Università degli Studi di Udine

Metriche del Software

prof. Maurizio Pighin

Dipartimento di Matematica e Informatica

Metriche

- Metriche: misurazione delle caratteristiche dei prodotti software e dei processi di sviluppo
- Trattazione
  - *Definizione empirica di misura e delle scale di misura*
- La misura degli attributi del software

Cosa è una misura

- Il cuore di molti sistemi che governano la nostra vita
  - *In economia determina il prezzo delle cose e le variazioni di prezzo*
  - *In medicina consente diagnosi accurate*
  - *In fisica consente la creazione di modelli del reale*
  - *In ingegneria è la base per la progettazione*
  - *In meteorologia consente la predizione delle condizioni atmosferiche*
- Misurare per
  - *controllare*
  - *verificare*
  - *consuntivare*
  - *stimare*
  - *supportare decisioni*

Cosa è una misura

- Misurazione: processo attraverso cui , secondo regole ben definite, numeri o simboli sono assegnati a particolari proprietà di un oggetto per poterlo descrivere
- Misurazione = processo, misura = risultato della misurazione
- Metrica: misura quantitativa del grado di possesso di un attributo da parte di un'entità [IEEE]
  - *Es: numero di linee di codice di un modulo, numero medio di errori nei componenti di un' applicazione, ...*
- Comunemente:
  - *metrica sinonimo di misura*
  - *la definizione di metrica include procedure e modalità di misurazione*

Cosa è una misura

- Conseguenze:
  - *La misura non è un numero (simbolo) ma una relazione tra attributi e numeri (simboli)*
- Osservazione:
  - *Per effettuare una misura dobbiamo avere una chiara idea di quali attributi vogliamo misurare e di quali entità possiedano tale attributo*
    - La comprensione empirica precede la misura
    - Es. temperature, altezza, inflazione, …
  - *Sembra ovvio ma la maggior parte delle misure nell'ingegneria del software non chiarisce esattamente la natura degli attributi che vengono misurati*

Cosa è una misura

- Misura
  - *Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules (Fenton).*
- Attributi
  - *caratteristiche o proprietà di un'entità*
- Osservazioni
  - *l'altezza di un uomo è un attributo misurabile; l'intelligenza (IQ)?*
  - *l'accuratezza dipende dallo strumento*
  - *comunque esistono margini di errore*
  - *esistono scale diverse di misura per lo stesso attributo*
  - *che manipolazioni si possono fare sui dati di misura?*

La misura nell'ingegneria

- Le misure rappresentano un aspetto cruciale in tutte le branche dell'ingegneria… (se esiste lo posso misurare) … tranne che nell'ingegneria del software
- Nell'ingegneria del software si parla di proprietà come usabilità, affidabilità e manutenibilità senza spiegare come possano essere misurate
- Le poche misure che vengono fatte lo sono in maniera infrequente, inconsistente e incompleta, spesso senza uno scopo preciso
- Alcuni riferimenti famosi:
  - *You cannot control what you cannot measure (DeMarco)*
  - *Projects without clear goals will not achieve their goals clearly (Gilb'principle of fuzzy targets)*
  - *Se qualcosa non è misurabile, rendila misurabile (Galileo)*

Cosa misurare

- Manager
  - *costo di sviluppo: per determinare il prezzo del prodotto*
  - *produttività: per dimensionare i team*
  - *qualità del prodotto: per confrontare prodotti diversi*
  - *efficacia di metodi e tool: per scegliere i migliori*
- Ingegneri
  - *qualità del prodotto: per valutare a che punto sia nel ciclo di sviluppo*
  - *qualità del processo: per decidere se e cosa variare per ottenere un processo più efficiente, affidabile, prevedibile, ecc.*

Misure dirette o indirette

- Misura diretta di un attributo
  - *La misura diretta di un attributo non dipende da altri attributi*
- Misura indiretta
  - *La misura di un attributo è funzione della misura di altri attributi*
- Esempio:
  - *Altezza di una persona: misura diretta*
  - *Temperatura di un corpo mediante termometro a mercurio: misura indiretta*
    - Misuriamo la lunghezza della barretta di mercurio

Misure dirette o indirette

- Esempi nell'ingegneria del software
  - *Misure dirette*
    - Durata progetto → mesi dall'inizio alla fine
    - Lunghezza programma → numero di linee di codice
    - Affidabilità di un programma → MTBF in ore CPU
  - *Misure indirette*
    - Produttività programmatore → LOC/Mese
    - Stabilità requisiti → Requisiti_iniziali/Requisiti_finali
    - Densità difetti → Numero_difetti/LOC_modulo

Misure e Sistemi Predittivi

- Misure
  - *Un sistema di misurazione è un sistema che permette di valutare alcuni attributi generati da eventi conclusi (controllo)*
  - *Esempio: tempo totale sviluppo progetto, come sommatoria dei tempi di sviluppo progetto per le varie persone impegnate nelle varie fasi*
- Sistemi predittivi
  - *A* Prediction System *consists of a mathematical model together with a set of prediction procedures for determining unknown parametres and interpreting results (Littlewood) (stima)*
  - *Esempio: distribuzione esponenziale per il tempo dell'iesimo errore di un prodotto, essendo N il numero di errori presenti nel prodotto*
    $F(t)=i-e^{-(N-i+1)at}$

Valutazione di Misure e Sistemi Predittivi

- Measures
  - *the process of ensuring that a numerical characterization of the claimed attribute is appropriate*
    - showing that the representation condition is satisfied
- Prediction Systems
  - *the process of establish the accuracy of the prediction system by empirical means*
    - comparing model performance with known data in given environment
  - *System types*
    - Deterministic
    - Stochastic (accuracy)

Valutazioni di Misure e Sistemi Predittivi

- Un sistema Predittivo stima valori non noti di caratteristiche. Va sempre valutata la bontà del modello previsto, attuando delle verifiche “a posteriori” con misure consuntive/analisi statistiche
- Esempio
  - *Stimo la durata di un progetto, con un modello matematico a partire dal numero dei requisiti*
  - *Valuto a posteriori la bontà verificando lo scostamento fra valore previsto e valore misurato*
  - *Eventualmente taro il modello*

