Qualità del Software

Riferimenti

- I. Sommerville – Ingegneria del Software – 8a edizione – Cap. 27
- R. Pressman – Principi di Ingegneria del Software – 5a edizione italiana - Cap. 17
- Ghezzi, Jazayeri, Mandrioli, Ingegneria del Software, 2a edizione, Capitolo 2

- International Standard ISO/IEC 9126

Che cos'è la Qualità del Software?

- La qualità del software può essere definita come:
  - conformità ai **requisiti funzionali** e prestazionali enunciati esplicitamente,
  - agli **standard di sviluppo** esplicitamente documentati,
  - a **caratteristiche implicite** che è lecito aspettarsi da un prodotto professionale [Pressman]
- Tre punti fondamentali:
  - I requisiti sono il fondamento su cui misurare la qualità;
  - Gli standard di qualità definiscono I criteri da seguire nello sviluppo software;
  - Esistono requisiti impliciti (spesso taciuti) la cui assenza compromette la qualità del software.

Che cos'è la Qualità del Software?

- Alcuni problemi nella definizione di qualità introdotta...
  - Le specifiche software sono in genere incomplete e spesso inconsistenti;
  - Alcuni requisiti di qualità sono difficili da specificare in maniera non ambigua;
  - Può esistere contrapposizione fra i requisiti di qualità attesi dal cliente (efficienza, affidabilità, etc.) e quelli dello sviluppatore (manutenibilità, riusabilità, etc.);
  - Alcuni requisiti di qualità sono difficili da valutare: non possiamo valutarli *direttamente*, ma soltanto *indirettamente*

Il problema fondamentale della qualità del software

Non possiamo valutare la qualità del software in assoluto, ma solo alcune sue manifestazioni!

- Ciò equivale a dire che non possiamo valutare direttamente la qualità del software, ma indirettamente, attraverso la valutazione di attributi che si correlano a questa, supponendo che la relazione tra la qualità e questi attributi sia valida!

- Ad esempio: non posso misurare l'usabilità e la manutenibilità in assoluto, ma debbo riferirmi ad altri attributi misurabili correlati ad esse.
- Abbiamo dunque bisogno di modelli di qualità condivisi !

Come caratterizzare la Qualità del Software?

- La qualità di un prodotto software si caratterizza attraverso un insieme finito e definito di attributi
  - *ragionevolmente esaustivi*
    - in modo che per una qualsiasi richiesta di caratteristica di qualità sia possibile associarvi un sottoinsieme degli attributi definiti in modo da poterla valutare
  - *privi di reciproche sovrapposizioni:*
    - per evitare che più attributi riguardino la stessa caratteristica del software

Modello di Qualità (MQ) del software

- E' un insieme di attributi del software che fornisce uno **schema di riferimento** che, con una opportuna distribuzione di pesi per ciascun attributo, va adeguato e tarato per la rappresentazione dei requisiti di qualità **desiderati** dal committente o **posseduti** dal software.

Struttura dei modelli di qualità

- I modelli di qualità del software pubblicati in letteratura sono tipicamente *gerarchici*, a *n* livelli.
  - Il primo livello descrive un insieme di caratteristiche (***proprietà***), che, nel loro complesso, rappresentano la qualità del prodotto software, eventualmente secondo diversi punti di vista.
  - Le proprietà (in genere qualitative, astratte) sono precisate attraverso degli ***attributi*** misurabili, quantitativi (in genere da una combinazione di attributi).
  - Il grado di possesso che il software ha di questi attributi può essere valutato su una scala di riferimento, facendo ricorso ad opportune ***metriche*** ed a meccanismi di *rating.*

