani di supporto per la gestione in piu` siti

5 L' applicazione e` descritta dal punto 3 inoltre e` fornita documentazione e piani di supporto per la gestione in piu` siti

- ***Facilita` di modifica***

- Per valutare l' influenza della facilita` di modifica vediamo quante fra le seguenti voci appaiono nell' applicazione in esame, il loro numero restituisce il grado di influenza (0 – 5)

  Sono fornite delle interrogazioni flessibili ed ausili per la produzione di prospetti che gestiscono semplici richieste, ad esempio and or logici applicati solamente ad un ILF

  Sono fornite delle interrogazioni flessibili ed ausili per la produzione di prospetti che gestiscono richieste di media complessita`, ad esempio and or logici applicati a piu` di un ILF (vale due voci)

  Sono fornite delle interrogazioni flessibili ed ausili per la produzione di prospetti che gestiscono richieste complesse, ad esempio combinazioni di and or logici applicati a uno o piu` ILF (vale tre voci)

  I dati di controllo per le funzioni sono in tabelle che l' utente puo` mantenere con elborazioni interattive, i cambiamenti diventano effettivi il giorno seguente

  I dati di controllo per le funzioni sono in tabelle che l' utente puo` mantenere con elaborazioni interattive, i cambiamenti diventano immediatamente effettivi il giorno seguente (vale due voci)

Calcolo del valore del fattore di aggiustamento VAF

- Si sommano tutti i gradi di influenza assegnati alle 14 caratteristiche sopraelencate ottenendo il totale grado di influenza TDI

- Il fattore di aggiustamento, VAF, si ottiene quindi :

$$VAF = TDI / 100 + 0.65$$

- Il VAF permette di arrivare al risultato finale che e` il ***numero di function point pesati***

- Il numero dei function point pesati va comunque tenuto in considerazione nel caso si pensi di fare una manutenzione evolutiva dell' applicazione

Calcolo del numero finale di function point pesati

- Il calcolo si differenzia nei tre casi :

  Progetto di sviluppo

  Manutenzione evolutiva

  Applicazione

Progetto di sviluppo

- Il numero di function point pesati per un progetto di sviluppo (Development Function Point o DFP) e` dato dal numero totale dei function point calcolato sulla base delle funzionalita` dell' applicazione incluse fra i requisiti utente, considerando separatamente le funzionalita` di conversione (CFP), moltiplicato per il fattore di aggiustamento

$$DFP = (UFP + CFP) * VAF$$

Progetto di manutenzione evolutiva

- Per il calcolo del numero di function point pesati per un progetto di manutenzione evolutiva (Evolution Function Point) occorre conoscere vari dati

  ADD (ADDed) il numero di function point non pesati dato dalle funzionalita` aggiunte nell' applicazione

  CFP (Conversion Function Point) il numero di function point non pesati dato dalle funzionalita` di conversione aggiunte nell' applicazione

  CHGA (After CHanGe) il numero di function point non pesati delle funzionalita` dell' applicazione dopo i cambiamenti

  VAFA, VAFB i valori dei fattori di aggiustamento dopo e prima la manutenzione evolutiva

  DEL (Deleted) il numero di function point non pesati delle funzionalita` dell' applicazione rimosse

$$EFP = (ADD + CFP + CHGA) * VAFA + (DEL) * VAFB$$

Nota : questo e` il "costo" della manutenzione, non il numero di f.p. dell' applicazione che va invece ricalcolato, in conformita` col parziale calcolo gia` eseguito seguendo lo schema sottostante

Applicazione

- Si considerano separatamente i due casi di un' applicazione nativa (di cui non esistono precedenti versioni) o di un' applicazione evoluta (con aggiornamenti fatti per passare da una versione alla successiva) per arrivare ai function point pesati per un' applicazione (AFP)

Applicazione nativa

$$AFP = ADD * VAF$$

- Non essendoci precedenti versioni non possono apparire in questa formula le funzionalita` di conversione

Applicazione evoluta

$$AFP = (UFPB + ADD + CHGA - CHGB - DEL) * VAFA$$

UFPB numero dei function point non pesati dell' applicazione prima della manutenzione evolutiva

CHGB, CHGA numero dei function point non pesati delle funzioni modificate dalla manutenzione evolutiva prima e dopo i cambiamenti

Tool esistenti

- Ancora in un primo stadio di sviluppo, pertanto hanno funzionalita` abbastanza limitate

Seat 2.42, Software Estimation and Analysis Tool,

- Liberamente disponibile, per chiunque fosse interessato, nel sito etsu.east-tenn-st.edu
- Sistema operativo DOS e l' ambiente grafico Microsoft Windows 3.xx
- Supporta sia il tipo di analisi con i function point sia il tipo di analisi con il modello a costi di costruzione (Constructive Cost Model o COCOMO)
- Calcola, a partire dai numeri relativi alle funzioni di base:

  il numero dei f.p. non pesati

  il valore del fattore di aggiustamento

  il numero dei f.p. pesati

  il numero di istruzioni occorrenti nel linguaggio scelto per la codifica

Function Point Analysis Spreadsheet ver. 1.1

- Basato su Excel 7.0 della Microsoft per il sistema operativo Windows 95
- Di pubblico dominio
- Questo software scende piu` nel dettaglio dell' analisi del precedente, risulta qundi essere piu` completo
- Anche questo tool e` solo di supporto all'analista che rimane il centro del processo riconoscitivo e decisionale (*identificazione*)