Processo definizione misura

- Identificare gli attributi delle entità da misurare
- Identificare le relazioni empiriche fra gli attributi che ci interessa tracciare
- Identificare mapping tra numeri (simboli) e gli attributi di ciascuna entità
- Verificare che il mapping rispetti le relazioni empiriche, quindi che le relazioni fra numeri (simboli) siano analoghe alle relazioni empiriche di partenza

Esempio: criticità dei malfunzionamenti

- Esempio
  - *Si supponga di dover misurare la criticità dei malfunzionamenti*
  - *Iniziamo con il classificare i malfunzionamenti in sintattici, semantici e crash del sistema. Supponiamo che ogni malfunzionamento appartenga ad una ed una sola di queste classi*
  - *Supponiamo che vi sia un ordinamento di criticità tale per cui i malfunzionamenti sintattici sono meno gravi di quelli semantici, che a loro volta sono meno gravi dei crash di sistema*
  - *Assegniamo un numero (simbolo) ad ogni classe*
    - Criticità Sintattico → 1
    - Criticità Semantico → 2
    - Criticità Crash → 3

Esempio: criticità dei malfunzionamenti

- *Verifichiamo il rispetto delle relazioni*
  - Criticità Sintattico < Criticità Semantico < Criticità Crash
  - 1 < 2 < 3
- *Attenzione agli operatori da usare 1+1=2 ?*
- *Vedremo ora che i tipi di operatori che si possono usare nei vari contesti*

Scale di misura

- Si definisce “scala di misura” l’insieme di
  - *un sistema di relazioni empiriche* Σ
  - *un sistema di relazioni numeriche* Γ
  - *un mapping M fra* Σ *e* Γ
- Trasformazioni di scala
  - L’ esempio precedente ammette infiniti sistemi di relazioni numeriche e di conseguenza infinite misure e scale
    - *L’ applicazione di una qualsiasi trasformazione monotona ad una di queste misure ne fornisce un’altra*
    - *Definiamo trasformazione di scala ammissibile una funzione che trasforma una qualsiasi scala in un’altra*

Tipo di scale

- Il “tipo” di una scala è determinato dall’insieme di trasformazioni di scala ammissibili per tale scala
- Più piccolo è tale insieme migliore è la scala
- Nozione empirica
  - *Classificazione di un insieme di possibili valori: scala* ***nominale***
  - *Ordinamento lineare dei valori: scala* ***ordinale***
  - *Esiste anche un concetto di “distanza relativa tra i valori”: scala* ***intervallo***
  - *Esiste anche un elemento “zero”: scala a* ***rapporto***
  - *Le scale* ***assolute*** *derivano da attributi che danno luogo ad un semplice conteggio di entità*

Scale di misurazione

- Nominal Scale
  - *The empirical relation system consists only of different classes There is no notion of ordering*
  - *Any distinct numbering or symbolic representation of the classes is an acceptable measure, but there is no notion of magnitude associated with the numebrs or symbols*
  - *Es. Tipo di errore (specifiche, progetto, codifica)*
- Ordinal Scale
  - *The empirical relation system consists of classes that are ordered with respect to the attribute*
  - *Any mapping that preserves the ordering (that is a monothonic function) is acceptable*
  - *The numbers represents ranking only, so addition, subtraction and other arithmetic operations have no meaning*
  - *Es. 5 livelli di complessità per un algoritmo*

Scale di misurazione

- Interval scale
  - *An interval scale preserves order, as with an ordinal scale*
  - *An interval scale preserves differences, but not ratios. That is, we know the difference between any two of the ordered classes in the range of the mapping, but computing the ratio of two classes in the range does not make sense*
  - *Addition and subtraction are acceptable in the ratio scale, but not multiplication and division*
  - *Es. Cinque livelli di complessità, con step OMOGENEI e DEFINITI (ad esempio legati al tempo di sviluppo)*
  - *Es. Temperatura (Celsius/Fahrenheit)*

Scale di misurazione

- Ratio scale
  - *It is a measurement mapping that preserves ordering, the size of intervals between entities, and the ratio between entities.*
  - *There is a zero element, representing total lack of attribute*
  - *The measurement mapping must start at zero and increase at equal intervals, known as units*
  - *All arithmetic can be meaningfully applied to the classes in the range of the mapping*
  - *Es. Lunghezza del codice (byte, linee eseguibili, ecc.) (si può dire "è lungo il doppio")*

Scale di misurazione

- Absolute scale
  - *The measurement for an absolute scale is made simply by counting the number of elements in the entity set*
  - *The attribute always takes the form "number of occurences of x in the entity"*
  - *There is only one possible measurement mapping, namely the actual count*
  - *All arithmetic analysis of the resulting count is meaningful*
  - *Es. Lines of code (LOC)*

Trasformazioni Possibili

Sistemi di relazioni empiriche più ricchi
= meno trasformazioni possibili
= scale più sofisticate

Trasformazioni possibili	*Tipo scala*	*Esempi*
M'= F(M) F mapping 1-1	Nominale	Label/classificazioni
M≐F(M) F trasf. monotona cresc	Ordinale	Preferenza
M≐aM+b a>0	Intervallo	Temperature °C °F
M≐aM a>0	Rapporto	Intervalli temporali, lunghezze, temperature K
M≐M	Assoluta	Entità numerabili

Esempi di scale

- Esempio relativo alla criticità dei malfunzionamenti senza la relazione d'ordine: scala nominale
- Esempio relativo alla criticità dei malfunzionamenti completo: scala ordinale
- Esempio relativo alla criticità dei malfunzionamenti con l'aggiunta dell'osservazione che la differenza di criticità tra malfunzionamenti sintattici e semantici coincide con la differenza di criticità tra malfunzionamenti semantici e crash di sistema: scala intervallo
- Distanza tra due oggetti: scala a rapporto