Esempio: I Modelli di McCall e Boehm

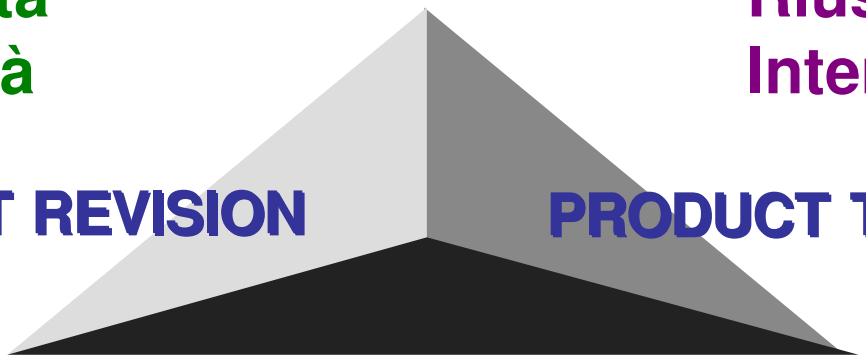
- I primi modelli di qualità del software sono stati sviluppati negli anni '70 da McCall (*Factor-Criteria-Model*, 1977) e da B. Boehm (1978). Hanno un'architettura a più livelli. Ad es. McCall prevede:
  - ***Fattori***, che descrivono il software da un punto di vista esterno, quello degli utenti; i fattori corrispondono a requisiti specificati dal cliente.
  - ***Criteri***, che descrivono gli elementi su cui agiscono gli sviluppatori per soddisfare i requisiti del cliente.
  - ***Metriche***, che servono a controllare che i criteri sviluppati corrispondano ai fattori specificati. Vengono utilizzate dagli auditors e/o dagli addetti alle verifiche.

Modello di McCall (1977)

3 settori e 11 fattori:

Manutenibilità
Flessibilità
Testabilità

Portabilità
Riusabilità
Interoperabilità



PRODUCT REVISION

PRODUCT TRANSITION

PRODUCT OPERATION

Correttezza **Usabilità** **Efficienza**
Affidabilità **Integrità**

Modelli di Qualità

- La difficoltà principale nell'adozione di un modello di qualità consiste nel riuscire a dare un valore (o un giudizio) a tutti gli attributi di qualità (esterni) proposti, in funzione di attributi di qualità che siano direttamente misurabili...
  - La cui relazione reciproca sia stata empiricamente validata
  - In cui gli attributi misurabili siano supportati da strumenti efficaci per la misurazione (non dimentichiamo I costi della misurazione)!

- Affinchè un modello possa essere affidabile e possa essere accettato, è auspicabile che esso provenga da un ente di standardizzazione.
  - Al momento, il modello più comunemente adottato è lo standard ISO 9126

LO STANDARD ISO ISO/IEC 9126-

Software engineering-Product Quality

Standard ISO/IEC 9126- *Software engineering- Product Quality*

È suddiviso in 4 parti:

1. Quality Model
   - *un insieme di caratteristiche di qualità che possano essere in grado di descrivere I principali fattori di qualità di un prodotto software*
1. External Metrics
   - *Un insieme di metriche indirette attraverso le quali sia possibile valutare la conformità di un prodotto software al modello di qualità*
1. Internal Metrics
   - *Un insieme di metriche direttamente misurabili che possano essere utilizzate allo scopo di valutare le External Metrics*
1. Quality In Use Metrics
   - *Metriche dirette rivolte alla valutazione del sottoinsieme di caratteristiche di qualità legate all'utente*

I punti di vista sulla qualità

A determinare la qualità complessiva di un prodotto software concorrono 3 punti di vista.

- ESTERNA
  - Esprime il comportamento dinamico del software, nell'ambiente d'uso.
- INTERNA (intrinseca)
  - Esprime la misura in cui il codice software possiede una serie di attributi statici, indipendentemente dall'ambiente di utilizzo e dall'utente.
- PERCEPITA (in uso)
  - Esprime l'efficacia ed efficienza con cui il software serve le esigenze dell'utente, ed è correlata alla percezione diretta dell'utente.

