La qualità del software secondo il modello ISO/IEC 9126

Modello, metriche e modalità di utilizzo nei progetti reali della qualità interna, esterna ed in uso del software

Ercole F. Colonese

Versione 2.0 – Giugno 2006

Note sulla versione 2.0

Nella presente versione è stato aggiunto:
1) descrizione delle metriche relative alla qualità interna, esterna ed in uso,
2) modalità di implementazione del modello nei progetti reali,
3) esempi di applicazione del modello a progetti reali.

Versione del documento

2005	2006	2007	2008	2009
v1.0	V2.0			

Sommario

1 La qualità del prodotto software

Il concetto di qualità del software si è evoluto nel tempo includendo, di volta in volta, varie esigenze importanti ai fini del corretto funzionamento del prodotto e del suo utilizzo da parte degli utenti. L'utilizzo del software si è diffuso sempre di più in tutte le aree applicative, divenendo sempre più rilevante e critico per il business e la sicurezza. Questo pone il problema della sua adeguatezza alle necessità d'utilizzo nei vari contesti previsti. Sviluppare o selezionare prodotti software di qualità adeguata all'utilizzo che se ne farà, rappresenta quindi un'attività rilevante. Per raggiungere l'obiettivo, occorre definire caratteristiche e metriche che permettano di valutare in maniera oggettiva e precisa il livello di qualità del prodotto software. Ma per essere efficaci, tali caratteristiche devono tener conto dello scopo e del contesto di utilizzo del prodotto:

1) *Profilo degli utenti del prodotto*: descrive le caratteristiche degli utenti in termini di livello scolastico, competenza informatica, dimestichezza con prodotti software analoghi, esperienza maturata nel settore specifico, ogni altra informazione che può aiutare a identificare la tipologia di utente.
2) *Obiettivi da raggiungere con l'utilizzo del prodotto*: descrive quali motivazioni sottendono l'utilizzo del prodotto come, ad esempio, aumentare la produttività nell'immissione di pratiche in back-office, semplificare la gestione di una pratica, ridurre i tempi di risposta di una transazione di inquiry, ecc.
3) *Modalità operative*: descrive come abitualmente sono svolte le attività per completare i compiti del lavoro quotidiano.

Ogni caratteristica rilevante del prodotto, quindi, sarà misurata applicando metriche adeguate e sarà valutata in base ad intervalli di accettabilità e valori di soglia stabiliti.

Semplificando, la qualità di un prodotto software è valutata secondo criteri semplici e comprensibili per tutti, utenti e sviluppatori, operatori e addetti alla manutenzione. Esempi semplici e comprensibili di queste caratteristiche sono: l'adeguatezza delle funzionalità alle necessità degli utenti, la facilità con cui le funzioni possono essere comprese ed utilizzate dagli utenti, la facilità con cui il prodotto può essere installato e personalizzato in ambienti e piattaforme diverse, le prestazioni fornite sia in termini di tempi di risposta che di utilizzo delle risorse, l'affidabilità, la capacità di evolvere nel tempo e di facilitare la manutenzione, il livello di sicurezza garantito rispetto agli accessi

non autorizzati, alla salvaguardia dei dati ed alla sicurezza fisica delle persone e dei beni. Queste sono solo alcune delle principali caratteristiche di un buon prodotto software.

Le caratteristiche di qualità del software rappresentano quindi uno strumento utile ed efficace per capire le necessità degli utenti, per progettare e sviluppare il software più adatto, per misurare la qualità finale. Esse prendono consistenza e diventano utili solo nel "contesto di utilizzo" del prodotto. Altrimenti, prese in sé, esse sono solo un esercizio astratto senza alcuna attinenza con la realtà. Il contesto di utilizzo di un prodotto software definisce "chi" sono le persone che dovranno utilizzarlo, "perché" hanno necessità di utilizzarlo, ed in "che modo" è loro più congeniale e semplice utilizzarlo. I tre elementi che identificano lo scenario d'utilizzo di un prodotto software sono: gli "utenti" del prodotto, gli "obiettivi" che si intende raggiungere con il suo utilizzo, le "modalità operative" generalmente utilizzate (preferite!) dagli utenti per svolgere i propri compiti.

1.1 Alcune domande sulla qualità

Le domande che bisognerebbe assolutamente porsi per pianificare prima, sviluppare dopo e valutare al termine del progetto, il livello di qualità del software sono quelle che, nella realtà quotidiana, si pongono gli utenti quando sono richiesti di utilizzare un nuovo prodotto.

- Il prodotto dispone di tutte le funzioni di cui gli utenti hanno bisogno per completare le attività richieste dal business?
- Il prodotto permette agli utenti di utilizzare le funzioni in maniera semplice ed efficace, in linea con le modalità operative abitualmente adoperate, utilizzando un linguaggio semplice ed adeguato al contesto, fornendo aiuto e supporto in caso di necessità?
- Il prodotto fornisce prestazioni accettabili per lo specifico contesto d'uso? Utilizza al meglio le risorse ed i materiali?
- Il prodotto è affidabile tanto da garantire il pieno funzionamento, senza interruzioni, per tutto il tempo necessario a completare le attività operative richieste dal business? Ed in caso d'interruzione o malfunzionamento, ripristina l'operatività senza perdita di dati e senza richiedere di ripetere le operazioni svolte prima del fermo?
- Il prodotto garantisce la sicurezza dei dati, delle informazioni, delle persone e dei beni? Impedisce gli accessi non autorizzati alle funzioni ed ai dati?
- Il prodotto può essere facilmente installato, personalizzato ed utilizzato nei diversi ambienti operativi previsti?

- Il prodotto è facilmente aggiornabile con rilasci di nuove funzioni e correzioni degli errori?

Per realizzare la qualità di un prodotto software occorre porsi tali domande all'inizio del progetto e fornire risposte chiare, complete e convincenti. Bisogna poi definire le caratteristiche di qualità che meglio rappresentano le esigenze, definire metriche appropriate, stabilire intervalli e soglie d'accettazione. Quindi si pianificano le azioni più opportune per progettare la qualità, controllarne lo sviluppo e valutare i risultati. La progettazione del software deve indirizzare gli elementi di qualità con opportune soluzioni tecniche ed applicative.

La qualità di un prodotto software riguarda caratteristiche concernenti differenti punti di vista. Si parla quindi di caratteristiche di qualità "interne", "esterne" e "d'uso". Le caratteristiche interne riguardano la qualità intrinseca del software come, ad esempio, la sua modularità, complessità, comprensibilità, difettosità, aderenza a standard ecc. Alcune di queste caratteristiche hanno solo rilevanza interna, determinano il livello di qualità del codice e sono importanti per gli sviluppatori ed i responsabili della manutenzione. Alcune invece influenzano anche aspetti esterni del prodotto come, ad esempio, l'usabilità, l'affidabilità, la sicurezza, ecc. Esistono altre caratteristiche che rappresentano esclusivamente punti di vista esterni al prodotto, in quanto hanno rilevanza solo per l'utente. Ad esempio, la facilità d'uso dovuta ad un'interfaccia ben disegnata, piacevole e consona all'ambiente culturale in cui il prodotto è utilizzato non ha alcuna rilevanza per la qualità del codice sorgente, mentre ne ha molta per gli utenti finali. Ci sono caratteristiche, infine, che hanno rilevanza solo nel momento in cui il prodotto è utilizzato in un determinato contesto. La soddisfazione che un utente ricava dall'utilizzo del prodotto nel proprio ambiente di lavoro, per esempio, è una caratteristica d'uso rilevabile solo dall'utilizzo del prodotto. Una caratteristica d'uso, come ad esempio la soddisfazione dell'utente, è l'effetto procurato da altre caratteristiche di qualità del prodotto come quelle esterne del tipo usabilità, affidabilità, prestazioni, ecc.

In conclusione, le caratteristiche di qualità interne possono influire sulle caratteristiche esterne e queste su quelle d'uso. Altre, invece, sono solo caratteristiche interne o esterne. Il modello che meglio rappresenta le diverse caratteristiche di qualità del software, è quello descritto dalle norme ISO/IEC 9126 trattato nel prossimo capitolo.

1.2 Come affrontare la qualità in un progetto software

L'approccio più corretto per indirizzare la qualità di un prodotto software da sviluppare è quello di dare una risposta chiara e convincente alle domande riportate nel paragrafo precedente. Di seguito è riportato un esempio di come interpretare tali domande e quali

risposte fornire. Costruire la qualità di un prodotto software ha un suo costo spesso non trascurabile. Il livello di qualità richiesto deve essere garantito allo stesso modo con cui garantiamo le funzioni richieste. La qualità, quindi, deve essere pianificata, progettata, sviluppata, verificata, misurata e valutata lungo l'intero ciclo di vita del progetto. E' importante evidenziare che la qualità, quando indirizzata fin dall'inizio del progetto, è "meno costosa" di quanto non risulti nella maggior parte dei progetti dove, inizialmente trascurata, è poi indirizzata con modifiche non strutturate, poco efficaci e "più costose". Fare bene sin dall'inizio paga sempre!

1. **Domanda**: Il prodotto dispone di tutte le funzioni di cui gli utenti hanno bisogno per completare le attività richieste dal business?

   Risposta: L'approccio giusto è quello di condurre un'analisi approfondita delle necessità del cliente, raccogliere i requisiti espliciti ed impliciti, funzionali e di qualità, identificare i limiti e le regolamentazioni cui aderire. I requisiti raccolti sono quindi interpretati, documentati, valutati e rivisti con il cliente per assicurarci di averli correttamente capiti. Essi diventano il punto di partenza per ogni altra attività del progetto. La metrica è molto semplice: "verificare che il prodotto sviluppi tutte le funzioni necessarie con la cura e la precisione necessaria". La mancanza di una funzione richiesta è una "mancanza di qualità"; sviluppare funzioni non richieste non compensa da eventuali funzioni necessarie e mancanti. Il piano di progetto deve perciò prevedere lo sviluppo di tutte le funzioni richieste. Eventuali problemi nei tempi di consegna sono risolti concordando con il cliente nuove priorità, definendo le funzioni da includere nel primo rilascio e quelle da spostate nei rilasci successivi. Eventuali problemi nei costi di sviluppo non saranno mai risolti a discapito della qualità: la negoziazione con il cliente rappresenta la strada maestra. Ogni approccio diverso crea problemi ed il fallimento di molti progetti ne è una dimostrazione.

2. **Domanda**: Il prodotto permette agli utenti di utilizzare le funzioni in maniera semplice ed efficace, in linea con le abituali modalità operative, utilizzando un linguaggio semplice ed adeguato al contesto, fornendo aiuto e supporto in caso di necessità?

   Risposta: L'usabilità incide moltissimo sulla qualità del prodotto! Disporre di funzioni ma non poterle utilizzare con facilità genera frustrazione. Non poter utilizzare una funzione con efficacia riduce la produttività. Non ritrovarsi nei modi abituali di operare, non riconoscersi nel linguaggio adoperato irrita l'utente. L'usabilità è importante quanto la funzionalità stessa. Trascurarla significa abbassare la qualità del prodotto, rendere insoddisfatto l'utente, dimostrare poca professionalità nello sviluppo del software! E' bene definire l'usabilità tenendo conto del profilo degli utenti, dei loro obiettivi e delle loro modalità operative. Su tali basi si definiscono le caratteristiche

dell'usabilità, le metriche da adottare ed i valori da raggiungere. L'usabilità non si ottiene casualmente; essa è progettata, rivista, verificata e valutata tramite prototipi, attività euristiche[1] e test di usabilità condotte da esperti. Il piano di progetto deve prevedere perciò il coinvolgimento di esperti nelle attività necessarie alla progettazione ed al collaudo dell'usabilità (*usability design, usability review, usability assessment, usability test*).

3. **Domanda**: Il prodotto fornisce prestazioni accettabili per lo specifico contesto d'uso? Utilizza al meglio le risorse ed i materiali?

   Risposta: Altro elemento di qualità del software. Dover attendere tempi lunghissimi per ottenere i risultati del lavoro svolto crea frustrazione ed irritazione, abbassa la produttività, rende insoddisfatti gli utenti! Utilizzare impropriamente le risorse ed i materiali è antieconomico e poco produttivo. Le prestazioni sono una necessità quanto le funzioni stesse ed acquistano valore nel contesto nel quale si applicano. Sono determinanti quando servono, poco rilevanti quando non necessarie. Il tempo di risposta di una transazione eseguita da uno sportellista che ha di fronte clienti in fila deve essere necessariamente breve. Un tabulato da produrre durante la notte può essere stampato in tempi ragionevoli. Le prestazioni devono essere definite in base alle necessità reali. Queste sono definite e concordate con gli utenti, valutate in termini di costi di sviluppo, pianificate, progettate fin dall'inizio e misurate e valutate tramite test opportuni (*performance test*). I requisiti relativi alle prestazioni sono raccolti, documentati e concordati prima di iniziare la progettazione del software. Il piano di progetto prevede le attività necessarie alla progettazione ed al collaudo delle prestazioni.

4. **Domanda**: Il prodotto è affidabile tanto da garantire il pieno funzionamento, senza interruzioni, per tutto il tempo necessario a completare le attività operative richieste dal business? Ed in caso d'interruzione o malfunzionamento, ripristina l'operatività senza perdita di dati e senza richiedere di ripetere le operazioni svolte prima del fermo?

   Risposta: Come non ritenere fondamentale tale caratteristica? E' frustrante, demotivante, irritante per utente "non poter completare il proprio lavoro a causa di uno stupido programma software fatto male e che si ferma!" (espressione dell'utente riportata qui in maniera edulcorata). L'affidabilità è definita in base alle caratteristiche del business ed alla criticità che la particolare funzione riveste. Non tutte le funzioni richiedono quindi altissima affidabilità (24/24 ore, 7/7 giorni); quando richiesta deve però essere

[1] Le attività euristiche sono condotte da esperti che, in base all'esperienza e competenza, sono in grado di "prevedere" i risultati finali analizzando il lavoro svolto alla data. Per esempio, gli esperti di usabilità possono valutare il livello di facilità d'uso di un prodotto tramite la revisione del progetto delle interfacce (specifiche, prototipo, ecc.)

garantita. Altrettanta importanza riveste la capacità del prodotto di ripristinare le condizioni iniziali ed evitare la perdita di dati. Il livello di affidabilità e la capacità di ripristino sono definite per le diverse funzioni. Le metriche ed i valori soglia sono concordati con i responsabili del business. La progettazione del software tiene conto di tali caratteristiche e verifica – per esempio, con prototipi - se il design garantisce l'affidabilità richiesta. Il software sviluppato è quindi collaudato tramite un test di affidabilità (*availability test, stress test*) e l'utilizzo di opportuni strumenti per simulare il carico di lavoro e ripetere i test automaticamente. I requisiti di affidabilità sono raccolti, documentati e concordati prima di iniziare la progettazione del software. Il piano di sviluppo prevede le attività necessarie alla progettazione ed al collaudo dell'affidabilità.

5. **Domanda**: Il prodotto garantisce la sicurezza dei dati, delle informazioni, delle persone e dei beni? Impedisce gli accessi non autorizzati alle funzioni ed ai dati?

   Risposta: Nessuno di noi lascia aperta la porta uscendo di casa. Perchè dunque dovremmo "lasciare aperta la porta del nostro business?" La sicurezza è una cosa seria; è definita e concordata con il cliente, progettata da esperti della materia, sviluppata con opportune tecniche e collaudata con test specifici (*intrusion test*) condotti da altrettanti esperti (*ethical hackers*). I requisiti di sicurezza sono quindi raccolti, documentati e concordati prima di iniziare la progettazione del software. La progettazione indirizza questi requisiti ed i test ne verificano la correttezza dell'implementazione. Il piano di progetto prevede quindi il coinvolgimento di esperti nelle attività necessarie alla progettazione ed al collaudo della sicurezza.

6. **Domanda**: Il prodotto può essere facilmente installato, personalizzato ed utilizzato nei diversi ambienti operativi previsti?