Indici di rappresentazione

Tendenza Centrale

We have measured an attribute for 13 entities, and the resulting data points in ranked order are:
2, 2, 4, 5, 5, 8, 8, 10 ,11, 11, 11, 15, 16
The **mean** of this set of data (that is, the sum divided by the number of items) is *9.1*
The **median** (that is, the value of the middle-ranked item) is *8*
The **mode** (that is, the value of the most commonly occurring item) is *11*

ure 2.12: Different ways to compute the central tendency of a set of numbers

Dispersione

-**Varianza, Deviazione Standard**

Tipi di scala ed operazioni ammesse

- Il tipo di scala di una misura determina l'insieme di operazioni ammesse (meaningful)
  - *Scale nominali ammettono:*
    - frequenza
  - *Scale ordinali ammettono:*
    - percentile, mediana
  - *Scale intervallo e a rapporto ammettono:*
    - Media, deviazione standard, media geometrica
  - *Ha senso parlare di:*
    - Valor medio della criticità delle failure? (ordinale)
    - Lunghezza media dei moduli di un programma? (rapporto)

Operazione Ammesse

Table 2.8: Summary of measurement scales and statistics relevant to each (Siegel and Castellan, 1988)

Scale type	Defining relations	Examples of appropriate statistics	Appropriate statistical tests
Nominal	Equivalence	Mode Frequency	Non-parametric
Ordinal	Equivalence Greater than	Median Percentile Spearman r Kendall τ Kendall W	Non-parametric
Interval	Equivalence Greater than Known ratio of any intervals	Mean Standard deviation Pearson product-moment correlation Multiple product-moment correlation	Non-parametric
Ratio	Equivalence Greater than Known ratio of any intervals Known ratio of any two scale values	Geometric mean Coefficient of variation	Non-parametric and parametric

Sensatezza

- Le definizioni date sino ad ora ci permette di definire formalmente la “sensatezza” (meaningfulness) di una affermazione relativa ad una misura
  - *Una affermazione relativa ad una misura è sensata sse la sua verità resta invariata a seguito di una trasformazione ammissibile di scala*
- Esempi:
  - *A è alto il doppio di B, “doppio” richiede scala a rapporto,*
    M = a M’, altezza → scala a rapporto → ammissibile
  - *A è caldo il doppio di B, temperatura °C °F K,*
    40°C = 2 x 20°C ---- 104°F = 2 x 68°F → non ammissibile: scala a intervallo non ammette prodotti

Esempi di sensatezza

- Si considerino le seguenti affermazioni:
  - *Il numero di errori scoperti in un programma è 100*
    - sensata, il numero di errori è una misura assoluta (scala assoluta) che non richiede l'indicazione della scala
  - *Il costo di correzione di ogni errore in un programma x è 100*
    - priva di significato, il costo è una misura avente scala a rapporto. Occorre specificare l'unità della scala adottata per fornire un valore "assoluto"
  - *Un errore semantico richiede, per essere corretto, il doppio del tempo rispetto ad un errore sintattico*
    - sensata, il tempo è una scala a rapporto ed è corretto parlare di "doppio" anche senza specificare l'unità della scala adottata (non avrebbe senso invece dare valori senza definire l'unità)

Ingegneria del Software
Progettazione e Laboratorio
Metriche del Software
Maurizio Pighin

Misure nell'ingegneria del SW

- La principale osservazione sulle misure:
  - prima di misurare occorre decidere quali entità prendere in considerazione e quali attributi di tali entità misurare
- Spesso errori di misura derivano dal non aver individuato esattamente l'oggetto della misura (prodotto, processo, risorsa) e le caratteristiche dell'attributo che si misura (interno o esterno)

Esempio

- Si consideri l'attributo "numero di malfunzionamenti individuati durante la fase di test"
  - *Tale attributo viene talvolta considerato un attributo di processo, tal altra un attributo di prodotto (numero di errori del programma)*
  - *Si osservi che il primo è un attributo esterno (dipende dalla durata della fase di test), il secondo un attributo interno*
- Comprendere che si tratta di un attributo esterno di processo ci aiuta a capire da cosa possa dipendere (durata della fase di test, esperienza dei tester) e a costruire un modello di misura adeguato

Cosa Misurare

- Entità:
  - *Processi*
    - Attività
    - Collection of software related activities (Fenton)
  - *Prodotti*
    - Output dei processi
    - Any artifacts, deliverables or documents that results from a process activity (Fenton)
  - *Risorse*
    - Input dei processi
    - Entities required by a process activity (Fenton)

Attributi Interni/Esterni

- Attributi interni
  - *Sono gli attributi di una entità che possono essere misurati a partire dalla sola entità*
  - *Measure of product, process or resource on its own, separate from its behaviour (Fenton)*
- Attributi esterni:
  - *Sono gli attributi di una entità che devono essere misurati in termini di come l' entità è in relazione con l'ambiente esterno*
  - *Attributs that can be measured only with respect to how the product, process or resource relates to its anvironments (Fenton)*
  - *Sono i più importanti ma anche i più difficili da misurare:*
    - efficienza
    - affidabilità
    - produttività
    - ...

Esempio:misure interne/esterne

Table 3.1: Components of software measurement

ENTITIES	ATTRIBUTES	
Products	*Internal*	*External*
Specifications	size, reuse, modularity, redundancy, functionality, syntactic correctness, ...	comprehensibility, maintainability, ...
Designs	size, reuse, modularity, coupling, cohesiveness, functionality, ...	quality, complexity, maintainability, ...
Code	size, reuse, modularity, coupling, functionality, algorithmic complexity, control-flow structuredness, ...	reliability, usability, maintainability, ...
Test data	size, coverage level, ...	quality, ...
...	...	...
Processes		
Constructing specification	time, effort, number of requirements changes, ...	quality, cost, stability, ...
Detailed design	time, effort, number of specification faults found, ...	cost, cost-effectiveness, ...
Testing	time, effort, number of coding faults found, ...	cost, cost-effectiveness, stability, ...
...	...	...
Resources		
Personnel	age, price, ...	productivity, experience, intelligence, ...
Teams	size, communication level, structuredness, ...	productivity, quality, ...
Software	price, size, ...	usability, reliability, ...
Hardware	price, speed, memory size, ...	reliability, ...
Offices	size, temperature, light, ...	comfort, quality, ...
...	...	...