La qualità esterna

- **La qualità esterna è quella rappresentata dalle prestazioni del prodotto e dalle funzionalità che offre (il prodotto è visto come una *black box* da testare).**
  - In sostanza, riguarda il comportamento "dinamico" del software in un dato ambiente operativo.
  - Va ricordato che il software non "funziona" mai da solo, ma è sempre parte di un ambiente (*environment*) che può contenere hardware, persone, processi etc…
  - Le caratteristiche di qualità esterne del software lo qualificano in relazione a questo ambiente e permettono di osservarne il comportamento mentre è utilizzato operativamente.

La qualità interna

- **La qualità interna rappresenta le proprietà intrinseche del prodotto (quelle misurabili direttamente sul codice sorgente, sul suo flusso di controllo). Si realizza a partire da:**
  - I requisiti di qualità dell'utente (External Quality Requirements), che rappresentano le specifiche di qualità così come le dà l'utente, fornendo il primo input alla progettazione,
  - Le specifiche tecniche (Internal Quality Requirements), che rappresentano la qualità richiesta dall'utente tradotta dallo sviluppatore nell'architettura del software, nella struttura del programma, nelle interfacce del software verso l'utente.

La qualità in uso

- La qualità in uso riguarda il livello con cui il prodotto si dimostra **utile all'utente nel suo effettivo contesto d'utilizzo**, in particolare la capacità del prodotto di dare efficacia ed efficienza al lavoro dell'utente, a fronte di una sicurezza di utilizzo e di una soddisfazione nel far uso del prodotto.
  - In sostanza, è una misura della interazione tra utente e prodotto, in un determinato contesto d'uso.

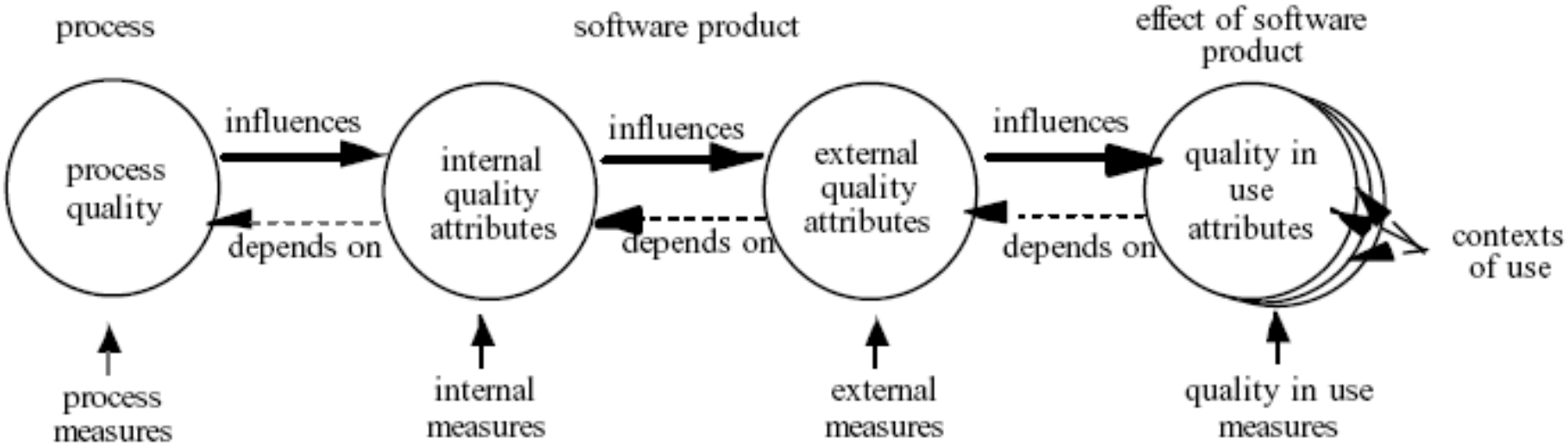
- I tre punti di vista sulla qualità si influenzano a vicenda: è chiaro che non può esservi qualità percepita positivamente dall'utente senza che vi sia una buona qualità intrinseca del codice e buone prestazioni!