   Risposta: Tali caratteristiche del prodotto dipendono strettamente dalle specifiche degli ambienti nei quali il prodotto sarà installato, dalla semplificazione delle azioni da eseguire e dalla complessità del prodotto stesso. Generalmente risultano chiare le caratteristiche tecniche dell'ambiente principale in cui sarà installato il prodotto, mentre poche informazioni sono invece fornite su eventuali altre piattaforme dove il prodotto potrebbe essere, o sarà, installato. In questo caso occorre fare molta attenzione ed individuare le caratteristiche tecniche di tali ambienti e tenerne ben conto in fase di progettazione. Il collaudo, in questo caso, deve poter verificare il corretto funzionamento del prodotto in tutti gli ambienti previsti. Il piano di progetto prevede in questo caso le attività – spesso costose – di predisposizione ed utilizzo degli altri ambienti di collaudo.

7. **Domanda**: Il prodotto è facilmente aggiornabile con rilasci di nuove funzioni e correzione degli errori?

Risposta: Questa caratteristica del prodotto incide sulla produttività del gruppo di manutenzione. Un software non progettato per essere facilmente modificato inciderà sul costo delle modifiche. La manutenzione, infatti, è fornita in garanzia nei primi 12 mesi dopo il rilascio (e quindi gratis) e con un contratto "a corpo" per tutto il tempo successivo. La difettosità residua (il numero di errori rimasti nel software rilasciato), la numerosità e complessità delle modifiche evolutive, la complessità del software, e quindi la difficoltà di modificarlo, sono tutti elementi importanti ai fini di un corretto dimensionamento dei costi di manutenzione. E' bene quindi considerare, in fase di pianificazione del progetto, questi elementi con cura e precisione. La metrica relativa alla "facilità di manutenzione" è quindi importante ai fini della stima dei costi di manutenzione. Una progettazione razionale e la scrittura di un buon codice sono gli ingredienti di una buona ricetta.

Nella pratica quotidiana si raccomanda di:

- Realizzare una buona strutturazione del software,
- Garantire l'information hiding,
- Ridurre la complessità del codice,
- Aumentare la coesione ed il disaccoppiamento,
- Ridurre una buona autodocumentazione del codice,
- Utilizzare standard di programmazione,
- Produrre una buona documentazione tecnica,
- Registrare i dati necessari per poter fare il debugging degli errori (produrre log, trace), ecc.

2 Il modello ISO/IEC 9126

Negli anni sono stati definiti molti modelli della qualità del software. Ciascuno di essi definisce un set di caratteristiche ed attributi propri, anche se molto simili tra di loro. Qui si descrive il modello proposto dall'ISO, standard internazionale.
Le norme ISO/IEC 9126 descrivono quindi un modello di qualità del software, definiscono le caratteristiche che la determinano e propongono metriche per la misurazione. Le norme sono emesse dall'ISO, l'organismo internazionale di standardizzazione (*International Organization for Standardization*), cui aderiscono moltissimi paesi al mondo e collaborano diversi enti nazionali, anche non governativi; tra questi, per esempio, IEC, l'organo internazionale che definisce gli standard nel settore delle tecnologie dell'informazione e comunicazione (*International Electrotechnical Commission*).

Le norme relative alla qualità del software sono emesse come ISO/IEC 9126 e constano di quattro parti:

Parte 1: Modello della qualità del software,
Parte 2: Metriche per la qualità esterna,
Parte 3: Metriche per la qualità interna,
Parte 4: Metriche per la qualità in uso.

Il modello, si è detto, definisce le caratteristiche di qualità. A ciascuna di esse sono associate sottocaratteristiche, dette anche attributi. La tabella 1 riassume le caratteristiche e gli attributi del software proposti dal modello. Maggiori dettagli si possono trovare direttamente nelle norme ISO/IEC 9126 i cui riferimenti sono riportati nella bibliografia presente alla fine del lavoro.

Il modello propone anche un approccio alla qualità affinché le società di software possano migliorare l'organizzazione ed i processi e, come ricaduta concreta ed immediata, la qualità del prodotto sviluppato. E' inoltre rappresentata la catena del valore che, partendo dalla maturità del processo, realizza la qualità interna ed esterna del prodotto e, da queste, quella in uso. E' anche mostrata la relazione che lega la qualità del prodotto al ciclo di vita del software, e viceversa.

Tabella 1. Modello di qualità del software (ISO/IEC 9126).

Caratteristica	Attributi
Funzionalità (*Functionality*)	Completezza (*Suitability*) Accuratezza (*Accuracy*) Interoperabilità (*Interoperability*) Sicurezza (*Security*) Aderenza alla funzionalità (*Functionality compliance*)
Affidabilità (*Reliability*)	Maturità (*Maturity*) Tolleranza ai guasti (*Fault tolerance)* Recuperabilità (*Recoverability*) Aderenza all'affidabilità (*Reliability compliance*)
Usabilità (*Usability*)	Comprensibilità (*Understandability*) Apprendibilità (*Learnability*) Operabilità (*Operability*) Attrattività (*Attractiveness*) Aderenza all'usabilità (*Aderenza all'usabilità*)
Efficienza (*Efficiency*)	Comportamento rispetto al tempo (*Time behaviour*) Utilizzo delle risorse (*Resource utilization*) Aderenza all'efficienza (*Efficiency compliance*)
Manutenibilità (*Maintainability*)	Analizzabilità (*Analysability*) Modificabilità (*Changeability*) Stabilità (*Stability*) Provabilità (*Testability*) Aderenza alla manutenibilità (*Maintainability compliance*)
Portabilità (*Portability*)	Adattabilità (*Adaptability*) Installabilità (*Installability*) Coesistenza (*Co-existence*) Sostituibilità (*Replaceability*) Aderenza alla portabilità (*Portability compliance*)
Qualità in uso (*Quality in use*)	Efficacia (*Effectiveness*) Produttività (*Productivity*) Sicurezza (*Safety*) Soddisfazione (*Satisfaction*)

2.1 L'approccio alla qualità

La qualità di un prodotto software dipende dalla maturità del processo adoperato per il suo sviluppo. Dalla qualità del processo deriva anche l'efficacia del progetto di sviluppo. E' ben nota la tesi secondo cui la qualità di un prodotto software dipende dalla qualità del processo di sviluppo adoperato. Ma grande importanza riveste anche, e soprattutto, la maturità dell'organizzazione. In termini più generali:

> *La qualità di un prodotto software dipende dalla maturità dell'organizzazione che lo produce, intesa come competenza delle persone, efficacia dei processi ed utilizzo di metodi, tecniche e strumenti adeguati.*

La Figura 1 mostra l'influenza del processo di sviluppo sulla qualità del prodotto software realizzato.

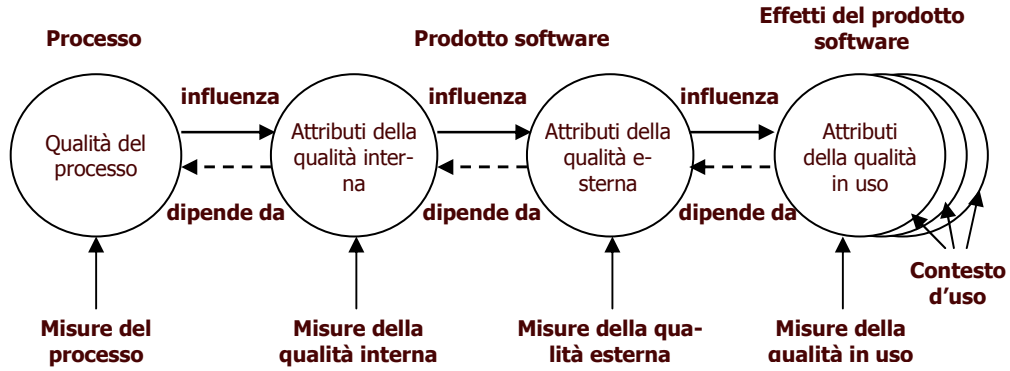

Figura 1. Qualità del prodotto software nel ciclo di sviluppo.

Come in una catena del valore, la qualità del processo di sviluppo influenza la qualità interna del prodotto realizzato. Questa, a sua volta, influenza la qualità esterna del prodotto che, per finire, influenza la qualità del prodotto in uso, cioè quella percepita dagli utenti finali. Occorre perciò definire gli attributi che caratterizzano ciascuna tipologia qualitativa e le relative metriche per misurarne il livello raggiunto.

La definizione dei "requisiti non funzionali" – detti anche "requisiti di qualità" – è bene sia fatta tenendo conto del "contesto d'utilizzo" del prodotto. I requisiti di qualità sono tradotti in una serie di caratteristiche di qualità – interne, esterne ed in uso – indirizzate poi dai progettisti nel disegno del prodotto, dagli sviluppatori nel codice, dai

tester nella verifica e validazione, dall'assicurazione qualità nella valutazione del prodotto finale.

La maturità del processo contribuisce quindi a migliorare la qualità del prodotto sviluppato; questa, a sua volta, contribuisce a migliorare la qualità in uso. Valutare e migliorare la maturità del processo è quindi un mezzo per migliorare la qualità del prodotto. Analogamente, valutare e migliorare la qualità del prodotto è uno dei mezzi per migliorare la qualità in uso. Seguendo il percorso inverso, valutare la qualità in uso fornisce un valido riscontro (feedback) per migliorare il prodotto; valutare la qualità del software – interna ed esterna – fornisce un riscontro per migliorare il processo.

Gli attributi interni del software, definiti in modo appropriato, sono un prerequisito per raggiungere il livello di qualità esterna del prodotto; le caratteristiche esterne del software, anch'esse definite in maniera opportuna, rappresentano un prerequisito per raggiungere la qualità in uso desiderata.

I requisiti di qualità del prodotto devono includere anche i criteri per valutare la qualità interna, esterna ed in uso; i criteri di valutazione, a loro volta, devono essere in linea con le necessità di quanti utilizzeranno il prodotto: gli utenti finali, i responsabili dell'installazione, della gestione e della manutenzione.

2.2 La qualità del prodotto ed il ciclo di vita del software

Il punto di vista della qualità interna, esterna ed in uso cambia durante il ciclo di vita del software. All'inizio del ciclo di sviluppo, per esempio, la qualità specificata dai requisiti descrive principalmente la qualità "esterna" del prodotto secondo il punto di vista degli utenti. Durante la fase di progettazione, invece, la "qualità del design" rappresenta l'aspetto interno del prodotto ed il punto di vista degli sviluppatori. Le tecnologie utilizzate per raggiungere il livello di qualità necessario devono perciò supportare questi diversi punti di vista. E' quindi necessario definire le diverse prospettive e le relative tecnologie associate, allo scopo di garantire la gestione più appropriata della qualità nelle diverse fasi del ciclo di sviluppo.

L'obiettivo di ogni progetto di sviluppo è quello di raggiungere il livello di qualità necessario e sufficiente cos'ì come espresso dalle necessità dell'utente. L'ISO 8402 definisce la qualità come "abilità a soddisfare le necessità degli utenti, esplicite ed implicite". Tuttavia, le necessità esplicitate dagli utenti non sempre corrispondono ai "veri" bisogni, per vari motivi: (1) un utente non sempre è consapevole di cosa realmente egli necessiti, (2) le necessità possono cambiare dopo che siano state esplicitate, (3) utenti diversi potrebbero avere differenti necessità legate a diversi contesti di utilizzo (organizzativo, applicativo, tecnologico), (4) potrebbe risultare impossibile contattare tutti gli

utenti, specialmente nel caso di prodotti software realizzati per il mercato. Non tutti i requisiti di qualità, quindi, potrebbero essere definiti prima dell'inizio della progettazione. Tuttavia, è necessario capire le reali necessità degli utenti con il massimo dettaglio e completezza possibile al momento, e tradurle nei requisiti. L'obiettivo non è tanto quello di raggiungere una qualità perfetta, quanto quello di raggiungere il livello di qualità necessario e sufficiente in ciascun contesto di utilizzo del prodotto finale.

La scala utilizzata per valutare quanto i requisiti di qualità siano stati soddisfatti, può essere divisa in categorie diverse, a seconda del livello di soddifazione. Per esempio, una scala semplice può essere divisa in due categorie: "soddisfatto" e "non soddisfatto". Una scala più completa può prevedere quattro categorie: "Requisiti superati", "pienamente raggiunti", "appena accettabili" e "non accettabili" (ISO/IEC 14598-1). La definizione delle categorie deve essere concordata da entrambe le parti, gli utenti e gli sviluppatori.

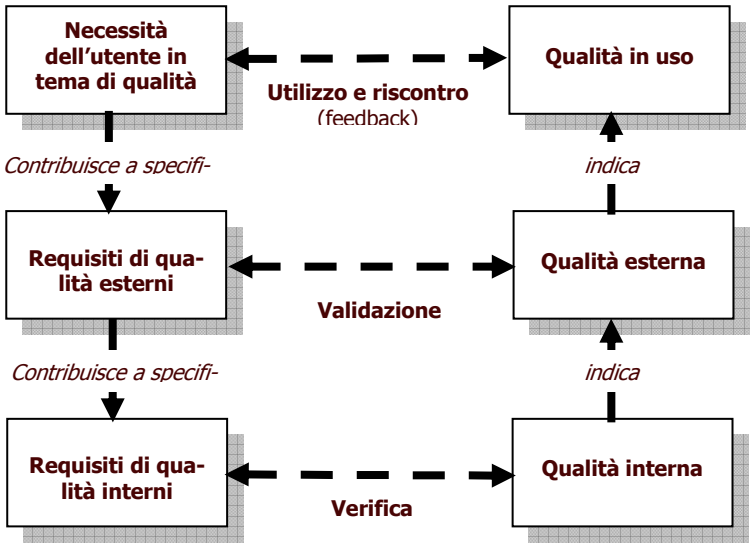

Figura 2. La qualità nel ciclo di vita del software.

Riassumendo, ci sono diversi punti di vista della qualità del prodotto e delle metriche associate nei diversi momenti del ciclo di vita del software, come mostrato nella Figura 2[2].

La parte sinistra della figura mostra, partendo dall'alto, le esigenze qualitative legate all'utilizzo del prodotto ed espresse direttamente dagli utenti. Esse contribuiscono a de-

[2] La figura è una semplificazione della versione riportata nella norma ISO/IEC 14598-1: 1998, Figura 4, qui modificata per renderla consistente con la norma ISO/IEC 9126-1.

finire le caratteristiche di qualità esterne del prodotto. Queste ultime, a loro volta, contribuiscono a definire le caratteristiche di qualità interna del prodotto.

Procedendo invece verso l'alto nella parte destra della figura, si evidenzia come la qualità interna del software sviluppato è confrontata con i relativi requisiti tramite un'attività di verifica della corrispondenza alle specifiche tecniche. I risultati della verifica forniscono indicazioni sulla possibile qualità esterna del prodotto. Questa, a sua volta, fornisce indicazioni sulla qualità in uso. In particolare, la qualità esterna è validata tramite attività di test e confronto con i relativi requisiti esterni. La qualità in uso, invece, è valutata tramite l'utilizzo del prodotto da parte degli utenti.

Il risultato del confronto, a tutti i livelli, tra requisiti e qualità raggiunta ha valenza bidirezionale: da un lato si valuta quanto siano stati raggiunti gli obiettivi, dall'altro si fornisce un riscontro (feedback) rispettivamente agli sviluppatori, ai progettisti, agli utenti.

2.3 Il modello della qualità interna ed esterna del software

Il modello di qualità del software descritto dalle norme ISO/IEC 9126 definisce le caratteristiche e sottocaratteristiche del software, ciascuna misurabile da metriche interne o esterne.

Il modello definisce sei caratteristiche (funzionalità, affidabilità, usabilità, efficienza, manutenibilità e portabilità), suddivise a loro volta in sottocaratteristiche, come mostrato nella Figura 3.