Schema di lavoro

- Come decidere cosa misurare
  - *Ad es. GQM*
- Che attributi misurare
  - *Interni*
  - *Esterni*
- Che tipo di informazioni raccogliere
  - *Che tipo di dati di misura usare*
  - *Come raccogliere i dati di misura*

The Goal Question Metric paradigm

- GQM
  - *List the major goals (for instance of the development or maintenance project)*
  - *Derive from each goal the question to be answered to determine if goals are being met*
  - *Decide what must be measured in order to able to answer the questions adequately*

The Goal Question Metric

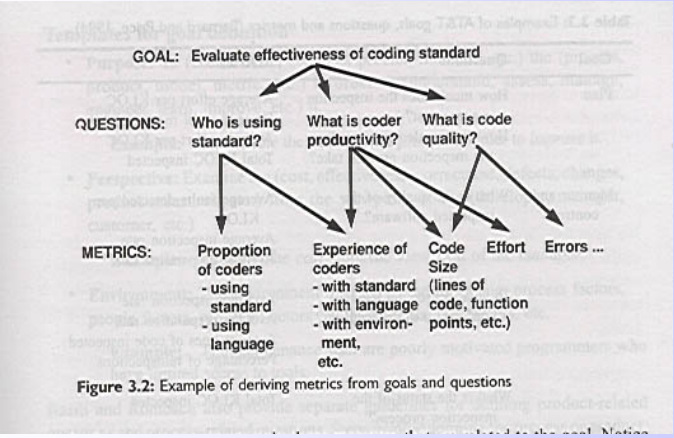


Figure 3.2: Example of deriving metrics from goals and questions

The Goal Question Metric

Table 3.3: Examples of AT&T goals, questions and metrics (Barnard and Price, 1994)

Goal	Questions	Metrics
Plan	How much does the inspection process cost?	Average effort per KLOC Percentage of reinspections
	How much calendar time does the inspection process take?	Average effort per KLOC Total KLOC inspected
Monitor and control	What is the quality of the inspected software?	Average faults detected per KLOC Average inspection rate Average preparation rate
	To what degree did the staff conform to the procedures?	Average inspection rate Average preparation rate Average lines of code inspected Percentage of reinspections
	What is the status of the inspection process?	Total KLOC inspected
Improve	How effective is the inspection process?	Defect removal efficiency Average faults detected per KLOC Average inspection rate Average preparation rate Average lines of code inspected
	What is the productivity of the inspection process?	Average effort per fault detected Average inspection rate Average preparation rate Average lines of code inspected

Internal product attributes

- Size
  - *length*
  - *reuse*
  - *functionality*
  - *complexity*

Internal product attributes:lenght

- Codice
- La dimensione di un programma secondo tale metrica viene solitamente indicata con la sigla SLOC, ossia *Source Line Of Code*, ma è possibile indicarla anche nei seguenti modi:
  - LOC: *Lines Of Code*;
  - SLOC: *Lines of Source Code*
  - DSLOC: *Delivered SLOC*;
  - DSI: *Delivered Source Instruction*;
  - KDSI: *Thousands of DSI*
  - NCLOC: *Non comment lines*
  - CLOC: *Comment lines*

Lunghezza: la metrica di Halstead

- Software Science (1972)
- I parametri di riferimento
  - *n1 numero operatori distinti*
  - *n2 numero di operandi distinti*
  - *N1 numero totale degli operatori utilizzati*
  - *N2 numero totale degli operandi utilizzati*
  - *nio numero di operandi concettuali di I/O*
- Vocabolario n=n1+n2
- Lunghezza N=N1+N2

La metrica di Halstead

- Ns = n1 log n1 + n2 log n2
  (lunghezza stimata)
- Esempi di conteggi

```
void selection_sort (long int A[ ])
  {
     extern long int max ;
     long int i, j , c ;
     for (i = 0 ; i < = max - 1 ; i + +)
        for (j = i ; j < = max ; j + +)
           if (A[ i ] > A[ j ])
              {
                 c = A [ i ] ;
                 A [ i ] = A [ j ] ;
                 A[ j ] = c ;
              }
  }
```


La metrica di Halstead

operatori	frequenza
;	7
for	2
(. . .) grouping	3
=	5
< =	2
-	1
+ +	2
>	1
[ ] ind. vettore	6
if	1
begin_end	2
n1 = 11	N1 = 32

La metrica di Halstead

operandi	frequenza
i	7
0	1
j	6
max	2
c	2
1	1
A	6
n2 = 7	N2 = 25

Ne segue che N = 57 (corrisponde alla lunghezza teorica), mentre la lunghezza stimata $N_S = 11 \times \log 11 + 7 \times \log 7 = 57.71$, fornendo un rapporto $N_S / N = 1.01$

La metrica di Halstead

- Volume V = N log n
  - *Rappresenta il numero di bit necessario per rappresentare un programma*
  - *Log n è il numero di bit per codificare il vocabolario (ogni elemento del programma)*
  - *Essendo N la lunghezza N log n dà il volume*
- Volume potenziale (funzione che implementa la primitiva richiesta)
  Vp= (2+nio) log (nio+2)
  - *Il +2 è rappresentato dal nome della procedura più l'operatore di raggruppamento (usualmente le parentesi) che raggruppano i parametri*

La metrica di Halstead

- Livello programma L=Vp/V
- Ls = (2 n2)/(n1 N2) (liv. stimato)
- 1/L = difficoltà = V/Vp