Approccio ISO alla qualità

- **La** qualità del processo **contribuisce a migliorare la qualità del prodotto, influenzando direttamente i valori degli attributi interni di qualità.**
- **Gli attributi di** qualità interni **influenzano insiemi di attributi di** qualità esterni**.**
- **Gli attributi di** qualità esterni **influenzano gli attributi di** qualità in uso **(percepiti dall'utente)**
- **In definitiva, migliorare la qualità del processo di sviluppo software e del prodotto software fa migliorare la qualità percepita dall'utilizzatore.**

Qualità del software nel ciclo di vita

La qualità del software, come percepita dall'utente, si determina progressivamente attraverso una sequenza logica di azioni, lungo il ciclo di vita.



I Modelli di Qualità dello standard ISO

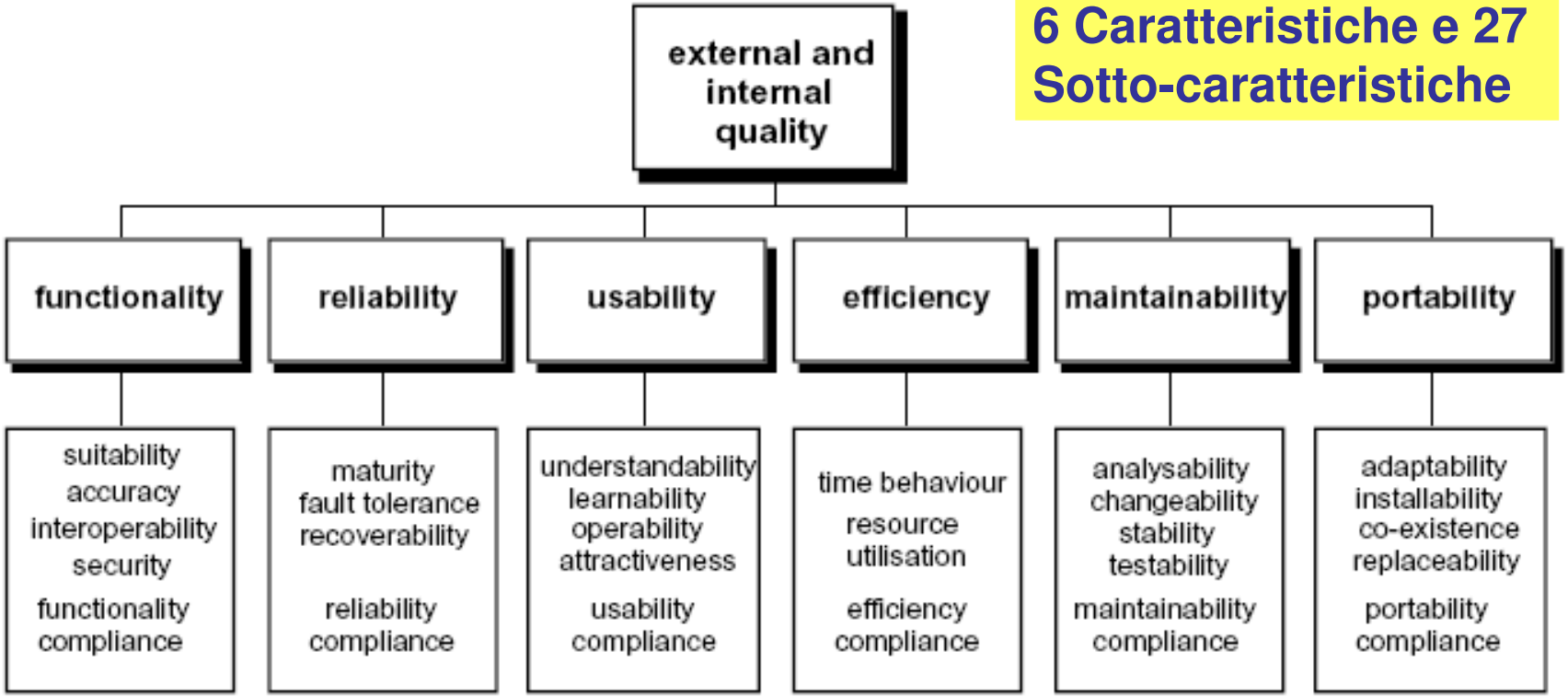
- Per descrivere le tre viste sulla qualità, lo standard propone ***due modelli***, uno per la qualità interna ed esterna, ed uno per la qualità in uso.