> ***Nota**: Per ciascuna caratteristica del software è definita una sottocaratteristica "conformità" o "aderenza" (compliance) del software a standard, normative e regolamentazioni definiti per la caratteristica in oggetto.*

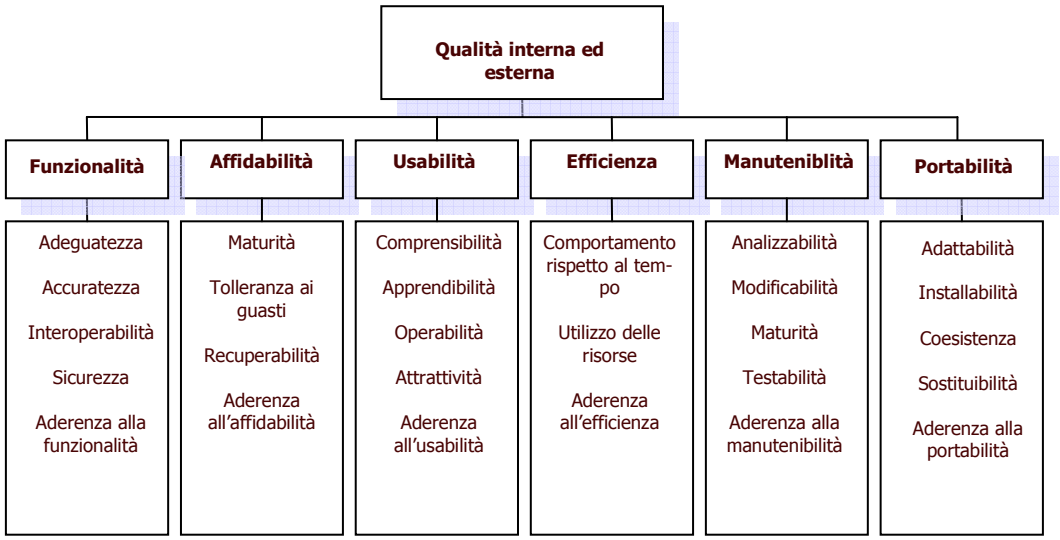

Figura 3. Modello della qualità esterna ed interna (tratto dalle norme ISO/IEC 9126-1).

2.3.1 Funzionalità (*Functionality*)

La *funzionalità* rappresenta la capacità del software di fornire le funzioni, espresse ed implicite, necessarie per operare in determinate condizioni, cioè in un determinato contesto. Gli attributi del software richiesti da questa caratteristica sono: adeguatezza, accuratezza, interoperabilità, sicurezza, aderenza.

Adeguatezza (*Suitability*)

L'*adeguatezza* di un prodotto software rappresenta la capacità di fornire un appropriato insieme di funzioni che permettano agli utenti di svolgere determinati task e di raggiungere gli obiettivi prefissati.

Accuratezza (*Accuracy*)

L'*accuratezza* di un prodotto software rappresenta la capacità di fornire i risultati o gli effetti attesi con il livello di precisione richiesta.

Interoperabilità (*Interoperability*)

L'*interoperabilità* di un prodotto software rappresenta la capacità di interagire con uno o più sistemi specificati.

Sicurezza (*Security*)

La *sicurezza* di un prodotto software rappresenta la capacità di proteggere le informazioni ed i dati in modo che, persone o sistemi non autorizzati, non possano accedervi e

quindi non possano leggerli o modificarli. Quanto detto si applica anche alla trasmissione dei dati. La sicurezza intesa come protezione delle persone fisiche (*Safety*) è indirizzata dalle caratteristiche della "qualità in uso".

Aderenza (*Functionality compliance*)

L'*aderenza alla funzionalità* di un prodotto software rappresenta la capacità di aderire a standard, convenzioni e regolamenti di carattere legale o prescrizioni simili che abbiano attinenza con la funzionalità.

2.3.2 Affidabilità (*Reliability*)

L'*affidabilità* rappresenta la capacità di un prodotto software di mantenere il livello di prestazione quando viene utilizzato in condizioni specificate. Possibili limitazioni all'affidabilità del software possono essere causate da errori nei requisiti, nella progettazione, nel codice. Le evidenze di tali errori possono essere rilevate a seconda delle condizioni in cui il prodotto è utilizzato oppure alle opzioni scelte, piuttosto che al momento in cui è utilizzato. Gli attributi richiesti da tale caratteristiche sono: maturità, tolleranza ai guasti, recuperabilità, aderenza.

Maturità (*Maturity*)

La *maturità* di un prodotto software rappresenta la capacità di evitare che si verifichino errori o siano prodotti risultati non corretti in fase di esecuzione.

Tolleranza ai guasti (*Fault tolerance*)

La *tolleranza ai guasti* di un prodotto software rappresenta la capacità di mantenere il livello di prestazioni in caso di errori nel software o di violazione delle interfacce specificate.

Recuperabilità (*Recoverability*)

La *recuperabilità* di un prodotto software rappresenta la capacità di ripristinare il livello di prestazioni e di recuperare i dati direttamente coinvolti in caso di errori o malfunzionamenti. A seguito di un errore o malfunzionamento, il software può risultare non accessibile per un determinato periodo di tempo; tale intervallo di tempo è valutato dalla caratteristica di recuperabilità.

La *disponibilità* rappresenta invece la capacità del prodotto software di rimanere operativo – cioè di poter eseguire le funzioni richieste – per un periodo di tempo specificato (per esempio, per tutto il tempo necessario a completare un determinato task), in determinate condizioni di utilizzo. Esternamente, la disponibilità può essere valutata per mezzo della percentuale di tempo in cui il prodotto rimane operativo. Questa caratteristica è quindi la combinazione di altre due: "maturità" (che governa la frequenza con cui si verificano i malfunzionamenti) e "tolleranza ai guasti" (che governa il tempo in cui il prodotto non è operativo). Per questo motivo la disponibilità non è stata definita come una caratteristica separata.

Aderenza (*Reliability compliance*)

L'*aderenza all'affidabilità* di un prodotto software rappresenta la capacità di aderire a standard, convenzioni e regole relative all'affidabilità.

2.3.3 Usabilità (*Usability*)

L'*usabilità* rappresenta la capacità di un prodotto software di essere comprensibile, di poter essere studiato, di risultare attraente da parte di un utente sotto determinate condizioni. Alcuni aspetti della "funzionalità", "affidabilità" ed "efficacia" possono influire sull'usabilità del prodotto software anche se, per il modello ISO/IEC 9126, queste non sono classificate come usabilità. Tra gli utenti sono inclusi gli operatori, gli utenti finali ed altri utenti indiretti che sono influenzati dal software o che dipendono da esso. L'usabilità dovrà indirizzare tutti gli ambienti e scenari di utilizzo del prodotto, inclusa la preparazione all'utilizzo del software e la valutazione dei risultati. Gli attributi della caratteristica usabilità sono: comprensibilità, apprendibilità, operabilità, attrattività e aderenza all'usabilità.

Comprensibilità (*Understandability*)

La *comprensibilità* di un prodotto software rappresenta la capacità di permettere all'utente di capire la sua funzionalità e come poterla utilizzare con successo per svolgere particolari task in determinate condizioni di utilizzo. Essa dipende dalla documentazione disponibile e dall'impressione iniziale che si riceve dal prodotto.

Apprendibilità (*Learnability*)

L'*apprendibilità* di un prodotto software rappresenta la capacità di permettere all'utente di imparare l'applicazione. L'attributo corrisponde alla funzionalità relativa all'apprendimento del software.

Operabilità (*Operability*)

L'*operabilità* di un prodotto software rappresenta la capacità di permetter all'utente di utilizzarlo e di controllarlo. Gli aspetti relativi alla funzionalità, modificabilità, adattabilità ed installabilità del software possono influire sull'operatività del prodotto. L'operabilità del software fa riferimento anche alle aspettative dell'unte sulla sua controllabilità, tolleranza ai guasti e conformità. Relativamente ad un sistema in uso, la combinazione delle caratteristiche di funzionalità, affidabilità, usabilità ed efficienza possono essere misurate esternamente come "qualità in uso".

Attrattività (*Attractiveness*)

L'*attrattività* di un prodotto software rappresenta la capacità di risultare "attraente" per l'utente. La qualità è relativa alla progettazione dell'aspetto grafico delle sue interfacce, all'utilizzo dei colori e immagini, ecc.

Aderenza all'usabilità (*Usability compliance*)

L'*aderenza all'usabilità* di un prodotto software rappresenta la capacità di aderire a standard, convenzioni, guida allo stile e regole relative all'usabilità.

2.3.4 Efficienza (*Efficiency*)

L'*efficienza* rappresenta la capacità di un prodotto software di realizzare le funzioni richieste nel minor tempo possibile ed utilizzando nel miglior modo le risorse necessarie, quando opera in determinate condizioni. Le risorse includono altri prodotti software, la configurazione hardware e software del sistema, i materiali (carta per stampanti, dischetti, ecc.). Gli attributi relativi all'efficienza del software sono: comportamento rispetto al tempo, utilizzo delle risorse, aderenza all'efficienza.

Comportamento rispetto al tempo (*Time behaviour*)

Il *comportamento rispetto al tempo* di un prodotto software rappresenta la capacità di fornire appropriati tempi di risposta, tempi di elaborazione e quantità di lavoro eseguendo le funzionalità previste sotto determinate condizioni di utilizzo.

Utilizzo delle risorse (*Resource utilization*)

L'*utilizzo delle risorse* da parte di un prodotto software rappresenta la capacità di utilizzare un appropriato numero e tipo di risorse quando esegue le funzionalità previste sotto determinate condizioni di utilizzo. Le persone (risorse umane) sono incluse nella definizione di risorse.

Aderenza all'efficienza (*Efficiency compliance*)

L'*aderenza all'efficienza* di un prodotto software rappresenta la capacità di aderire a standard e convenzioni relative all'efficienza.

2.3.5 Manutenibilità (*Maintainability*)

La *manutenibilità* rappresenta la capacità di un prodotto software di essere modificato. Le modifiche possono includere correzioni o adattamenti del software a modifiche negli ambienti, nei requisiti e nelle specifiche funzionali. In altre parole, "*Il software segue l'evoluzione dell'organizzazione*". Gli attributi previsti per la manutenibilità del software sono: analizzabilità, modificabilità, stabilità, provabilità, aderenza alla manutenibilità.

Analizzabilità (*Analysability*)

L'*analizzabilità* di un prodotto software rappresenta la capacità di poter effettuare la diagnosi sul software ed individuare le cause di errori o malfunzionamenti.

Modificabilità (*Changeability*)

La *modificabilità* di un prodotto software rappresenta la capacità di consentire lo sviluppo di modifiche al software originale. L'implementazione include modifiche al codice, alla progettazione ed alla documentazione.

Nel caso in cui le modifiche debbano essere fatte dagli utenti, la modificabilità può influire sull'operatività del prodotto.

Stabilità (*Stability*)

La *stabilità* di un prodotto software rappresenta la capacità di evitare effetti non desiderati a seguito di modifiche al software.

Provabilità (*Testability*)

La *provabilità* di un prodotto software rappresenta la capacità di consentire la verifica e validazione del software modificato, cioè di eseguire i test.

Aderenza alla manutenibilità (*Maintainability compliance*)

L'*aderenza all'efficienza* di un prodotto software rappresenta la capacità di aderire a standard e convenzioni relative alla manutenibilità.

2.3.6 Portabilità (*Portability*)

La *portabilità* rappresenta la capacità di un prodotto software di poter essere trasportato da un ambiente ad un altro. L'ambiente include aspetti organizzativi e tecnologici (hardware e software). In altre parole, "*Il software segue l'evoluzione tecnologica*". Gli attributi previsti dalla portabilità sono: adattabilità, installabilità, coesistenza, sostituibilità, aderenza alla portabilità.

Adattabilità (*Adaptability*)

L'*adattabilità* di un prodotto software rappresenta la capacità di essere adattato a differenti ambienti senza richiedere azioni specifiche diverse da quelle previste dal software per tali attività. L'adattabilità include la scalabilità delle capacità interne del prodotto (campi delle schermate, tabelle, volumi delle transazioni, formato dei report, ecc.). Se il software deve essere adattato dall'utente, l'adattabilità corrisponde ad una funzionalità specifica e può influire sull'operatività del prodotto.

Installabilità (*Installability*)

L'*installabilità* di un prodotto software rappresenta la capacità di essere installato in un determinato ambiente. Se il software deve essere installato dall'utente, la caratteristica in oggetto può influire sulla funzionalità e l'operatività.

Coesistenza (*Co-existence*)

La *coesistenza* di un prodotto software rappresenta la capacità di coesistere con altre applicazioni indipendenti in ambienti comuni e di condividere le risorse.

Sostituibilità (*Replaceability*)

La *sostituibilità* di un prodotto software rappresenta la capacità di sostituire un altro software specifico indipendente, per lo stesso scopo e nello stesso ambiente. Per esempio, la sostituibilità di nuova versione di un prodotto software è molto importante per l'utente che dovrà eseguire l'aggiornamento. In questo caso, il termine sostituisce la parola "compatibilità" per evitare possibili ambiguità con la caratteristica interoperabilità. La sostituibilità include quindi attributi come "installabilità" e "adattabilità".

Aderenza alla portabilità (*Portability compliance*)

L'*aderenza alla portabilità* di un prodotto software rappresenta la capacità di aderire a standard e convenzioni relative alla portabilità.

2.4 Il modello della qualità in uso del software

Qui di seguito è descritto il modello relativo alla qualità del software in uso. Gli attributi della qualità in uso sono rappresentati da quattro categorie: efficienza, produttività, sicurezza fisica e soddisfazione, come mostrato nella Figura 4. La qualità in uso rappresenta il punto di vista dell'utente sulla qualità. Il livello di qualità in uso è raggiunto quando sia stato raggiunto anche il livello di qualità esterna e, ancora prima, quella interna. Occorre quindi effettuare, nell'ordine, misure sulla qualità interna, esterna ed in uso. Le norme ISO/IEC 9126-4 forniscono esempi di metriche da utilizzare per la misurazione della qualità rispetto ai diversi punti di vista (interno, esterno, in uso).

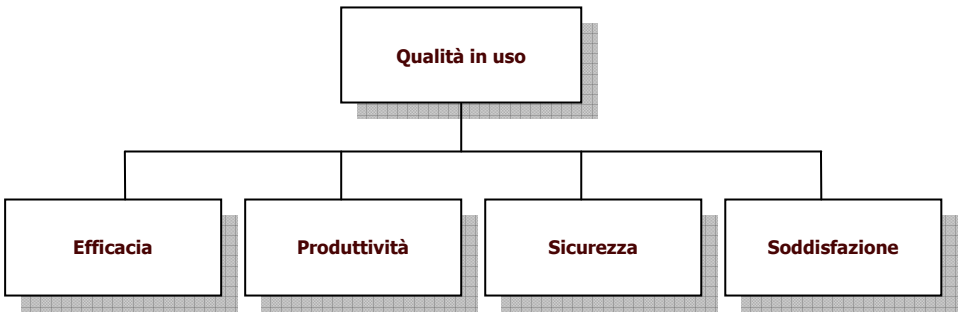

Figura 4. Modello della qualità in uso (*tratto dalle norme ISO/IEC 9126-1*).

Il "contesto di utilizzo" rappresenta le caratteristiche che definiscono la capacità di un prodotto software di consentire agli utenti di raggiungere gli obiettivi di efficacia, produttività, sicurezza e soddisfazione.

2.4.1 Efficacia (*Effectiveness*)

L'*efficacia* di un prodotto software rappresenta la capacità di permettere all'utente di raggiungere obiettivi specifici con accuratezza e completezza in uno specifico contesto di utilizzo.