La metrica di Halstead

- E = V/L=V^2/Vp
  - *Sforzo. Si suppone N log n discriminazioni. Ogni operando o operatore è scelto con la difficoltà di una scelta binaria*
- T=E/S
  - *Tempo, essendo S il numero di discriminazioni al secondo*
- Stroud 5-25 usato 18
- LL=Vp*L
  - *Livello linguaggio*

La metrica di Halstead

- Continuando con l'esempio
  - *V=(N1+N2)log(n1+n2)=(32+25) log (11+7)=271.9 bit*
    - Volume
  - *Vp=(2+nio)log(2+nio)=(2+3) log (2+3)=11.6 bit*
    - Volume potenziale - 3=vettore + pos_ini+pos_fin
  - *L=(Vp/V)=11.6/271.9=0.0427*
    - Livello programma
  - *Ls=2*n2/(n1*N2)=(2*7)/(11*25)=0.0509*
    - Livello stimato

La metrica di Halstead

- *1/L=23.4*
  - difficoltà
- *E=V/L=6374*
  - Sforzo
- *T=E/18=354 sec= 5min 54sec*
  - Tempo previsto
- *LL=Vp*L=11.6*0.0427=0.452*
  Livello Linguaggio

- Critiche
  - *non tutto è operatore e operando*
  - *operatori e operandi danno scarso valore semantico*
  - *validità in media, su un intero campione, non su programmi singoli*

Internal product attributes:lenght

- Specification and design
  - *oggetti atomici DFD (processes, external entities, data stores, data flows)*
- Predicting lenght
  - *Essendo N la lunghezza secondo Halstead*
    - LOC=N/C_k (ad es. FORTRAN C_k=7)

Internal product attributes:reuse

- Software reused
  - *completely reused*
  - *slightly modified (<25%)*
  - *extensively modified (>25%)*
  - *new*
- Reuse ratio

Table 7.5: Reuse of code at Programming Research Ltd (Hatton, 1995)

Product	Reusable lines of code	Total lines of code	Reuse ratio (%)
QAC	40 900	82 300	50
QA FORTRAN	34 000	73 000	47
QA Manager (X)	18 300	50 100	37
QA Manager (Motif)	18 300	52 700	35
QA C++	40 900	82 900	49
QA C Dynamic	11 500	30 400	38

Internal product attributes:functionality

- Intuitive notion of the amount of function contained in a delivered product or in a description the product is supposed to be
- Albrecht metric (FUNCTION POINT)
  - *la misura isola la dimensione intrinseca del sistema dai fattori ambientali, facilitando lo studio dei fattori che influenzano la produttività*
  - *la misura è basata sulla vista esterna dell'utente ed è tecnologicamente indipendente*
  - *la misura può essere determinata in fase iniziale di sviluppo del progetto, permettendo ai Function Points di essere usati nel processo di stima*
  - *i Function Points possono essere capiti e valutati da utenti non tecnici.*

Funzionalità: i Function Points

- Il valore del Function Point (FP) viene calcolato in cinque passi generici:
  - *Identificazione e classificazione di cinque indici;*
  - *Definizione della complessità di ogni indice;*
  - *Determinazione del Function Count (FC);*
  - *Determinazione del Technical Complexity Factor (TCF);*
  - *Determinazione del Function Point (FP).*
- Vengono classificati, a tre livelli di complessità, i seguenti tipi di indici:
  - *input esterno: informazione distinta fornita dall'utente e utilizzata dal programma come dato di ingresso;*
  - *output esterno: output distinto che il programma ritorna all'utente come risultato delle proprie elaborazioni;*
  - *file logico interno: file creato ed utilizzato internamente al programma;*

Function Points

  - interfaccia esterna*: file o altri insiemi di dati scambiati dal programma con altri programmi;*
  - interrogazione esterna*: interrogazione in linea che produce una risposta immediata del sistema.*
- Questi fattori sono le manifestazioni esterne di ogni applicazione che coprono tutte le loro funzioni.

INDICE	SEMPLICE	MEDIO	COMPLESSO
N. INPUT	3	4	6
N. OUTPUT	4	5	7
N. INTERROGAZIONI	3	4	6
N. FILES	7	10	15
N. INTERFACCE ESTERNE	5	7	10

Function Points

- FUNCTION COUNT (FC) (Unadjusted Function Point (UFP))
  - *FC = S (VALi*PESOi)*
    i=1..5
  - *VALi = rappresenta il numero fornito dal progettista relativo agli elementi dell'indice i-esimo;*
  - *PESOi =rappresenta il valore pesato relativo all'indice i-esimo*
- TECHNICAL COMPLEXITY FACTOR (TCF) (Adjusted Factor (AF))
  - *TFC = 0.65 + 0.01*DI*
  - *DI = rappresenta il grado di influenza totale del programma*

Function Points

CARATTERISTICA	DESCRIZIONE
1. Data Communications	E' richiesta la trasmissione di dati?
2. Distributed Data Processing	Vi sono funzionalità che richiedono elaborazioni distribuite?
3. Performance	Le prestazioni sono critiche?
4. Heavily Used Configuration	Il programma funzionerà in un ambiente operativo pesantemente utilizzato?
5. Transaction Rate	La quantità di transazioni gestita è alta?
6. On-Line Data Entry	Il sistema richiede funzionalità avanzate per l'emissione e la consultazione in linea dei dati?
7. End User Efficiency	Il sistema ha particolari necessità di efficienza per l'utente?
8. On-Line Update	Gli archivi principali sono aggiornati in tempo reale?
9. Complex Processing	Il sistema richiede elaborazioni complesse?
10. Reuseability	Il sistema deve essere in parte riusabile?
11. Installation Ease	Il sistema richiede facility di installazione e conversione?
12. Operational Ease	Il sistema richiede particolari procedure di recovery e backup?
13. Multiple Sites	Il sistema deve installato presso diversi utenti?
14. Facilitate Changes	Il sistema deve essere facilmente modificabile per venire incontro alle esigenze dell'utente?