- Entrambi i modelli definiscono la qualità in termini di un insieme di ***caratteristiche di primo livello*** a cui sono associati insieme di sottocaratteristiche di ***secondo livello*** che meglio descrivono ciascuna caratteristica.

- Le caratteristiche di secondo livello saranno valutate in funzione di un ampio insieme di ***metriche*** sia interne che esterne.

Il Modello di Qualità ISO- 9126 per la qualità interna ed esterna

6 Caratteristiche e 27 Sotto-caratteristiche



le sub-caratteristiche possono essere misurate con metriche interne o esterne

1a Caratteristica: Funzionalità (functionality)

- La capacità del prodotto software di fornire funzioni che soddisfano esigenze stabilite ed implicite quando il software è usato sotto condizioni specificate

  - **Appropriatezza** (suitability): la capacità del prodotto software di fornire un appropriato insieme di funzioni all'utente per i compiti e gli obiettivi specificati.
  - **Accuratezza** (accuracy): la capacità del prodotto software di fornire i giusti o concordati risultati o effetti, con la precisione richiesta.
  - **Interoperabilità** (interoperability): la capacità del prodotto software di interagire con uno o più sistemi specificati
  - **Sicurezza** (security): la capacità del prodotto software di proteggere informazioni e dati in modo che persone o sistemi non autorizzati non possano leggere o modificarli e che a persone o sistemi autorizzati non sia negato l'accesso ad essi
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni o regolamentazioni in leggi e prescrizioni relative alla funzionalità.

2a Caratteristica: Affidabilità (reliability)

- La capacità del prodotto software di mantenere uno specificato livello di prestazioni quando usato sotto condizioni specificate

  - **Maturità** (maturity): la capacità del prodotto software di evitare malfunzionamenti, quali risultati di anomalie nel software.
  - **Tolleranza all'errore** (fault tolerance): la capacità del prodotto software di mantenere uno specificato livello di prestazioni in caso di anomalie software o di violazione delle sue specificate interfacce.
  - **Recuperabilità** (recoverability): la capacità del prodotto software di ristabilire uno specificato livello di prestazioni e di ripristinare i dati direttamente intaccati in caso di malfunzionamenti .
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni o regolamentazioni relativamente all'affidabilità.

3a Caratteristica: Usabilità (usability)

- La capacità del prodotto software di essere capito, appreso, usato e gradito all'utente, quando usato sotto condizioni specificate

  - **Comprensibilità** (understandability): la capacità del prodotto software di mettere in grado l'utente di comprendere se il software è appropriato, e come esso possa essere usato per particolari compiti e condizioni d'uso.
  - **Apprendibilità** (learnability): la capacità del prodotto software di permettere all'utente di imparare ad usare l'applicazione.
  - **Operabilità** (operability): la capacità del prodotto software di permettere all'utente di operare con esso e di controllarlo.
  - **Attrattività** (attactiveness): la capacità del prodotto software di essere attraente all'utente (cioè avere un livello di gradimento nell'utilizzo).
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni, stili guida o regolamentazioni relativamente all'usabilità.