2.4.2 Produttività (*Productivity*)

La *produttività* di un prodotto software rappresenta la capacità di permetter all'utente di impegnare un numero definito di risorse, in relazione all'efficienza raggiunta in uno specifico contesto di utilizzo. Le risorse considerate rilevanti includono il tempo necessario per completare il task, il tempo impiegato dall'utente per svolgere il proprio compito, i materiali o i costi finanziari per l'utilizzo.

2.4.3 Sicurezza fisica (*Safety*)

La *sicurezza fisica* di un prodotto software rappresenta la capacità di raggiungere un livello accettabile di rischio per i dati, le persone, il business, la proprietà o gli ambienti in uno specifico contesto di utilizzo. I rischi sono generalmente considerati come il risultato di mancanze nella funzionalità (inclusa la sicurezza), l'affidabilità, l'usabilità e la manutenibilità.

2.4.4 Soddisfazione (*Satisfaction*)

La *soddisfazione* di un prodotto software rappresenta la capacità di soddisfare gli utenti in uno specifico contesto di utilizzo. La soddisfazione si riferisce alla risposta data dall'utente a seguito delle interazioni con il prodotto, e include l'attitudine all'utilizzo del prodotto stesso.

3 Le metriche della qualità del software

3.1 Attributi interni ed esterni della qualità del software

Gli attributi interni di un prodotto software influenzano la qualità esterna; così esiste un legame tra caratteristiche interne e caratteristiche esterne del prodotto. In questo caso, si dice che gli attributi interni rappresentano "indicatori" delle caratteristiche esterne.

La Figura 5 mostra come una caratteristica interna può influire su più di una caratteristica esterna; inoltre, una caratteristica esterna può essere influenzata da più attributi interni.

Per esempio, l'affidabilità può essere misurata "esternamente" rilevando il numero di malfunzionamenti rilevati in un determinato intervallo di tempo durante l'esecuzione del software. La misurazione può essere fatta anche "internamente" tramite revisioni delle specifiche tecniche ed ispezioni del codice sorgente per valutare il livello di tolleranza ai guasti.

In tale modello, l'insieme di tutti gli attributi di qualità di un prodotto software sono classificati in una struttura ad albero gerarchico di "caratteristiche", "sottocaratteristiche" ed "attributi". Nella struttura, le caratteristiche rappresentano l'elemento più alto, mentre gli attributi quello più basso. La struttura gerarchica può non essere perfetta, in quanto alcuni attributi, come si è detto, possono contribuire a più sottocaratteristiche. Non esiste, quindi, una correlazione perfetta e predeterminata, ma essa deve essere costruita in base all'esperienza ed alle caratteristiche del prodotto e del particolare contesto nel quale sarà utilizzato.

Le sottocaratteristiche possono essere misurate sia da attributi interni che esterni. Analogamente, le proprietà esterne (come, ad esempio, l'adeguatezza, l'accuratezza, la tolleranza ai guasti, il comportamento nel tempo) potranno influenzare la qualità osservata, cioè quella percepita dall'utente (per esempio, l'efficacia, la produttività, l'operatività).

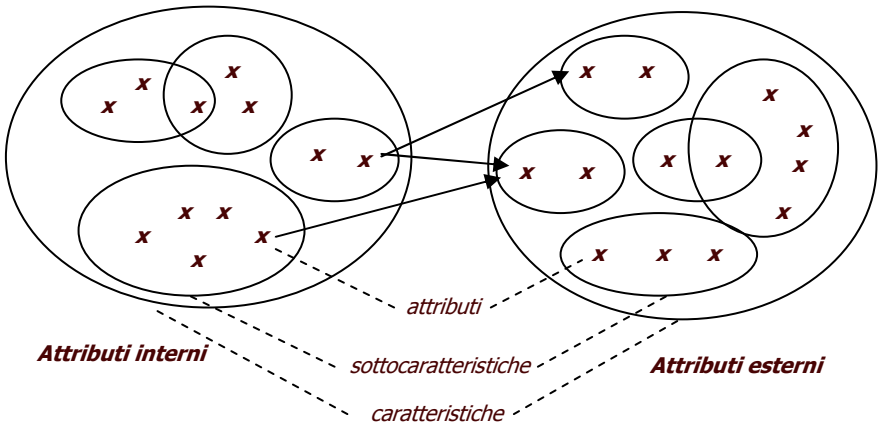

Figura 5. Caratteristiche, sottocaratteristiche ed attributi di qualità.

3.2 Metriche interne

Le metriche interne si applicano al software non eseguibile (come, ad esempio, le specifiche tecniche ed il codice sorgente) durante le fasi di progettazione e codifica. Durante le fasi di sviluppo del software, quindi, i prodotti intermedi sono valutati tramite metriche interne che misurano le proprietà intrinseche del prodotto. Tali metriche includono quelle che possono misurare il comportamento del prodotto finale tramite simulazioni. Le misure effettuate permettono di prevedere il livello di qualità esterna ed in uso del prodotto finale in quanto, come detto in precedenza, gli attributi interni influenzano le caratteristiche esterne e quelle in uso. Risulta quindi molto importante la scelta degli attributi da misurare. Le metriche interne permettono ad utenti, valutatori (testatori) e sviluppatori di indirizzare per tempo - prima che sia realizzato il software eseguibile - eventuali problemi che potrebbero inibire la qualità finale del prodotto.

Le metriche interne misurano attributi interni del software e forniscono indicazioni sulle caratteristiche esterne del prodotto finale, tramite l'analisi statica dei prodotti intermedi (specifiche tecniche e codice sorgente). Le misure effettuate con le metriche interne si basano sui numeri o le frequenze degli elementi che compongono il software e che sono rilevati, per esempio, nel codice sorgente, nei grafi di controllo, nel flusso dei dati, nelle rappresentazioni delle transizioni di stato, ecc. La norma ISO/IEC 9126-3 presenta una varietà di metriche interne su cui operare la scelta.

Le metriche interne si applicano anche alla documentazione del prodotto.

3.3 Metriche esterne

Le metriche esterne misurano i comportamenti del prodotto software rilevabili dai test, dall'operatività, dall'osservazione durante la sua esecuzione. L'esecuzione del prodotto software è fatta in base al suo utilizzo in funzione degli obiettivi di business stabiliti ed in un contesto organizzativo e tecnico rilevante.

Le metriche esterne sono scelte in base alle caratteristiche che il prodotto finale dovrà dimostrare durante la sua esecuzione in esercizio. La norma ISO/IEC 9126-2 presenta un numero consistente di metriche esterne su cui operare la scelta. Esse forniscono ad utenti, tester e sviluppatori elementi concreti per misurare e valutare le caratteristiche esterne e comportamentali del prodotto finale durante il suo utilizzo.

3.4 Relazione tra metriche interne ed eterne

Una volta specificati i requisiti di qualità del prodotto software, si identificano le caratteristiche e sottocaratteristiche di qualità che contribuiscono ad indirizzare i requisiti elencati. Quindi, si definiscono le metriche esterne ed i relativi livelli ed intervalli di accettazione (*target*, *range*) per quantificare i criteri che permettano di valutare se il prodotto finale avrà raggiunto gli obiettivi. Dopo, si definiscono gli attributi di qualità interni del software che permettano di pianificare, in anticipo, le caratteristiche esterne e quelle in uso del prodotto finale, e di realizzarle durante le fasi di sviluppo. La definizione di metriche interne, intervalli e valori di soglia permette di quantificare gli attributi interni del software e di verificare, durante lo sviluppo del software, che i prodotti intermedi abbiano la qualità richiesta.

La raccomandazione è di scegliere le metriche interne che maggiormente influenzano le caratteristiche esterne e quelle in suo del prodotto finale, in modo che esse possano predire quanto più possibile il risultato finale. E' molto difficile definire un modello teoretico rigoroso che permetta di costruire le relazioni tra metriche interne ed esterne. Occorre esperienza e senso pratico.

3.5 Metriche della qualità in uso

Le metriche della "qualità in uso" misurano il grado con cui il prodotto software permette agli utenti di svolgere le proprie attività con efficacia, produttività, sicurezza e

soddisfazione nel contesto operativo previsto. La valutazione della qualità finale del prodotto software è quindi fatta in specifici scenari d'uso (*user-task scenarios*).

La qualità in uso rappresenta la vista esterna che l'utente ha del prodotto ed è misurata in base ai risultati ottenuti dal suo utilizzo, piuttosto che da caratteristiche interne al software. Essa, però, è anche l'effetto combinato di più caratteristiche interne ed esterne di qualità.

La relazione tra la qualità in uso e le altre caratteristiche di qualità del prodotto software dipendono dal suo utilizzo e dai diversi tipi di utente:

- l'utente finale, per esempio, è sensibile principalmente alle caratteristiche di funzionalità, affidabilità, usabilità, efficienza;
- l'addetto alla manutenzione, invece, punterà l'attenzione sulle caratteristiche di manutenibilità;
- il responsabile del porting del prodotto negli ambienti previsti, infine, verificherà le caratteristiche di portabilità.

La norma ISO/IEC 9126-4 fornisce una varietà di metriche da scegliere per misurare la qualità in uso.

3.6 Scelta delle metriche e dei criteri di misurazione

La scelta delle metriche dipende dagli obiettivi di business definiti per il prodotto e dalle necessità del valutatore. Tali necessità sono specificate dai criteri di misura. Il modello definito dalle norme ISO/IEC 9126 supporta una grande varietà di requisiti, come, ad esempio:

- un utente può misurare l'adeguatezza del prodotto software utilizzando le metriche della qualità in uso;
- un acquisitore potrebbe valutare il prodotto software utilizzando metriche di qualità esterne come funzionalità, affidabilità, usabilità ed efficienza;
- un addetto alla manutenzione potrebbe valutare il prodotto software utilizzando le metriche relative alla manutenibilità;
- una persona responsabile di installare il prodotto software in un ambiente specifico potrebbe utilizzare le metriche relative alla portabilità;
- uno sviluppatore potrebbe valutare il prodotto software in base alle caratteristiche di qualità interne.

Le norme ISO/IEC 14598-1 forniscono una guida alla scelta di metriche per la valutazione di un prodotto software.

3.7 Metriche utilizzate per la comparazione

Nel riportare i risultati di un'attività di comparazione tra prodotti software, o diversi criteri di valutazione, occorre tener presente che il rapporto deve evidenziare che le metriche siano oggettive, empiriche ed utilizzino valori conosciuti e siano riproducibili.

Le comparazioni risultano affidabili quando le metriche utilizzate sono rigorose. Le procedure utilizzate devono permettere di misurare le caratteristiche (o sotto-caratteristiche) di qualità del prodotto software con sufficiente accuratezza. Bisogna anche tener conto dei possibili errori dovuti all'utilizzo di strumenti e ad errori umani.

Le misurazioni, si è detto, debbono essere oggettive, empiriche e con l'utilizzo di scale di valori valide, e riproducibili.

- Per essere *oggettive*, le misure devono fare riferimento a procedure scritte che definiscano le caratteristiche e gli attributi del prodotto software.
- Per essere *empiriche*, i dati devono essere ottenuti da osservazioni o questionari psicometrici validi.
- Per utilizzare una *scala di valori valida*, i dati devono fare riferimento ad elementi di uguale valore e conosciuti. Utilizzando liste di controllo (*checklist*) per generare i dati occorre "pesare" gli elementi utilizzati.
- Per essere *riproducibili*, le procedure per la misurazione devono permettere di ottenere gli stessi valori (con la dovuta tolleranza) a fronte di misurazioni identiche eseguite da persone diverse.

Le metriche interne devono avere anche una validità predittiva, in quanto sono correlate a misure esterne di cui si ha bisogno. Così, la misurazione di un particolare attributo di qualità interna dovrebbe avere una qualche correlazione con alcune caratteristiche esterne misurabili. Anche l'assegnazione dei valori deve essere correlata. Per esempio, ad un valore alto della qualità del prodotto software deve corrispondere un altrettanto alto valore della soddisfazione dell'utente.

4 Utilizzo del modello

Di seguito è descritto come può essere utilizzato il modello della qualità ISO/IEC 9126 e le relative metriche nello sviluppo del software in modo da raggiungere gli obiettivi di qualità attesi dagli utenti. L'adozione delle norme nello sviluppo del software prescinde dal particolare modello di ciclo di vita utilizzato.

4.1 Processo di sviluppo e di qualità

Ogni modello di ciclo di vita del software contempla fasi concettualmente simili con analoghi contenuti. Le fasi sono spesso chiamate con nomi differenti nei diversi modelli, il contenuto è invece simile.

In sintesi, il modello[3] cui si fa riferimento in questo documento contempla le seguenti fasi:

- Analisi dei requisiti
- Progettazione del prodotto (progettazione dell'architettura del sistema software)
- Progettazione di dettaglio
- Codifica e test unitario
- Integrazione del prodotto e collaudo funzionale
- Collaudo del sistema
- Rilascio in esercizio.

La Tabella 2 riassume gli elementi del ciclo di vita relativi alla qualità del software da prendere in considerazione e sviluppare adeguatamente.

[3] Il modello di ciclo di vita cui si fa riferimento nel documento è quello descritto nelle norme ISO/IEC 12207: Information Technology –Life Cycle Management – System Life Cycle Processes e nelle relative guide all'utilizzo del modello.

Tabella 2. Elementi di qualità nel modello di ciclo di vita del software.

Phase	Analysis	Macro design	Detailed design	Coding & UT	Integration test	System test	Delivery
Riferimento al modello 9126	Requisiti di qualità in uso. Requisiti di qualità esterna. Requisiti di qualità interna.	Qualità in uso "predetta". Qualità esterna "predetta". Qualità interna "misurata".	Qualità in uso "predetta". Qualità esterna "predetta". Qualità interna "misurata".	Qualità in uso "predetta". Qualità esterna "predetta" e "misurata". Qualità interna "misurata".	Qualità in uso "predetta". Qualità esterna "predetta" e "misurata". Qualità interna "misurata".	Qualità in uso "predetta". Qualità esterna "misurata". Qualità interna "misurata".	Qualità in uso "misurata". Qualità esterna "misurata". Qualità interna "misurata".
Principali prodotti di fase	Documento dei requisiti di qualità (interna, esterna ed in uso).	Documento di progettazione architetturale.	Documento di progettazione di dettaglio.	Codice. Risultato dei test unitari.	Prodotto software. Risultato dei test di integrazione.	Sistema software. Risultato dei test di sistema.	Prodotto finale nell'ambiente di esercizio.
Misurazioni sui prodotti di fase	Metriche di qualità interna.	Metriche di qualità interna.	Metriche di qualità interna.	Metriche di qualità esterna. Metriche di qualità interna.	Metriche di qualità esterna. Metriche di qualità interna.	Metriche qdi ualità esterna. Metrichedi qualità interna.	Metriche di qualità in uso. Metriche di qualità esterna. Metriche di qualità interna.

Di seguito sono descritti gli elementi di qualità più importanti da tenere presente in ciascuna fase del ciclo di vita.

4.1.1 Analisi dei requisiti

In questa fase si collezionano, analizzano, documentano e concordano i requisiti degli utenti di quanti altri abbiano un interesse diretto (*stakeholders*) nel prodotto software da realizzare. I requisiti sono quelli esplicitamente dichiarati dagli utenti e quelli implicitamente richiesti dalle caratteristiche stesse del prodotto e dal contesto di utilizzo. I "requisiti di qualità" sono quindi da collezionare, analizzare, documentare e concordare. Essi si riferiscono a caratteristiche "interne" al prodotto, "esterne" ed "in uso". Le metriche più adatte a misurare la qualità del prodotto in questa fase sono solo quelle "interne" e possono essere applicate all'unico prodotto di fase disponibile: il documento delle specifiche dei requisiti. Nel caso sia prodotto anche un documento di specifiche funzionali (documento descrittivo piuttosto che documento dei casi d'uso), allora sarà possibile misurare anche la qualità "esterna" del prodotto. In questo caso si tratta di "predire" tale qualità in quanto non esiste, al momento, alcuna parte di prodotto che possa mostrare tali caratteristiche esterne.