Function Points

- GRADO INFLUENZA
  - *0 Non presente o non influente se presente.*
  - *1 Insignificante.*
  - *2 Moderata.*
  - *3 Media.*
  - *4 Significativa.*
  - *5 Ovunque forte.*
  - $DI = \sum_{i=1..14} (DI\ i)$
    - DI i = rappresenta il grado di influenza della caratteristica generale i-esima.

Ingegneria del Software
Progettazione e Laboratorio
Metriche del Software
Maurizio Pighin

Function Points

- Per calcolare il Function Point di un programma si esegue la seguente moltiplicazione:
  - *FP = FC*TFC*
  - *FC = rappresenta il Function Count;*
  - *TFC = rappresenta il Technical Complexity Factor;*

Function Points

- Problemi con i Function Counts (FC).
  - *la determinazione dei FC tende a considerare il sistema come un black box;*
  - *gli indici definiti non possono essere completamente appropriati per la tecnologia corrente;*
  - *la classificazione di questi indici in semplice, media e complessa ha il merito di essere semplice e chiara ma appare essere più che semplificata, infatti un sistema di componenti avente più di 100 elementi e quindi considerato complesso, ha una complessità al più il doppio di quella di un sistema costituito da un solo elemento e dunque considerato semplice;*
  - *la scelta dei pesi è determinata da sperimentazioni ed è stata giustificata da Albrecht come riflettente il relativo valore della funzione per l'utente/cliente;*
  - *il FC non è una misura tecnologicamente indipendente dalla dimensione.*

Function Points

- Problemi con i Technical Complexity Factors (TCF)
  - *la restrizione a 14 caratteristiche generali non è stata fatta per essere soddisfacente per sempre. Altri fattori possono essere suggeriti ora o in futuro, dunque è desiderabile un approccio migliore*
  - *queste 14 caratteristiche generali non sono incorporate nella metrica dei Function Points in maniera adeguata;*
  - *queste caratteristiche contribuiscono minimamente alla determinazione della stima dei costi*
  - *i gradi di influenza di ognuna delle 14 caratteristiche sono ristretti in un intervallo da 0 a 5 e ciò risulta essere una classificazione semplice ma non sempre valida.*

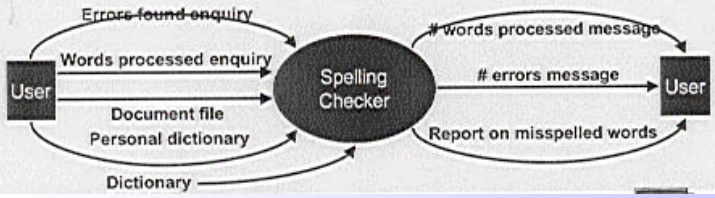
Esempio di calcolo di FP

FP: un esempio

- Descrizione dello Spelling Checker:

  "The checker accepts as inputs a document file and an optional personal dictionary file. The checker lists all words not contained in either of these files. The user can query the number of words processed and the number of spelling errors found at any stage during processing".

- Diagramma di flusso dei dati per lo SpellingChecker:



Esempio di calcolo di FP

Esempio FP: calcolo UFP

- External inputs:
  - document filename
  - personal dictionary name
- External outputs:
  - misspelled word report
  - number of words processed message
  - number of errors so far message
- External inquiries:
  - words processed
  - errors so far
- External files:
  - document file
  - personal dictionary
- Internal file:
  - dictionary
- Riepilogando:
  - 2 external inputs
  - 3 external outputs
  - 2 external inquiries
  - 2 external files
  - 1 internal file

Esempio di calcolo di FP

Esempio FP: calcolo UFP

- Ipotesi:
  - Dictionary file, misspelled words report: complessità elevata
  - Tutto il resto: complessità media
- Calcolo di UFC:

UFC = Σ (numero elementi di tipo i) x (peso del tipo i)

UFC = (2 external input medi) x 4 + (2 external output medi) x 5 + (1 external output complesso) x 7 + (2 external inquiries medi) x 4 + (2 external file medi) x 10 + (1 internal file complesso) x 10

UFC = 2 x 4 + 2 x 5 + 1 x 7 + 2 x 4 + 2 x 10 + 1 x 10 =

63 UFP

Esempio di calcolo di FP

Esempio FP: fattori di complessità

- Technical complexity factors: TCF = 0.65 + 0.01 Σ Fi
- Ipotesi:
  - F3, F5, F9, F11, F12 e F13 = 0
  - F1, F2, F6, F7, F8 e F14 = 3
  - F4 (performance) e F10 (complex processing) = 5
- TCF = 0.65 + 0.01 x (6 x 0 + 6 x 3 + 2 x 5) = 0.93

Esempio FP: calcolo finale

- FP = UFC x TCF = 63 x 0.93 = 59 Function Points

Esempio di calcolo di FP

Valutazione dei FP

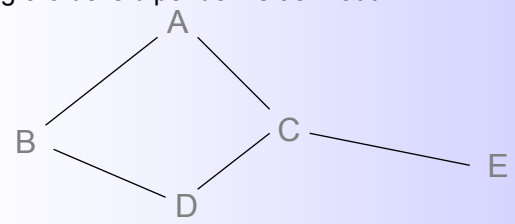
- Come è stata definita la modalità di conteggio?
  - Analisi statistica di 63 progetti completati
  - Risentono delle caratteristiche dei progetti , dominio applicativo, ...
  - "Taratura"in base allo specifico contesto
- Come si usano i FP?
  - A partire dalle specifiche , o da una descrizione sommaria
  - Problemi di soggettività , ripetibilità , accuratezza
  - Uso "ragionato", evitando di usarli come unico elemento di giudizio

Internal product attributes: Complexity

- Complexity
  - *problem*
    - of the problem itself: resource required for optimal solution
    - of the solution: resorces needed to implement a solution
      - *time (computer time)*
      - *space (computer memory)*
  - *alghorithmic complexity (asymptotic behaviour)*
  - *cognitive complexity*
  - *structural complexity*