4a Caratteristica: Efficienza (efficiency)

- La capacità del prodotto software di fornire appropriate prestazioni relativamente alla quantità di risorse usate, sotto condizioni stabilite

  - **Comportamento rispetto al tempo** (time behaviour): la capacità del prodotto software di fornire un appropriato responso e tempi di elaborazione e velocità di 'attraversamento' nell'eseguire le sue funzioni sotto specificate condizioni.
  - **Utilizzo di risorse** (resource utilisation): la capacità del prodotto software di usare appropriate quantità e tipo di risorse quando il software esegue le sue funzioni sotto specificate condizioni.
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni o regolamentazioni relativamente all'efficienza.

5a Caratteristica: Manutenibilità (maintainability)

- La capacità del prodotto software di essere modificato. Le modifiche possono includere correzioni, miglioramenti o adattamenti del software per cambiamenti nell'ambiente operativo, nei requisiti e nelle specifiche funzionali.

  - **Analizzabilità** (analysability): la capacità del prodotto software ad essere diagnosticato per deficienze o cause di malfunzionamenti nel software o per l'identificazione delle parti da modificare.
  - **Modificabilità** (changeability): la capacità del prodotto software di permettere l'implementazione di una specificata modifica
  - **Stabilità** (stability): la capacità del prodotto software di evitare effetti inaspettati derivanti da modifiche ad esso
  - **Testabilità** (testability): la capacità del prodotto software di permettere a software modificato di essere validato
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni o regolamentazioni relativamente alla manutenibilità.

6a Caratteristica: Portabilità (portability)

- La capacità del prodotto software di essere trasferito da un ambiente ad un altro.

  - **Adattabilità** (adaptability): la capacità del prodotto software di essere adattato per differenti e specificati ambienti senza dover applicare altre azioni o mezzi diversi da quelli forniti per tale scopo per il software considerato.
  - **Installabilità** (installability): la capacità del prodotto software di essere installato in uno specificato ambiente.
  - **Coesistenza** (co-existence): la capacità del prodotto software di coesistere con altri software indipendenti in un ambiente comune condividendo risorse comuni .
  - **Sostituibilità** (replaceability): la capacità del prodotto software di essere usato al posto di un altro specificato prodotto software per gli stessi scopi e nello stesso ambiente
  - **Conformità** (compliance): la capacità del prodotto software di aderire a standard, convenzioni o regolamentazioni relativamente alla portabilità.