4.1.2 Progettazione

In questa fase si progetta il prodotto software nella sua interezza, definendo l'architettura complessiva dell'applicazione e indirizzando tutti i requisiti documentati e concordati. In particolare, si indirizza la qualità del software dal punto di vista interno,

esterno ed in uso. In questa fase è prodotto il documento che descrive l'architettura applicativa del sistema software. Le caratteristiche di qualità che possono essere misurate in questa fase sono quelle "interne", mentre quelle "esterne" e quelle "in uso" possono essere solo "predette". Le metriche da adottare in questa fase quindi sono solo quelle applicabili al documento di progettazione disponibile e possono misurare quindi solo le caratteristiche "interne" del prodotto.

4.1.3 Progettazione di dettaglio

In questa fase si realizza la progettazione di dettaglio dell'intero prodotto in tutti i suoi componenti, sottocomponenti e moduli elementari. Anche in questo caso, si possono utilizzate solo le metriche di qualità "interna" applicabili ai prodotti di fase disponibili: i documenti di progettazione di dettaglio. Le caratteristiche relative alla qualità "esterna" e quella "in uso" possono essere invece "predette".

4.1.4 Codifica e test unitario

In questa fase è sviluppato il codice e sono eseguiti i test unitari di validazione dei singoli moduli prodotti. La misurazione della qualità è effettuata sui prodotti di fase disponibili (codice e risultati dei test unitari) ed è rivolta quindi alle caratteristiche di qualità "interna" del prodotto. Essendo il prodotto realizzato in tutti i suoi componenti, anche se disponibili in un ambiente di sviluppo, le misure possono riguardare anche alcune caratteristiche di qualità "esterna" del prodotto; altre caratteristiche di qualità "esterna", così come quelle "in uso" possono essere solo "predette". Le metriche utilizzate sono quindi quelle interne e quelle esterne.

4.1.5 Integrazione e collaudo dei componenti

In questa fase è integrato l'intero prodotto in un ambiente di collaudo ed è quindi possibile "misurare" sia le caratteristiche di qualità "interna" del prodotto sia quelle "esterne". Quelle "in uso" non possono invece essere misurate in quanto il prodotto non è realmente utilizzato dagli utenti. Tali caratteristiche possono essere quindi solo "predette". Analogamente, non tutte le caratteristiche di qualità "esterna" possono essere misurate e saranno quindi anch'esse "predette". Le metriche realmente utilizzate in questa fase sono quelle interne e quelle esterne.

4.1.6 Collaudo di sistema

In questa fase il prodotto è installato in un ambiente di collaudo simile a quello di produzione ed è utilizzato eseguendo scenari di test che simulano l'utilizzo del prodotto da parte degli utenti. Le caratteristiche di qualità che possono realmente essere misurate in questa fase sono quelle "intere" e quelle "esterne". Le caratteristiche di qualità "in uso" possono essere solo "predette", in questa fase in maniera molto precisa, simulando

l'utilizzo del prodotto da parte degli utenti in un ambiente simile a quello di produzione.

4.1.7 Rilascio in produzione

In questa fase il prodotto finale è installato nell'ambiente di esercizio ed è realmente utilizzato dagli utenti. Le caratteristiche di qualità "in uso" del prodotto sono quindi direttamente "misurate", insieme a quelle interne e a quelle esterne. Le metriche utilizzate in questa fase sono quelle interne, esterne ed in uso.

4.2 Implementazione del modello nei progetti reali

Si descrive brevemente qui di seguito un modo pratico di utilizzare il modello nei progetti reali.

Si suggeriscono i seguenti passi successivi, descritti successivamente nel dettaglio:

1. Definire gli obiettivi di qualità;
2. Identificare le metriche da adoperare ed i livelli da raggiungere;
3. Identificare i prodotti di fase su cui eseguire le misurazioni;
4. Documentare i risultati conseguiti.

4.2.1 Definire gli obiettivi di qualità

Elencare le caratteristiche di qualità del prodotto e valutare il grado di importanza che la caratteristica riveste per la valutazione finale del prodotto. La rappresentazione può essere fatta utilizzando la tabella riportata di seguito (Tabella 3). Occorre produrre una singola tabella per ciascuna categoria di qualità (interna, esterna, in uso), oppure utilizzare una sola tabella che includa tutte e tre le categorie. L'importanza può essere espressa, per esempio, con tre valori: Alta, Media e Bassa.

E' basilare che l'attribuzione dell'importanza delle caratteristiche sia sempre concordata con gli utenti. Sono loro che utilizzeranno il prodotto e valuteranno di conseguenza la qualità del prodotto, ognuno per quelle che gli competono. Così sarà l'utente finale ad esprimere la propria valutazione sulle varie caratteristiche relative alla funzionalità, all'usabilità, all'efficienza ed all'affidabilità. Il responsabile della gestione operativa dirà invece la sua sulle caratteristiche dell'installabilità, personalizzazione, migrazione, efficienza relativa all'uso delle risorse e dei materiali, ecc. Il responsabile della manutenzione, a sua volta, valuterà le caratteristiche della manutenibilità.

Tabella 3. Esempio di caratteristiche di qualità e relativa importanza.

Caratteristica	Attributi	Importanza
Funzionalità	Completezza	Alta
	Accuratezza	Alta
	Interoperabilità	Bassa
	Sicurezza	Alta
	Aderenza alla funzionalità	Media
Affidabilità	Maturità	Bassa
	Tolleranza ai guasti	Bassa
	Recuperabilità	Alta
	Aderenza all'affidabilità	Alta
Usabilità	Comprensibilità	Media
	Apprendibilità	Bassa
	Operabilità	Bassa
	Attrattività	Bassa
	Aderenza all'usabilità	Alta
Efficienza	Comportamento rispetto al tempo	Alta
	Utilizzo delle risorse	Alta
	Aderenza all'efficienza	Alta
Manutenibilità	Analizzabilità	Alta
	Modificabilità	Media
	Stabilità	Bassa
	Provabilità	Media
	Aderenza alla manutenibilità	Alta
Portabilità	Adattabilità	Alta
	Installabilità	Bassa
	Coesistenza	Mdia
	Sostituibilità	Alta
	Aderenza alla portabilità	Alta
Qualità in uso	Efficacia	Alta
	Produttività	Alta
	Sicurezza	Bassa
	Soddisfazione	Media

4.2.2 Definire le metriche ed i livelli da raggiungere

Partendo dalla tabella precedente (Tabella 3), definire per ciascuna caratteristica di qualità quali misure si intende effettuare, quali valori si intende raggiungere, quali valori siano stati effettivamente raggiunti e misurati (Tabella 4). Occorre ripetere l'operazione in ciascuna fase del ciclo di vita e per ciascuna categoria di qualità (interna, esterna, in uso). Le misure sono scelte tra quelle proposte dal modello in base alla complessità e tipologia di prodotto. Nei paragrafi successivi sono suggerite alcune misure semplici ed efficaci per le tre tipologie di qualità (interna, esterna ed in uso).

Tabella 4. Misure della qualità.

Caratteristica	Attributi	Misura	Livello richiesto	Livello raggiunto
Funzionalità	Completezza Accuratezza Interoperabilità Sicurezza Aderenza alla funzionalità			
Affidabilità	Maturità Tolleranza ai guasti Recuperabilità Aderenza all'affidabilità			
Usabilità	Comprensibilità Apprendibilità Operabilità Attrattività Aderenza all'usabilità			
Efficienza	Comportamento rispetto al tempo Utilizzo delle risorse Aderenza all'efficienza			
Manutenibilità	Analizzabilità Modificabilità Stabilità Provabilità Aderenza alla manutenibilità			
Portabilità	Adattabilità Installabilità Coesistenza Sostituibilità Aderenza alla portabilità			
Qualità in uso	Efficacia Produttività Sicurezza Soddisfazione			

4.2.3 Definire i prodotti di fase

Si tratta di sviluppare un piano delle misure in cui si identificano i prodotti di fase (*deliverable*) da utilizzare come input per le misurazioni. Per ciascun deliverable si identificano le metriche da utilizzare ed i livelli da raggiungere. Il piano può essere realizzato utilizzando la tabella mostrata nel paragrafo successivo.

4.2.4 Documentare i risultati

I risultati delle misure effettuate sono documentati e riassunti, per esempio, utilizzando la tabella riportata di seguito (Tabella 5). Essa rappresenta quindi sia il piano delle misure sia il risultato delle misure. Il documento è oggetto di valutazione e verifica del reale livello di qualità raggiunto dal prodotto.

Tabella 5. Piano delle misure della qualità e risultati ottenuti.

Sotto caratteristica	Deliverable da valutare	Metrica qualità "interna"	Metrica qualità "esterna"	Metrica qualità "in uso"
Completezza	1. Requisiti 2. Specifiche funz.	Livello di copertura delle funzioni sviluppate	Livello di soddisfazione degli utenti	Non applicabile
Accuratezza	1. Requisiti 2. Specifiche funz.	Livello di accuratezza delle funzioni sviluppate	Livello di soddisfazione degli utenti	Livello di soddisfazione degli utenti
Ecc.	1.			
Ecc.	1.			
Ecc.	1.			

4.3 Esempi di metriche

E' necessario individuare, in ciascun progetto, quali metriche risultino particolarmente semplici ed efficaci. Quali metriche siano applicabili dipenderà da molti fattori quali, ad esempio, la tipologia del prodotto (software applicativo, reale-time, di base, ecc.), dalla sua criticità per il business (mission critical, normal, ecc.), dal livello di sicurezza logica e fisica richiesta per i dati e per le persone, dal contesto di utilizzo (da parte di esperti, di personale poco esperto, ecc.), altri fattori. Le caratteristiche da valutare sono tratte dai requisiti di qualità, espressi o impliciti, e dalle specifiche.

Nel seguito sono elencate le metriche interne, esterne ed in uso più comuni distinte per categoria e sottocategoria. Gli esempi sono rappresentati in tabelle dove, per ciascuna metrica suggerita, è riportato: nome della metrica, scopo della metrica, metodo di calcolo, origine dei dati su cui effettuare le misurazioni.

4.4 Esempi di metriche interne

Le metriche della qualità "interna" del software sono utilizzate durante la fase di sviluppo e permettono di valutare il comportamento del software dal punto di vista degli sviluppatori e di predire quello che sarà dal punto di vista esterno degli utenti. Esse misurano le caratteristiche intrinseche del software, quindi, ma permettono di predire le caratteristiche esterne. Trattandosi di caratteristiche interne al prodotto, queste sono sempre valutate dal punto di vista degli sviluppatori.

Le metriche si possono applicare alle fasi di analisi, design e codifica.

I documento oggetto di valutazione sono quindi: le specifiche (dei requisiti e funzionali), la progettazione (design) ed il codice sorgente.

I dati per le misure sono estratti dai risultati delle sessioni di valutazione congiunta eseguite con gli utenti, delle revisioni tecniche interne effettuate sui documenti di specifiche e di design, e delle ispezioni del codice.

4.4.1 Funzionalità

Le "metriche interne della funzionalità" sono utilizzate per verificare in anticipo, e quindi per predire, se il software soddisferà i requisiti funzionali ed indirizzerà le necessità degli utenti.

Adeguatezza

Le "metriche interne dell'adeguatezza" consentono di valutare esplicitamente le funzioni utilizzate per completare i task prescritti e di determinare la loro adeguatezza. Una misura dell'adeguatezza può essere, per esempio, il livello di copertura delle funzionalità sviluppate rispetto a quelle richieste e descritte nel documento di specifiche funzionali.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Adeguatezza delle funzioni sviluppate	Misurare il livello di adeguatezza delle funzioni sviluppate rispetto alle esigenze specificate	M = A / B A = numero di funzioni ritenute adeguate allo svolgimento del task specificato; B = numero totale di funzioni sviluppate.	- Specifiche - Design - Codice sorgente - Revisioni tecniche
2. Completezza delle funzioni sviluppate	Misurare il livello di completezza delle funzioni sviluppate rispetto alle esigenze specificate	M = A / B A = numero di funzioni ritenute complete allo svolgimento del task specificato; B = numero totale di funzioni sviluppate. (vedi Nota)	- Specifiche - Design - Codice sorgente - Revisioni tecniche
3. Correttezza delle funzioni sviluppate	Misurare il livello di correttezza delle funzioni sviluppate rispetto alle esigenze specificate	M = A / B A = numero di funzioni ritenute corrette; B = numero totale di funzioni previste nelle specifiche.	- Specifiche - Design - Codice sorgente - Revisione tecnica
Nota: Si può misurare anche come "numero di funzioni omesse" (non sviluppate) rispetto al numero totale di funzioni previste.			

Accuratezza

Le "metriche interne dell'accuratezza" consentono di valutare se i risultati ottenuti siano corretti, graditi ed in linea con le aspettative. Per esempio, la correttezza del risultato di un calcolo, oppure la consistenza tra quanto descritto nel manuale utente ed il documento di specifiche funzionali.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Accuratezza delle funzioni sviluppate.	Misurare il livello di accuratezza con cui sono sviluppate le funzionalità richieste.	M = A / B A = numero di funzioni sviluppate con l'accuratezza richiesta; B = numero totale di funzioni sviluppate.	- Specifiche - Design - Codice sorgente - Revisioni tecniche
2. Accuratezza dei dati.	Misurare il livello di accuratezza dei dati sviluppati.	M = A / B A = numero di dati sviluppati con l'accuratezza (correttezza e precisione) richiesta; B = numero totale di dati previsti.	- Specifiche - Design - Codice sorgente - Revisioni tecniche

Interoperabilità

Le "metriche interne dell'interoperabilità" consentono di valutare la capacità del prodotto software di interagire con gli altri sistemi definiti.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Correttezza delle interfacce e dei protocolli sviluppati	Misurare il livello di correttezza delle interfacce sviluppate e dei protocolli implementati rispetto alle esigenze specificate	M = A / B A = numero di interfacce o protocolli sviluppati correttamente; B = numero totale di interfacce e protocolli previsti nelle specifiche. (vedi Nota)	- Specifiche - Design - Codice sorgente - Revisioni tecniche
Nota: Può misurare anche il "numero di interfacce o protocolli omessi, errati o incompleti".			

Sicurezza

Le "metriche interne della sicurezza" consentono di valutare la capacità del software ad evitare che si verifichino accessi non consentiti al sistema ed ai dati.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Controllo degli accessi	Misurare il livello di controllo effettuato sugli accessi al sistema rispetto alle esigenze specificate	M = A / B A = numero di accessi (log in) controllati con successo dal sistema; B = numero totale di accessi controllati previsti nelle specifiche.	- Specifiche - Design - Codice sorgente - Revisioni tecniche
2. Prevenzione del danneggiamento dei dati	Misurare il livello di controllo effettuato per prevenire il danneggiamento dei dati	M = A / B A = numero di controlli effettuati con successo dal sistema e relativa prevenzione di danneggiamento dei dati; B = numero totale di controlli previsti nelle specifiche.	- Specifiche - Design - Codice sorgente - Revisioni tecniche
3. Codifica/decodifica dei dati	Misurare il livello di codifica e decodifica operato sui dati	M = A / B A = numero di istanze di dati codificati e decodificati (criptati e decriptati) con successo; B = numero totale previsti nelle specifiche.	- Specifiche - Design - Codice sorgente - Revisioni tecniche

Aderenza alla funzionalità

Le "metriche interne dell'aderenza alla funzionalità" sono utilizzate per valutare (e predire) se il prodotto software risulta aderente a standard, convenzioni, regolamentazioni dell'organizzazione degli utenti relativamente alle funzionalità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza delle funzioni e/o delle interfacce	Misurare il livello di aderenza delle funzioni e delle interfacce sviluppate a quanto attiene all'argomento in termini di standard, normative e regolamentazioni	M = A / B A = numero di funzioni (e/o interfacce) sviluppate che risultano aderenti a standard, regole e normative emesse al riguardo; B = numero totale di funzioni (e/o interfacce) che devono essere aderenti a tali regole come descritto nelle specifiche.	- Standard, convenzioni, regolamentazioni - Specifiche - Design - Codice sorgente - Revisioni tecniche

4.4.2 Affidabilità

Le "metriche interne dell'affidabilità" sono utilizzate per predire se il software in questione potrà soddisfare i requisiti prescritti per l'affidabilità dal punto di vista degli sviluppatori.