Structural Complexity

- Control flow structure (graphs)
  - *Modularity*
    - Descrive il modo in cui un programma è suddiviso in moduli
    - Possibili misure del livello di modularizzazione
      - *Lunghezza media di un modulo*
      - *Num. moduli/Num. procedure*
      - *Num. moduli/Num. variabili*
    - Il grafo delle dipendenze tra moduli



Structural Complexity

- Misure di modularizzazione basate sull'osservazione del grafo delle dipendenze
  - *Dimensione: numero di nodi e numero di archi*
  - *Profondità: lunghezza del più lungo cammino dalla radice ad una delle foglie*
  - *Larghezza*
  - *Densità di connessione: rapporto tra numero di nodi e numero di archi*

Structural Complexity

- *Reuse*
  - Distinguiamo tra:
    - *Riuso pubblico: di codice prodotto da altri o in altri progetti (già discusso)*
    - *Riuso privato: di codice all'interno dello stesso progetto*
      - » *Siamo qui interessati a misurare il secondo tipo di riuso. Il grafo delle dipendenze ci dice che un certo modulo ne usa altri*
      - » *Non ci dice quante volte li usi (una sola chiamata o molte) Misure significative dovrebbero tenere conto del numero di chiamate e della dimensione dei moduli*

Structural Complexity

- *Coupling (interdependences)*
  - L' accoppiamento tra due moduli può essere
    - *Nullo: non c' è alcuna relazione tra i due moduli*
    - *Relativo ai dati: se i due moduli "comunicano" attraverso dati scambiati come parametri o valori di ritorno di mutue invocazioni*
    - *Relativo ai tipi: se i due moduli usano lo stesso tipo di dato (record o struct) come parametro*
    - *Relativo al controllo: se un modulo passa all' altro come parametro un' informazione che influisce sul flusso di controllo di questo*
    - *Relativo all' ambiente globale: se i due moduli comunicano attraverso variabili globali*
    - *Relativo al contenuto: se un modulo modifica variabili di un altro o "vi salta dentro"*
  - Misura (scala ordinale) dell'accoppiamento tra moduli
    - *L'accoppiamento complessivo di un sistema può essere misurato come valore mediano dell' accoppiamento dei suoi moduli quando ad ogni tipo di accoppiamento si sia dato un valore numerico (da 0 a 5)*

Structural Complexity

- *Cohesion (individual component needed to perform a precise task or not)*
  - La coesione è un attributo interno al singolo modulo, classificabile (in ordine decrescente) come:
    - *Funzionale: il modulo realizza un'unica funzione concettuale*
    - *Sequenziale: il modulo racchiude più funzionalità ma tutte invocate in sequenza dal sistema*
    - *Dati: il modulo racchiude più funzioni che operano sugli stessi dati*
    - *Procedurale: il modulo racchiude più funzioni tutte collegate ad una procedura generale del software*
    - *Temporale: il modulo racchiude più funzioni che vengono invocate dal software nello stesso "periodo di tempo"*
    - *Logico: il modulo racchiude più funzioni aventi una certa relazione logica*
    - *Accidentale: il modulo racchiude più funzioni senza alcuna relazione tra loro*
  - Sulla base della classificazione precedente è possibile definire una misura di scala ordinale

Ingegneria del Software
Progettazione e Laboratorio
Metriche del Software
Maurizio Pighin

Structural Complexity

- *Information flow*
  - fan-in of a module M is the number of information flows that terminates in M plus the number of data structures from which information is retrieved in M
  - fan-out of a module M is the number of information flows that emanate from M plus the number of data-structures which are updated in M

Complessità: la metrica di McCabe

- Numero Ciclomatico (1976)
- Basato sul numero di cicli presenti sul grafo del programma
- Analizzare il max insieme dei cammini linearmente indipendenti

La metrica di McCabe

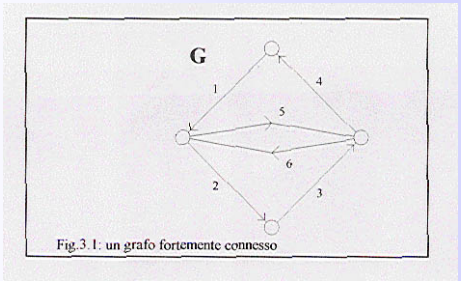


Fig.3.1: un grafo fortemente connesso

La metrica di McCabe

Avendo il grafo in figura 6 archi, è necessario un vettore con 6 componenti. Ogni cammino può essere rappresentato come una sestupla:

<1, 2, 3, 4 > = (1 1 1 1 0 0)

<1, 5, 6, 2, 3, 4, 1 > = (2 1 1 1 1 1)

cioè la i-esima posizione nel vettore è il numero di volte che l'arco i compare nel cammino.

Un cammino p è definito combinazione lineare dei cammini $p_1,.....p_n$ se ci sono degli interi $a_1,.....,a_n$ tali che

$p=\Sigma a_i p_i$

nella rappresentazione vettoriale.

Quindi il cammino <1, 2, 3, 4, 5, 6, 2 > è una combinazione dei cammini <1, 2, 3, 4 > e <5, 6, 2 > poiché

(1 2 1 1 1 1) = (1 1 1 1 0 0) + (0 1 0 0 1 1)

Un insieme di cammini è *linearmente indipendente* se nessun cammino dell'insieme è una combinazione lineare degli altri cammini dell'insieme stesso.