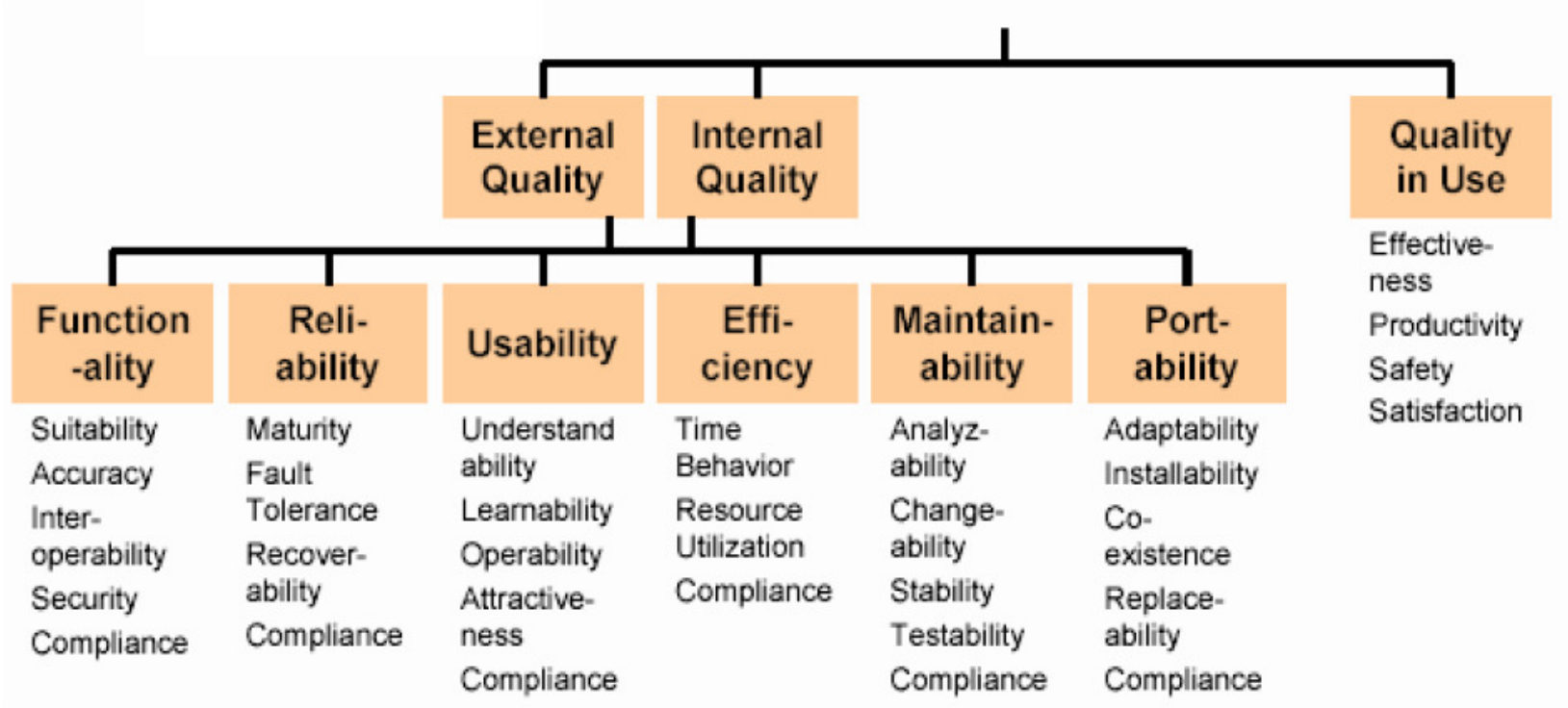
Il Modello di Qualità per la Qualità in uso

- La **qualità in uso** è rappresentata da 4 caratteristiche, che rappresentano il punto di vista dell'utente sulla qualità del software.
- Rappresenta la capacità del software di supportare specifici utenti a raggiungere determinati obiettivi, con efficacia, produttività, soddisfazione e sicurezza personale, in determinati contesti d'uso.

Caratteristiche della qualità in uso

- **Efficacia**, la capacità di supportare un utente nel raggiungere i suoi obiettivi con accuratezza e completezza in un dato contesto.
- **Produttività**, la capacità di supportare un utente nello spendere l'appropriata quantità di risorse in relazione all'efficacia dei risultati da raggiungere.
- **Soddisfazione**, la capacità di soddisfare un utente in un dato contesto d'uso.
- **Sicurezza**, la capacità di raggiungere accettabili livelli di rischio di danni a persone, al software, ad apparecchiature, o all'ambiente operativo in un dato contesto d'uso.

Riepilogo modello 9126



Caratteristiche Interne

- **Sono collegate alle caratteristiche esterne di 2° livello.**
- **Alcune caratteristiche interne vanno ad influenzare più di una caratteristica esterna di 2° livello.**

Completeness		
Access control	Informativeness	Self-descriptiveness
Instrumentability		Expressiveness
Data-commonality		Self-containedness
Communication-commonality	Well-equipmentness	
Traceability	Timeliness	
Robustness	Integrity	Modularity
Simplicity	Coherency	
Accessibility	Uniformity	Accuracy
Hierarchieness	Consistency	
Metaphorability	Attractiveness	
Access audit	Memorability	
Conciseness	Choosability	Guideability

Caratteristiche Interne

ad esempio, per la funzionalità:

- suitability = f(completeness, traceability, consistency, self-descriptiveness, coherency)
- accurateness = f(completeness, traceablity, consistency, self-descriptiveness, coherency)
- inter-operability = f(data-commonality, communication-commonality, accessibility)
- compliance =f(accuracy, data-commonality, accessibility)
- security =f (access control, access audit, robustness)