Maturità

Le "metriche interne della maturità" permettono di valutare la maturità del prodotto software in termini di riduzione delle possibili cause di malfunzionamenti.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Rilevamento dei difetti	Misurare l'efficacia nel rilevare i difetti presenti nel software durante le diverse fasi di sviluppo del prodotto	M = A / B A = numero di difetti rilevati nelle revisioni tecniche, ispezioni e test del prodotto in ciascuna fase di sviluppo; B = numero totale di difetti previsti nella fase di sviluppo. (vedi Nota 1)	A: Revisioni tecniche, ispezioni, test B: DB storico
2. Rimozione dei difetti	Misurare l'efficacia della rimozione degli errori rilevati nelle diverse fasi di sviluppo del prodotto	M = A / B A = numero di difetti corretti; B = numero totale di difetti rilevati nelle diverse fasi di sviluppo del prodotto. (vedi Nota)	A: Rapporto rimozione errori B: Rapporto revisioni tecniche, ispezioni, test
3. Copertura dei test	Misurare il livello di copertura dei test	M = A / B A = numero di casi di test pianificati; B = numero totale di casi di test necessari a garantire il livello di copertura richiesto dalla tipologia e dalla criticità del prodotto.	A: Piano di test B: Specifiche

4. Strutturazione del software	Misurare il livello coesione dei componenti progettati e del codice dei singoli moduli sviluppati	M = Livello di coesione di un componente o di un singolo modulo secondo una scala a 7 valori decrescenti (vedi Nota 2)	Rapporto revisione tecnica
	Misurare il livello disaccoppiamento dei componenti progettati e del codice dei singoli moduli sviluppati	M = Livello di disaccoppiamento di un componente o di un singolo modulo secondo una scala a 6 valori decrescenti (vedi Nota 3)	Rapporto revisione tecnica

Nota 1: I dati del parametro B sono ricavati dal modelli statistici creati sulla base dei dati presenti negli archivi storici dei progetti.

Nota 2: Il livello di coesione di un componente aumenta secondo la seguente scala di valori: 1. Casuale, 2. Associazione logica, 3. Temporale, 4. Procedurale, 5. Comunicazione, 6. Sequenziale, 7. Funzionale. Occorre garantire il massimo di coesione per ciascun modulo sviluppato.

Nota 3: Il livello di disaccoppiamento tra moduli aumenta secondo la seguente scala di valori: 1. Contenuto, 2. Aree dati comuni (Common area), 3. Elementi esterni, 4. Controllo, 5. Strutture dati, 6. Dato. Occorre garantire il massimo di disaccoppiamento tra i vari moduli.

Tolleranza ai guasti

Le "metriche interne della tolleranza ai guasti" permettono di valutare la capacità del prodotto software a mantenere le prestazioni desiderate anche in caso di malfunzionamenti o di violazione delle regole stabilite per le interfacce specificate.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Livello di controllo dei guasti	Misurare il numero di condizioni di errore messe sotto controllo per evitare i guasti al prodotto	M = A / B A = numero di condizioni di errore gestite correttamente; B = numero totale di condizioni di errori possibili nel sistema.	A: Rapporto revisioni tecniche B: Specifiche
2. Confidenza nella correttezza funzionale	Misurare il numero di funzioni in cui le condizioni di errore sono gestite come previsto	M = A / B A = numero di funzioni in cui le condizioni di errore sono gestite correttamente; B = numero totale di funzioni in cui è previsto ci siano condizioni di errori.	A: Rapporto revisioni tecniche B: Specifiche

Recuperabilità

Le "metriche interne della recuperabilità" permettono di valutare la capacità del prodotto software di ristabilite un adeguato livello di prestazioni in caso di malfunzionamento con il minimo possibile di perdita di dati.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Grado di recuperabilità	Misurare la capacità del prodotto a recuperare le condizioni di errore	M = A / B A = numero di situazioni in cui il prodotto ripristina con successo le condizioni iniziali; B = numero totale di situazioni in cui è previsto si debba ripristinare la situazione iniziale. (vedi Nota)	A: Rapporto revisione tecnica B: Specifiche o documenti di design
2. Efficacia del ripristino	Misurare la rapidità dell'attività di ripristino effettuata	M = A / B A = tempo necessario a ripristinare la situazione iniziale; B = tempo previsto per il ripristino.	A: Rapporto revisione tecnica B: Specifiche o documenti di design
Nota: Le funzioni di ripristino possono essere il "checkpoint" di una transazione o di un dato, una funzione di "undo" ecc.).			

Aderenza all'affidabilità

Le "metriche interne dell'aderenza all'affidabilità" sono utilizzate per valutare (e predire) se il prodotto software risulta aderente a standard, convenzioni, regolamentazioni dell'organizzazione degli utenti relativamente all'affidabilità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza all'affidabilità	Misurare il livello di aderenza del prodotto sviluppato a standard, normative e regolamentazioni previsti per l'affidabilità	M = A / B A = numero di elementi sviluppati che risultano aderenti a tali standard, regole e normative; B = numero totale di elementi che devono risultare aderenti a tali regole come previsto dalle specifiche.	- Standard, regolamentazioni e convenzioni - Specifiche - Design - Codice sorgente - Revisioni tecniche

4.4.3 Usabilità

Le "metriche interne dell'usabilità" misurano quanto il prodotto software sia comprensibile, apprendibile, facile da usare, attrattivo e aderente a regole e linee guida di usabilità.

Deve essere possibile stabilire, per le metriche utilizzate, criteri di accettazione ed effettuare confronti con altri prodotti. Questo significa che le metriche devono riferirsi ad elementi conosciuti e di valore noto. I risultati delle misurazioni devono riportare i valori medi e le deviazioni standard.

Comprensibilità

Gli unteti devono essere in grado di selezionare un prodotto software che sia adeguato all'utilizzo che intendono farne. Le "metriche interne della comprensibilità" valutano se nuovi utenti possono comprendere:

- se il software è adeguato alle loro esigenze e
- come può essere utilizzato per particolari task.

Le metriche si applicano alle fasi di analisi e di progettazione.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Completezza descrittiva	Misurare il livello di completezza delle funzionalità descritte nella documentazione del prodotto	M = A / B A = numero di funzioni descritte nella documentazione; B = numero totale di funzioni previste.	- Specifiche - Design - Rapporto revisioni tecniche
2. Comprensibilità delle funzioni per gli utenti	Misurare quale percentuale di funzioni siano comprensibili agli utenti	M = A / B A = numero di funzioni presenti nelle interfacce utente e giudicate comprensibili da parte loro; B = numero totale di funzioni previste nelle interfacce utente.	- Specifiche - Design - Rapporti revisioni tecniche

Apprendibilità

Le "metriche interne dell'apprendibilità" valutano l'efficacia dell'aiuto in linea (help on-line) e della documentazione. L'apprendibilità è strettamente legata alla comprensibilità. Le metriche relative a questa ultima caratteristica, infatti, possono essere utilizzate come indicatori del grado di apprendibilità del prodotto software.

La metrica è applicabile alle fasi di analisi, di progettazione e di codifica.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Completezza della documentazione utente e dell'aiuto in linea	Misurare la percentuale di funzioni documentate nei manuali utente e dell'aiuto in linea (help on-line)	M = A / B A = numero di funzioni sviluppate adeguatamente documentate nei manuali utente e con relativo aiuto in linea; B = numero totale di funzioni previste nel prodotto.	- Specifiche - Design - Rapporti revisioni tecniche

Operabilità

Le "metriche interne dell'operabilità" valutano se gli utenti sono in grado di operare con il prodotto e tenerlo sotto controllo. Le metriche dell'operabilità di un software appartengono a categorie che seguono i principi dei dialoghi definiti nelle norme ISO 9241-10:

- adeguatezza del software a completare i compiti (task) richiesti;
- auto-descrizione del software;
- controllabilità del software;
- conformità del software alle aspettative degli utenti;
- tolleranza ai guati del software;
- adeguatezza del software alla personalizzazione.

La scelta delle funzioni da testare dipende dalla loro frequenza di utilizzo, dalla criticità per il business, da eventuali problemi di usabilità evidenziati.

Le metriche si applicano alle fasi di analisi, di progettazione e di codifica.

Esempi:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Validità dei dati d'input	Misurare il livello di correttezza dei dati forniti in input all'applicazione	M = A / B A = numero di dati di input di cui si effettua il controllo di validità; B = numero totale di dati di input previsti.	- Specifiche - Design - Rapporti revisioni tecniche
2. Cancellazione delle operazioni improprie	Misurare la capacità del prodotto a permettere la cancellazione delle operazioni effettuate e ritenute improprie. La cancellazione può essere eseguita, per esempio, con una funzione di "undo"	M = A / B A = numero di operazioni che possono essere cancellate, una volta eseguite, se ritenute improprie; B = numero totale di operazioni che possono essere cancellate, una volta eseguite, secondo le specifiche.	- Specifiche - Design - Rapporti revisioni tecniche
3. Livello di personalizzazione	Misurare il livello di personalizzazione permesso per le funzioni previste (vedi Nota)	M = A / B A = numero di personalizzazioni sviluppate; B = numero di personalizzazioni previste.	- Specifiche - Design - Rapporti revisioni tecniche

4. Controllo e monitoraggio delle operazioni	Misurare la capacità del prodotto di controllare e monitorare lo stato delle operazioni eseguite	M = A / B A = numero di funzioni sviluppate con un adeguato controllo e monitoraggio delle operazioni; B = numero totale di funzioni per le quali è previsto il controllo nelle specifiche.	- Specifiche - Design - Rapporti revisioni tecniche
5. Qualità della messaggistica	Misurare il grado di chiarezza, completezza e correttezza dei messaggi previsti rispetto alle diverse condizioni gestite dal prodotto come, ad esempio, il completamento di una funzione, le condizioni di errore, le scelte da effettuare, ecc.	M = A / B A = numero di messaggi che risultano chiari, completi e corretti; B = numero totale di messaggi previsti.	- Specifiche - Design - Rapporti revisioni tecniche
6. Tolleranza del prodotto agli errori commessi dagli utenti	Misurare il livello di tolleranza del prodotto relativamente agli errori commessi dagli utenti. Si tratta di condizioni di errore frequenti che il software può rilevare, correggere automaticamente e segnalare con opportuno messaggio.	M = A / B A = numero di messaggi che risultano chiari, completi e corretti; B = numero totale di messaggi previsti.	- Specifiche - Design - Rapporti revisioni tecniche
Nota: La metrica può essere particolarmente utile per valutare soluzioni di situazioni di particolare handicap.			

Attrattività

Le "metriche interne dell'attrattività" valutano come il software appare agli utenti e sono influenzate dalla progettazione, dalla grafica, dai colori. Tali caratteristiche sono particolarmente importanti per i prodotti di largo consumo.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Attrattività delle interfacce utente	Misurare quanto attrattive risultino le interfacce agli utenti dal punto di vista grafico	M = V (q) Valore medio dei risultati di un questionario compilato da almeno tre utenti (vedi Nota 1). Può essere utilizzata una scala a quattro valori: Molto attrattivo, Attrattivo, Poco attrattivo, Non attrattivo.	- Questionario - Specifiche - Design - Rapporti revisioni tecniche
2. Personalizzazione dell'attrattività	Misurare la capacità del prodotto nel permettere agli utenti di personalizzare le interfacce grafiche in modo da influire sul livello di attrattività. (vedi Nota 2)	M = A / B A = numero di interfacce rese personalizzabili; B = numero totale di interfacce previste personalizzabili nelle specifiche.	- Specifiche - Design - Rapporti revisioni tecniche

Nota 1: Il questionario è preparato in modo da indirizzare le caratteristiche del prodotto ritenute importanti per il contesto di utilizzo del prodotto.

Nota 2: La metrica, molto utile nei casi in cui la grafica delle interfacce utente sia particolarmente critica ai fini del raggiungimento della soddisfazione dell'utenza, si applica alle fasi di progettazione e di codifica.

Aderenza all'usabilità

Le "metriche dell'aderenza all'usabilità" valutano l'aderenza a standard, convenzioni, guide allo stile, regolamentazioni relativi all'usabilità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza all'usabilità	Misurare il livello di aderenza del prodotto sviluppato a standard, normative e regolamentazioni previsti per l'usabilità	M = A / B A = numero di elementi sviluppati che risultano essere aderenti a tali standard, regole e normative; B = numero totale di elementi previsti nelle specifiche che devono risultare aderenti a tali regole	- Standard, regolamentazioni e normative - Specifiche - Design - Codice sorgente - Rapporti revisioni tecniche

4.4.4 Efficienza

Le "metriche interne dell'efficienza" sono usate per predire l'efficienza del comportamento del prodotto software durante il test o l'operatività dal punto di vista degli sviluppatori. Per misurare l'efficienza, devono essere stabilite determinate condizioni come, ad esempio, la definizione della configurazione hardware e software dell'ambiente di riferimento come descritto nelle specifiche software. Quando si citano misure di efficienza come, ad esempio i tempi di risposta del sistema, occorre sempre specificare l'ambiente di riferimento.

Comportamento rispetto al tempo

Le "metriche interne relative al comportamento del prodotto software rispetto al tempo" permettono di valutare il tempo necessario al a completare una operazione (esempio: tempo di risposta di una transazione, tempo di produzione di un report, tempo di trasmissione di un dato, ecc.).

Utilizzate in fase di progettazione, le metriche rappresentano una "stima" del tempo richiesto e sono ricavate basandosi sulla conoscenza del sistema operativo e sulle caratteristiche tecniche della progettazione del prodotto software o del codice sorgente sviluppato.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Tempo di risposta	Misurare l'efficienza con cui l'applicazione completa una transazione (task)	E' calcolato in base alle caratteristiche del sistema operativo e di quelle del progetto prima, e del codice dopo, del prodotto software	- Specifiche del sistema operativo - Stima dei tempi di esecuzione di un singolo task
2. Efficienza/Produttività	Misura la quantità di lavoro effettuata nell'unità di tempo	E' calcolato come numero di transazioni (task) completati nell'unità di tempo	- Specifiche del sistema operativo - Stima dei tempi di esecuzione dei singoli task
3. Tempo di completamento di un lavoro	Misura il tempo necessario a completare un lavoro costituito da più passi (task).	E' calcolato come somma dei tempi di esecuzione di tutti i task necessari a completare un lavoro	- Specifiche del sistema operativo - Stima dei tempi di esecuzione dei singoli task

Utilizzo delle risorse

Le "metriche interne relative all'utilizzo delle risorse" permettono di valutare le risorse hardware e software utilizzate dal sistema per completare le funzioni previste. Alcuni esempi di risorse: dimensioni della memoria occupata dall'applicativo, numero e dimensione dei buffer utilizzati, numero di stampanti impegnate per produrre un report, numero di linee occupate per la trasmissione dei dati, numero di bit trasmessi per transazione, ecc.