La metrica di McCabe

- Base di cicli è il più grande insieme di cicli linearmente indipendenti
- Ogni cammino è una combinazione lineare di elementi appartenenti alla base
- Si dimostra che la cardinalità del grafo è data da
  - *e-n+1 (e=arco n=nodo)*

La metrica di McCabe

- Complessità NC=e-n+2

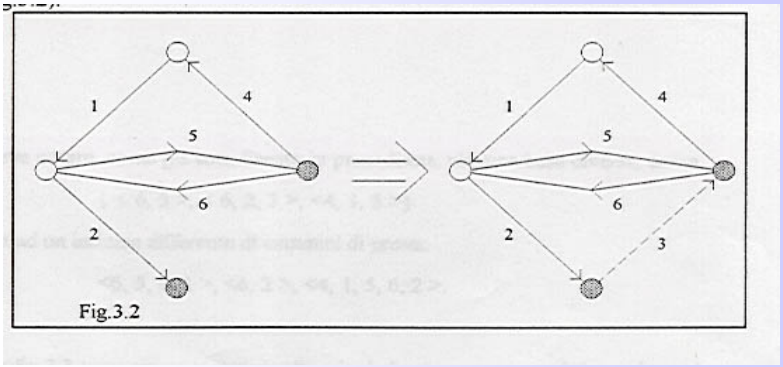


Fig.3.2

La metrica di McCabe

- Gli elementi fondamentali

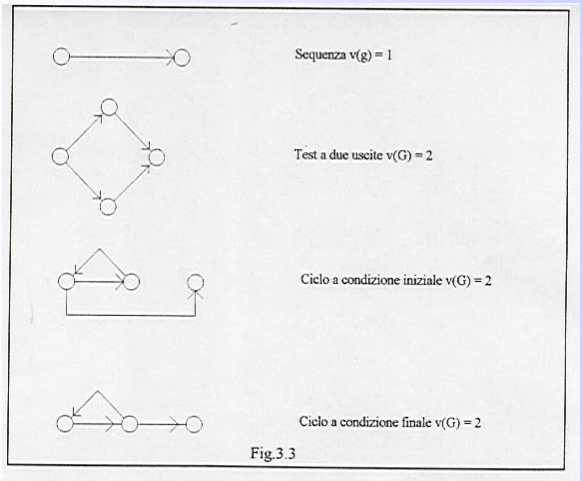


Fig.3.3

La metrica di McCabe

- Teorema di Mills
  - *NC = d+1 (d è il n° punti decisione)*
  - *Basta quindi calcolare con un parser i punti di decisione presenti per calcolare il Numero Ciclomatico*

La metrica di McCabe

- Critiche
  - *complessità computazionale e complessità psicologica*
  - *indipendenza dai dati*
  - *indipendenza da altri fattori quali*
    - istruzioni imperative
    - leggibilità codice
    - numero variabili
    - ecc..

External product attributes

- Quality of software products
  - *fitness for purpose*
  - *conformance to specification*
  - *degree of excellence*
  - *timeliness (attualità)*
- Modelli di qualità di McCall/ Bohem e l' ISO 9126
  - *vedremo in dettaglio nel capitolo relativo alla Qualità*

External product attributes

- Defect based quality measures
  - *defect density measures*
    - DD=number_known_defects/product_size
    - MTTF
  - *system_spoilage= time_to_fix_post-release_defects/ total_system_developing_time*
- Usability measures
  - *usability is the extent to which the product is convenient and practical to use*
  - *the probability that the operator of a system will not experience a user interface problem during a given period of operation under a given operational profile*

External product attributes

- *evidence of good usability*
  - well-structured manuals
  - good use of menus and graphics
  - informative error message
  - help function
  - consistent interface
- *external view of usability*
  - entry level (respect similar classes of application)
  - learnability (speed of learning)
  - handling ability (speed of working)
  - D =PD/SLOC
    - *PD = rappresentano le pagine di documentazione;*
    - *D = rappresenta il livello di documentazione del software.*

External product attributes

- Maintainability measures
  - *mean time to repair (MTTR)*
    - problem recognition time
    - administrative delay time
    - maintainance tools collection time
    - problem analyis time
    - change specification time
    - change time (including testing and review)
  - *total_change_implementation_time/total_number_changes*
  - *number unresolved problems*
  - *time spent on unresolved problems*
  - *percentage of changes that introduced faults*
  - *number of modules modified to implement a change*
  - *DE=CLOC/LOC*
    - DE = rappresenta la densità commento

Resource Measurement

- Productivity
  - *the COCOMO model (basic and intermediate)*
  - *size/effort*
    - lines_of_code/person_months
    - number_function_points/person_months

Ingegneria del Software
Progettazione e Laboratorio
Metriche del Software
Maurizio Pighin

Resource Measurement

Table 11.3: Productivity ranges and modes for selected software project sizes (Jones, 1991)

Productivity rates in function points per person month	100 function points	1000 function points	10 000 function points
> 100	1.0%	0.01%	0.0%
75–100	3.0%	0.1%	0.0%
50–75	7.0%	1.0%	0.0%
25–50	15.0%	5.0%	0.1%
15–25	40.0%	10.0%	1.4%
5–15	25.0%	50.0%	13.5%
1-5	10.0%	30.0%	70.0%
< 1	4.0%	4.0%	15.0%

Resource Measurement

Table 11.4: Approximate number of source statements required to code one Albrecht Function Point (Behrens, 1983)

Language	Source statements per function point
Assembler	320
C	150
ALGOL	106
COBOL	106
FORTRAN	106
Pascal	91
RPG	80
PL/1	80
Modula-2	71
Prolog	64
LISP	64
BASIC	64
4-GL for databases	40
APL	32
Smalltalk	21
Query languages	16
Spreadsheet languages	6

Data Collection

- Good data
  - *correctness*
  - *accuracy*
  - *precision*
  - *consistent*
  - *associated with particular activity or time period*
  - *can be replicated*

Data Collection

- How to collect data
  - *keep procedure simple*
  - *avoid unnecessary recording*
  - *train staff in need to record data and in the procedure to be used*
  - *provide the results of data capture and analysis to the original providers promptly and in a useful form that will assist them in their work*
  - *validate the data collected at a central collection point*

Data Collection

- How to analyse data
  - *measure of central tendency (mean, median,...)*
  - *measure of dispersion (std. dev., scatter plot, ...)*
  - *distribution (normal, ...)*
  - *confidence limits*
  - *correlation*
  - *.... other more sophisticated statistical analysis*