In fase di progettazione le metriche forniscono una "stima" dell'utilizzo delle risorse calcolata in base alle caratteristiche tecniche del sistema operativo e di quelle dell'applicativo progettato. Quando è disponibile il codice sorgente è possibile effettuare stime più accurate.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Utilizzo di risorse	Misurare la quantità di risorse (input/output, elaborazione, trasmissione) utilizzate dal sistema per completare una singola attività (task) od un lavoro completo (insieme di più task)	E' calcolato come numero e dimensione delle risorse impiegate per svolgere un determinato task o un insieme di task.	- Caratteristiche tecniche del sistema operativo - Stima delle risorse e loro impegno

Aderenza all'efficienza

Le "metriche interne dell'aderenza all'efficienza" valutano l'aderenza a standard, convenzioni, guide allo stile, regolamentazioni definiti dall'organizzazione dell'utente relativamente all'efficienza.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza all'efficienza	Misurare il livello di aderenza del prodotto sviluppato a standard, normative e regolamentazioni previsti per l'efficienza	M = A / B A= numero di elementi sviluppati che risultano essere aderenti a tali standard, regole e normative; B = numero totale di elementi previsti nelle specifiche che devono risultare aderenti a tali regole.	- Standard, regolamentazione e normative - Rapporto revisioni tecniche

4.4.5 Manutenibilità

Le "metriche interne di manutenibilità" sono usate per predire il livello di impegno (*effort*) richiesto per modificare il prodotto software dal punto di vista degli sviluppatori.

Analizzabilità

Le "metriche interne dell'analizzabilità" permettono di valutare l'impegno delle persone addette alla manutenzione, degli utenti coinvolti e delle risorse utilizzate per analizzare i problemi, effettuare le diagnosi delle cause dei malfunzionamenti ed individuare le parti del prodotto software da modificare.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Registrazioni	Misurare il livello di registrazione delle operazioni eseguite nel file di log	M = A / B A = numero di operazioni effettivamente registrate; B = numero totale previsto dalle specifiche.	A: Rapporti revisioni tecniche B: Specifiche
2. Diagnostica	Misurare il livello di diagnostica che il prodotto consente tramite le apposite funzioni	M = A / B A = numero di funzioni di diagnostica sviluppate; B = numero totale di funzioni di diagnostica previste nelle specifiche.	A: Rapporti revisioni tecniche B: Specifiche

Modificabilità

Le "metriche interne della modificabilità" permettono di valutare la facilità con cui poter effettuare le modifiche ed il livello di documentazione (registrazione) delle modifiche effettuate al prodotto software.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Complessità del software	Misurare la complessità ciclomatica dei singoli moduli sviluppati (vedi Nota 1)	M = v(G) = e – n + 2 G = grafo del modulo e = cammino n = nodo	Rapporto revisione tecnica
1. Registrazione delle modifiche	Misurare se tutte le modifiche apportate al software sono commentate nei singoli moduli e nella documentazione tecnica (vedi Nota 2)	M = A / B A = numero di modifiche commentate nel codice e nella documentazione tecnica; B = numero totale di modifiche eseguite.	A: Rapporti revisioni tecniche, Sistema di controllo della configurazione, Registrazione delle versioni B: Specifiche

Nota 1: La complessità ciclomatica di un modulo software è una metrica definita da McCabe ed è misurata sulla base del suo grafo tramite l'ausilio di uno strumento apposito. Risulterebbe infatti molto oneroso calcolarlo manualmente per un numero grande di moduli. Tale metrica si applica anche alla sottocaratteristica "Maturità" della caratteristica "Affidabilità" del software vista prima.

Nota 2: La metrica, quando applica alla fase di progettazione si riferisce alla sola documentazione tecnica; quando applicata alla fase di codifica si riferisce sia alla documentazione tecnica che al codice.
In entrambi i casi è rilevata tramite le revisioni tecniche interne e l'ispezione del codice.

Stabilità

Le "metriche interne della stabilità" permettono di valutare quanto stabile risulterà il prodotto software dopo le modifiche.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Impatto delle modifiche	Misurare l'impatto negativo sulla corretta esecuzione del software procurato dalle modifiche al codice (vedi Nota)	M = A / B A = numero di modifiche che hanno procurato un malfunzionamento del software o hanno influito negativamente sulle prestazioni; B = numero totale di modifiche eseguite.	A: Rapporti revisioni tecniche B: Registro delle modifiche
2. Estensione dell'impatto delle modifiche	Misura l'entità dell'impatto di una modifica calcolato in base al numero di elementi coinvolti nella modifica stessa (vedi Nota)	M = A / B A = numero di elementi impattati dalla modifica; B = numero totale di elementi presenti nel software.	A: Rapporti revisioni tecniche B: Sistema di gestione della configurazione
Nota: Il calcolo è effettuato in base ai risultati delle revisioni tecniche interne eseguite per valutare le modifiche tecniche al software.			

Provabilità

Le "metriche interne della provabilità" (testabilità) permettono di valutare la capacità di testare il prodotto quando venga modificato.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Completezza delle build per il test	Misurare il numero di build necessarie per eseguire i test delle funzioni	M = A / B A = numero di build necessarie a completare il test di funzioni; B = numero totale di build previste.	A: Revisione dei documenti B: Specifiche e documenti di design
2. Autonomia dei test	Misurare quanto siano indipendenti da altri sistemi i test eseguiti	M = A / B A = numero di dipendenze da altri sistemi che possono essere simulati (per esempio con "stub"); B = numero totale di dipendenze previste dalle specifiche tecniche del prodotto.	A: Revisione dei documenti B: Specifiche e documenti di design
3. Disponibilità di strumenti di test	Misurare la disponibilità di strumenti per eseguire i test (casi di test, scenari di test, script di test, base dati, ecc.)	M = A / B A = numero di funzioni di cui sono disponibili strumenti di test; B = numero totale di funzioni disponibili nel prodotto.	A: Documentazione di test B: Specifiche

Aderenza alla manutenibilità

Le "metriche interne dell'aderenza alla manutenibilità" valutano l'aderenza a standard, convenzioni o regolamenti definiti dall'organizzazione dell'utente relativamente alla manutenibilità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alla manutenibilità	Misurare il livello di aderenza del prodotto software a standard, normative e regolamentazioni previsti per la manutenibilità	M = A / B A = numero di elementi sviluppati che risultano essere aderenti a tali standard, regole e normative; B = numero totale di elementi previsti nelle specifiche che devono risultare aderenti a tali regole.	- Standard, convenzione e regolamentazioni - Design - Codice sorgente - Rapporto revisioni tecniche

4.4.6 Portabilità

Le "metriche interne della portabilità" sono usate per predire l'effetto del prodotto software sul comportamento del sistema o di coloro che dovranno svolgere le attività di porting, dal punto di vista degli sviluppatori.

Adattabilità

Le "metriche interne dell'adattabilità" permettono di valutare l'impegno richiesto per adattare il prodotto software in uno specificato ambiente.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Adattabilità della struttura dati	Misurare quanto è adattabile il prodotto software alle modifiche della struttura dati	M = A / B A = numero di strutture dati che sono in grado di operare senza limitazioni dopo le modifiche; B = numero totale di strutture dati che devono poter essere adattate senza limitazioni, così come definito nelle specifiche.	- Rapporto revisioni tecniche - Specifiche - Design
2. Adattabilità organizzativa	Misurare quanto adattabile sia il prodotto software quando utilizzato in un diverso contesto organizzativo dell'utente	M = A / B A = numero di funzioni in grado di operare correttamente e senza limitazioni in contesti organizzativi diversi; B = numero totale di tali funzioni previste dalle specifiche.	- Rapporto revisioni tecniche - Specifiche - Design
3. Adattabilità all'ambiente hardware	Misurare il grado di adattabilità del prodotto a diversi contesti hardware, inteso come capacità di continuare ad operare correttamente e senza limitazioni	M = A / B A = numero di funzioni svolte correttamente e senza limitazioni nei diversi ambienti previsti; B = numero totale di tali funzioni previste nelle specifiche.	- Rapporto revisioni tecniche - Specifiche - Design

4. Adattabilità all'ambiente software	Misurare il grado di adattabilità del prodotto a diversi sistemi operativi (software di base), inteso come capacità di continuare ad operare correttamente e senza limitazioni	M = A / B A = numero di funzioni svolte correttamente e senza limitazioni nei diversi sistemi operativi previsti; B = numero totale di tali funzioni previste nelle specifiche.	- Rapporto revisioni tecniche - Specifiche - Design
5. Facilità d'adattamento	Misurare la quantità di lavoro necessario ad adattare il prodotto ad un diverso ambiente	M = A / B A = numero di funzioni modificate per adattare il prodotto; B = numero totale di funzioni previste dalle specifiche per operare tale adattamento.	- Rapporto revisioni tecniche - Specifiche - Design

Installabilità

Le "metriche interne dell'installabilità" permettono di valutare l'impegno richiesto per installare il prodotto software in uno specifico ambiente dell'utente.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Facilità di impostazione iniziale	Misurare la facilità delle operazioni di impostazione iniziale dell'installazione (setup)	M = A / B A = numero di volte che occorre "ripetere" (per vari motivi) l'attività di impostazione; B = il numero totale di volte previste dalle specifiche.	Rapporto revisione tecnica Specifiche
2. Lavoro di installazione	Misurare la quantità di lavoro necessario per eseguire l'installazione del prodotto software	M = A / B A = numero di passi (step) eseguiti in maniera automatica per completare l'installazione; B = numero totale di passi previsti dalle specifiche.	Rapporto revisione tecnica Specifiche
3. Personalizzazione dell'installazione	Misurare il grado di flessibilità e di personalizzazione consentita dal prodotto software in fase di installazione in diversi contesti d'uso	M = A / B A = numero di personalizzazioni richieste in fase di installazione; B = numero totale di personalizzazioni previste dalle specifiche.	Rapporto revisione tecnica Specifiche

Coesistenza

Le "metriche interne della coesistenza" permettono di valutare l'impatto del prodotto software su altri prodotti con i quali condivide risorse hardware a livello operativo.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Capacità di coesistenza	Misurare il capacità del prodotto software a condividere risorse con altri prodotti con cui interagisce senza limitare le funzionalità o degradare le prestazioni	M = A / B A = numero di elementi effettivamente condivisi senza limitazioni o degrado di performance; B = numero totale di elementi che possono essere condivisi come previsto nelle specifiche.	- Specifiche - Rapporto esito test - Rapporto revisioni tecniche

Sostituibilità

Le "metriche interne della sostituibilità" permettono di valutare l'impegno richiesto agli utenti per utilizzare il prodotto software al posto di altri prodotti in un determinato ambiente operativo e contesto d'uso.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Continuità dei dati	Misurare la capacità di utilizzare i dati originali anche con il prodotto in sostituzione di uno precedente	M = A / B A = numero di elementi di dati effettivamente utilizzati anche con la sostituzione del prodotto; B = numero totale di elementi di dati previsti dalle specifiche	Design Codice sorgente Rapporto revisione tecnica Rapporto esito test
2. Continuità funzionale	Misurare il numero di funzioni originali che rimangono inalterate anche dopo la sostituzione del prodotto	M = A / B A = numero di funzioni effettivamente rimaste invariate dopo la sostituzione del prodotto; B = numero totale di funzioni presenti nel vecchio prodotto.	Design Codice sorgente Rapporto revisione tecnica Rapporto esito test

Aderenza alla portabilità

Le "metriche interne dell'aderenza alla portabilità" valutano l'aderenza a standard, convenzioni o regolamenti definiti dall'organizzazione dell'utente relativamente alla portabilità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alla portabilità	Misurare il livello di aderenza del prodotto software a standard, normative e regolamentazioni previsti per la portabilità	M = A / B A = numero di elementi sviluppati che risultano essere aderenti a tali standard, regolamentazioni e normative; B = numero totale di elementi previsti nelle specifiche che devono risultare aderenti a tali regole.	Standard, convenzione e regolamentazioni Design Codice sorgente Rapporto revisioni tecniche

4.5 Interpretazione delle metriche interne

Le misure possono essere interpretate non adeguatamente se le metriche non sono applicate correttamente ed i dati non raccolti accuratamente. Ma ancora più importante è il contesto in cui le metriche sono utilizzate. Sono quindi suggerite alcune linee guida per l'applicazione delle metriche e la valutazione dei risultati.

- **Differenze tra l'ambiente di test e l'ambiente di esercizio**.

  La differenza è sostanziale ed i risultati possono essere notevolmente differenti; occorre quindi fare molta attenzione. Gli elementi che possono differire sono: CPU, rete di comunicazione, sistema operativo, interfacce utente, base dati, ecc.

- **Differenze tra l'esecuzione dei test e l'operatività reale**.

  Gli elementi di differenza tra i due casi possono essere: livello di copertura funzionale, rapporto tra casi di test e funzionalità, transazioni eseguite manualmente o automaticamente, carico di lavoro (*stress load*), tempo di esecuzione (es.: 24 ore su 24, 7 giorni su 7), adeguatezza della base dati, processi periodici, utilizzo di risorse, ecc.

- **Profilo degli utenti coinvolti**.

  Il diverso profilo degli utenti coinvolti nell'utilizzo del prodotto software può determinare valutazioni moto differenti nella valutazione dei risultati delle misurazioni. Alcuni elementi di distinzione sono: composizione dei gruppi di utenti, livello di competenza (*skill*) degli utenti, utilizzo di specialisti piuttosto che utenti generici, gruppi limitati di utenti, ecc.

- **Validità dei risultati ottenuti durante i test e l'esercizio**.

  Occorre fare molta attenzione ad eventuali problemi associati con i dati utilizzati ed i risultai collezionati nell'ambiente di test ed in quello di esercizio. Alcuni elementi da tener presente sono, per esempio: procedure diverse utilizzate per collezionare i risultati da valutare (procedure manuali piuttosto che automatiche, questionari piuttosto che interviste), sorgente dei risultati da valutare (rapporti prodotti dagli sviluppatori piuttosto che dagli utenti o dai valutatori), validazione dei risultati (autocontrollo effettuati dagli sviluppatori piuttosto che ispezioni indipendenti).

- **Livello di estensione nella misurazione delle performance**. Differenze tra procedure diverse utilizzate in ambiente di test e ambiente di esercizio possono portare a risultai molto diversi nella valutazione delle prestazioni misurate. Per esempio, l'utilizzo di metriche della qualità interna utilizzate nell'ambiente di sviluppo e test per estrapolare (predire) le prestazioni finali possono portare a risultati molto diversi da quelli ottenuti con l'utilizzo di metriche della qualità in uso nell'ambiente di esercizio. E' quindi molto importante bilanciare opportunamente le procedure utilizzate per valutare i risultati delle misure effettuate nei due ambienti.
- **Correzione delle specifiche**. Occorre verificare se esitano differenze sostanziali tra le specifiche ed i requisiti applicabili all'ambiente di esercizio. Le misure effettuate durante lo sviluppo del prodotto software sono valutate in relazione a quanto riportato nelle specifiche. E' quindi importante verificare se le specifiche siano in linea con i requisiti attesi nell'ambiente di esercizio.

4.6 Esempi di metriche esterne

Le metriche relative alla qualità "esterna" indirizzano le caratteristiche esteriori del software, cioè quelle rilevabili direttamente dagli utenti e dagli operatori.

4.6.1 Funzionalità

Le metriche esterne della funzionalità si basano su due principi generali:

a) differenza tra i risultati ottenuti ed i requisiti attesi;

b) funzionalità insufficienti rispetto all'operatività richiesta.

In particolare, per ciascuna sottocaratteristica, si possono applicare metriche specifiche.

Adeguatezza:

Una "metrica esterna dell'adeguatezza" misura il livello d'insoddisfazione dimostrato dall'utente nell'eseguire le funzioni disponibili per completare un determinato tak che risulta:

a) non conforme ai requisiti, specifiche, manuale utente, ecc.

b) non appropriato alla modalità operativa dell'utente.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Adeguatezza funzionale	Misurare il livello di adeguatezza delle funzioni sviluppate	M = 1 – A/B A = Numero di funzioni che risultano non adeguate alle necessità dell'utente; B = Numero di funzioni previste nelle specifiche.	Rapporto revisione tecnica Specifiche
2. Completezza delle funzionalità sviluppate	Misurare il livello di completezza delle funzioni sviluppate	M = 1 – A/B A = Numero di funzioni omesse; B = Numero totale di funzioni previste nelle specifiche.	Rapporto revisione tecnica Specifiche
3. Correttezza funzionale	Misurare il livello di correttezza delle funzioni sviluppate e valutate in fase di test funzionale (black box testing)	M = 1 – A/B A = Numero di funzioni che rilevano errori; B = Numero totale di funzioni previste nelle specifiche.	Rapporto esito test Specifiche
4. Stabilità delle specifiche funzionali	Misurare il livello di stabilità delle specifiche dopo che il software è stato sviluppato (black box testing)	M = 1 – A/B A = Numero di funzioni modifiche nelle specifiche per correggere errori o carenze rilevati durante i test; B = Numero di funzioni descritte nelle specifiche.	Rapporto esito test Specifiche

Accuratezza:

Una "metrica esterna dell'accuratezza" misura il numero di errori rilevati come:

a) inconsistenza tra le procedure operative reali e quelle descritte nei manuali;

b) differenze tra i risultati ottenuti e quelli ragionevoli da attendersi..

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Accuratezza attesa	Misurare il livello di accuratezza "attesa" dei risultati delle funzioni in base ai casi di test progettati ed eseguiti	M = 1 – A/B A = Numero di casi di test i cui risultati effettivi sono diversi da quelli attesi (descritti nelle specifiche); B = Numero totale di casi di test progettati.	Rapporto risultato dei test Specifiche di test
2. Accuratezza rilevata	Misurare il livello di accuratezza "reale" dei risultati delle funzioni in base ai test eseguiti dagli utenti	M = 1 – A/B A = Numero di errori o discrepanze rilevati nell'output prodotto dai casi di test; B = Numero totale di casi di test eseguiti.	Rapporto risultato dei test Specifiche di test Specifiche funzionali

Interoperabilità:

Una "metrica esterna dell'interoperabilità" misura la mancanza di problemi nelle funzioni e nelle strutture dati richiesti per lo scambio di dati e di comandi tra il prodotto ed altre applicazioni con cui il prodotto stesso interagisce ed è connesso. La caratteristica è misurata tramite il rilevamento del numero di problemi incontrati nell'eseguire le funzioni richieste.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Correttezza dello scambio dei dati (correttezza dei formati e dei contenuti)	Misurare il livello di correttezza del formato e del contenuto dei dati scambiati dal prodotto con altri applicativi	M =1 – A/B A = Numero di errori o mancanze rilevati nello scambio dei dati con altre applicazioni; B = Numero totale di funzioni di interfaccia previste nel prodotto.	Rapporto esito test Specifiche
2. Correttezza dello scambio dei dati (percezione degli utenti)	Misurare il livello di correttezza dei dati scambiati con altri applicativi così come percepito dagli utenti	M =1 – A/B A = Numero di volte in cui il prodotto fallisce nello scambiare dati con altri applicativi; B = Numero totale di volte in cui l'utente cerca di scambiare dati.	Rapporto esito test Specifiche

Sicurezza:

Una "metrica esterna della sicurezza" misura il numero di problemi di sicurezza o il numero di funzioni con problemi di sicurezza, che falliscono nel:

a) proteggere dati o informazioni;

b) prevenire la perdita di dati o informazioni;

c) proteggere da accessi o operazioni illegali.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Verificabilità del sistema	Misurare la capacità di registrazione degli accessi al sistema ed ai dati	M = A/B A = Numero di elementi registrati nell'archivio storico a fronte di accessi al sistema ed ai dati; B = Numero totale di accessi eseguiti nel corso del test.	Rapporto esito test Specifiche di test Archivio storico
2. Controllo degli accessi	Misurare quanto siano controllati gli accessi al sistema	M = A/B A = Numero di accessi illegali controllati ed impediti; B = Numero totale di accessi illegali provati durante il test.	Rapporto esito test Specifiche di test
3. Prevenzione della perdita dei dati	Misurare la capacità di prevenire la perdita dei dati, *oppure* Misurare la frequenza con cui si verifica una perdita di dati	M = 1 – A/B A = Numero di volte in cui si verifica una perdita di dati; B = Numero di volte in cui si tenta di distruggere i dati.	Rapporto esito test Specifiche di test

Aderenza:

Una "metrica esterna dell'aderenza" misura il numero di funzioni che risultano essere conformi a standard, convenzioni e regolamenti, leggi, ecc. relativi alla funzionalità in oggetto.

L'approccio suggerito consiste in:

- identificare tutti gli elementi oggetto di verifica di aderenza;
- progettare casi di test per verificare tale aderenza;
- eseguire i casi di test e verificare gli elementi che rispettano e quelli che non rispettano gli standard;
- produrre un report con i risultati delle verifiche.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza funzionale	Misurare quanto le funzionalità del prodotto risultino aderenti alle regolamentazioni, standard, normative applicabili al prodotto	M = A/B A = Numero di elementi che risultano aderenti; B = Numero di elementi verificati.	- Specifiche di aderenza e relativi standard, ecc. applicabili - Specifiche di test - Rapporto esito test
2. Aderenza delle interfacce	Misurare quanto le interfacce risultino conformi a regolamentazioni, standard e convenzioni in materia	M = A/B A = Numero di interfacce realizzate correttamente; B = Numero di interfacce che devono essere conformi.	- Specifiche di aderenza e relative norme applicabili - Specifiche di test - Risultato dei test

4.6.2 Affidabilità

Le "metriche esterne dell'affidabilità" devono poter misurare gli attributi relativi al comportamento del prodotto durante l'esecuzione dei test dimostrando il livello di affidabilità del prodotto stesso quando opererà nel sistema in cui è previsto debba operare.

In particolare, per ciascuna sottocaratteristica, si possono applicare metriche specifiche.

Maturità:

Una "metrica esterna della maturità" misura quanto il prodotto sia esente da malfunzionamenti causati da errori nel software.

La difettosità del software in una determinata fase del ciclo di sviluppo si misura come rapporto tra il numero di errori rilevati e le dimensioni del prodotto. Essa può essere misurata solo al termine di una fase e dopo avere contato il numero totale di errori rilevati nella fase (per esempio, al termine dei test). La previsione del numero di errori che potranno essere rilevati in una fase successiva è fatta adoperare modelli di previsione basati, per esempio, su dati statistici..

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Difettosità residua	Misurare quanti errori ci sono ancora nel software e che potranno essere rilevati in seguito come malfunzionamenti durante l'utilizzo del prodotto in esercizio	M = [(S – E) / D][4] S = Numero totale di errori latenti stimati; E = Numero totale di errori effettivamente rimossi; D = Dimensione del software (KLocs o FP).	- Rapporto esito test - Rapporto problemi rilevati
2. Malfunzionamento residuo	Misurare il numero di malfunzionamenti rimasti nel prodotto	Stessa metrica vista prima ma riferita ai "malfunzionamenti" invece che agli "errori"	Stessa origine dei dati

[4] Qualora il numero di errori effettivamente rilevati (E) risulti superiore alla stima effettuata (S), occorre verificare se la discrepanza sia dovuta ad un nostro problema nel rilevare gli errori o nell'applicare il modello, oppure si tratti di inadeguatezza del modello. Nel secondo caso occorrerà trovare un modello più adatto al nostro ambiente. Esistono diversi modelli statistici; occorre conoscerli, sperimentarli e, a volte, usarne più di uno prima di effettuare la stima più attendibile.

3. Tempo medio di caduta dell'applicativo	Misurare il tempo medio che intercorre tra due malfunzionamenti successivi (MTBF)[5]	M = OP/OS OP = Tempo totale di operatività OS = Tempo totale di osservazione	- Rapporto esito test - Rapporto esito operazioni
4. Copertura dei test	Misurare il livello di copertura dei test eseguiti rispetto al livello richiesto per la complessità e la criticità del prodotto software	M = A/B A = Numero di casi di test eseguiti B = Numero di casi di test necessari per garantire il livello di copertura richiesto	- Specifica di test - Piano di test - Rapporto esito test
5. Maturità dei test	Misurare la percentuale di casi di test eseguiti con successo rispetto al numero totale previsto per garantire piena copertura dei requisiti sia funzionali che qualitativi (usabilità, affidabilità, efficienza, portabilità)	M = A/B A = Numero di casi di test eseguiti con successo B = Numero totale di casi di test previsto	- Piano di test - Rapporto esito test
6. Caduta del sistema a causa dell'applicativo	Misura quanto spesso il prodotto software causa la caduta generale del sistema	M = A/B A = Numero do volte in cui è l'applicativo a causare il fermo del sistema B = Numero totale di volte in cui il sistema è fermo	- Rapporto esito test - Rapporto esito operazioni
7. Gestione delle situazioni d'errore	Misura la capacità dell'applicativo di gestire le condizioni di errore. Le condizioni di errore possono essere distinte in base alla criticità: Alta, Media, Bassa	M = A/B A = Numero di volte in cui la condizione di errore è gestita positivamente dall'applicativo B = Numero di prove eseguite simulando condizioni di errore	- Rapporto esito test - Rapporto esito operazioni
8. Funzionalità a prova di errori operativi	Misura la capacità delle singole funzioni di gestire positivamente i possibili errori operativi (evitare anche la perdita di dati e la caduta del sistema)	M = A/B A = Numero di funzioni in grado di gestire correttamente le possibili condizioni di errore B = Numero totale di funzioni provate con possibili condizioni di errore	- Rapporto esito test - Rapporto esito operazioni

[5] Si tratta della metrica più conosciuta come "Mean Time Between Failures" (MTBF).

Tolleranza ai guati:

Una "metrica esterna della tolleranza ai guasti" misura la capacità del software di mantenere il livello delle prestazioni anche in caso di problemi causati da errori operativi o violazione nelle interfacce specificate.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Frequenza/Assenza di caduta del sistema	Misura quanto spesso/quanto non spesso il prodotto si fermi a seguito di malfunzionamenti rilevati durante l'operatività del sistema	M = 1 – A/B A = Numero di cadute del sistema B = Numero di malfunzionamenti rilevati	- Rapporto esito test - Rapporto esito operazioni
2. Frequenza/Assenza di malfunzionamenti	Misura quanto correttamente siano gestite (e risolte) le situazioni in cui si verificano malfunzionamenti durante l'operatività del sistema (vedi Nota)	M = A/B A = Numero di malfunzionamenti correttamente gestiti e risolti B = Numero totale di malfunzionamenti rilevati	- Rapporto esito test - Rapporto esito operazioni
3. Capacità di gestire gli errori operativi degli utenti	Misura la percentuale di funzioni in grado di rilevare e correggere potenziali errori operativi degli utenti nell'utilizzo del prodotto	M = A/B A = Numero di funzioni che rilevano e correggono errori causati dagli utenti B = Numero totale di funzioni in cui si prevedono potenziali errori operativi da parte degli utenti	- Specifiche - Piano di test - Rapporto esito test - Rapporto esito operazioni
Nota: E' bene distinguere i malfunzionamenti in diverse categorie (per esempio, 1. Gravi (Applicazione bloccata e nessuna funzione disponibile), 2. Media (Applicazione bloccata nelle funzioni principali ma ancora utile con qualche espediente – workaround -), 3. Minore (Nessuna funzione principale è bloccata).			

Recuperabilità:

Una "metrica esterna della recuperabilità" misura la capacità di ripristinare le prestazioni in caso di problemi con il minimo di perdita di dati.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Tempo medio di recuperabilità	Misura il tempo medio richiesto al sistema per recuperare pienamente le situazioni di caduta e ripristinare le condizioni iniziali, senza perdita di dati (vedi Nota)	M = Sum(T)/N T = Tempo impiegato dal sistema per recuperare una situazione di errore N = Numero totale di cadute il sistema	- Rapporto esito test - Rapporto esito operazioni
2. Capacità di ripartenza (manuale)	Misura la capacità del sistema di ripartire entro il tempo stabilito tramite procedura manuale	M = A/B A = Numero di ripartente manuali andate a buon fine B = Numero totale di ripartenze eseguite dall'utente	- Rapporto esito test - Rapporto esito operazioni
3. Capacità di ripartenza (automatica)	Misura la capacità del sistema di ripartire automaticamente entro il tempo stabilito	M = A/B A = Numero di ripartenze automatiche completate con successo B = Numero totale di ripartenze eseguite	- Rapporto esito test - Rapporto esito operazioni
4. Efficacia delle ripartenze	Misura l'efficacia delle ripartente, cioè eseguite e completate con successo entro il tempo richiesto	M = A/B A = Numero di ripartente completate con successo entro il tempo richiesto B = Numero totale di ripartenze provate	- Rapporto esito test - Rapporto esito operazioni
Nota: E' bene valutare, oltre al tempo medio, anche il tempo "massimo" di ripartenza il quale, in determinate situazioni, potrebbe risultare critico e quindi "non accettabile" per l'operatività del sistema.			

Aderenza:

Una "metrica esterna dell'aderenza" misura il numero di funzioni che risultano essere non conformi, o che generano problemi di aderenza a standard, convenzioni e regolamenti, leggi, ecc. relative all'affidabilità e che debbano essere seguite.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alla recuperabilità	Misurare il grado di aderenza del prodotto alle regolamentazioni, normative e standard relative alla recuperabilità	M = 1 – A/B A = Numero di elementi non aderenti alle norme B = Numero totale di elementi che devono risultare aderenti alle norme in materia	- Specifiche, Standard, normative e regolamentazioni in materia - Specifiche di test - Rapporto esito test

4.6.3 Usabilità

Le "metriche esterne dell'usabilità" devono poter misurare la facilità di comprendere, apprendere ed operare con il prodotto. Inoltre, misura il livello di aderenza ad eventuali regole, standard e linee guida sull'usabilità.

Le metriche relative alla comprensibilità, apprendibilità ed operabilità sono applicate seguendo due diversi metodi: "test utente" o "test del prodotto in uso".

Test utente

Alcune metriche esterne dell'usabilità sono applicate all'utilizzo delle funzioni da parte degli utenti. Queste misure possono variare, anche sostanzialmente, con il variare degli utenti. I risultati possono essere influenzati dalle capacità degli utenti e dalle caratteristiche del sistema nel quale si esegue il prodotto. Ciò non costituisce un problema in quanto l'usabilità è una misura soggettiva. Importante è che gli utenti coinvolti siano "rappresentativi" di quelli reali e gli scenari eseguiti nell'utilizzo del prodotto siano molto simili, se non proprio gli stessi, a quelli reali. Risultati ragionevoli sono ottenuti coinvolgendo almeno 5 (cinque) utenti diversi. Le attività devono essere svolte dagli utenti senza alcun supporto esterno che possa facilitare l'utilizzo delle funzionai o risolvere problemi operativi. Ovviamente, gli utenti sono istruiti in anticipo sull'utilizzo delle funzioni del prodotto.

Occorre stabilire criteri di accettazione delle misure eseguite (valori di soglia), oppure paragonare i risultati con altre misure disponibili (per esempio, relative ad un prodotto simile preso come riferimento). In questi casi si calcolano, si confrontano e si valutano sia il valore medio delle misure sia la deviazione standard media.

Molte di queste misure possono essere eseguite su di un prototipo, se esiste, anticipando così eventuali problemi di usabilità. La scelta delle metriche da adoperare dipende dalle caratteristiche di usabilità richieste al prodotto finale in base al suo utilizzo in determinati contesti.

Test del prodotto in uso

Alcune metriche di usabilità osservano l'utilizzo di una sequenza di più funzioni particolari per completare determinati compiti ben precisi (task.) legati all'uso quotidiano del prodotto nell'ambiente di lavoro. Questa tecnica ha il vantaggio di richiedere un numero minore di utenti coinvolti nei test. Lo svantaggio è che non tutte le funzioni potrebbero essere valutate in termini di usabilità con lo stesso livello di approfondimento.

Per ciascuna sottocaratteristica di usabilità, si possono applicare metriche specifiche.

Comprensibilità

Una "metrica esterna della comprensibilità" misura quanto gli utenti sono in grado di comprendere:

a) se il prodotto fornisce le funzionalità richieste;

b) come possono essere utilizzate le funzionalità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Completezza della descrizione funzionale	Misurare la percentuale di funzioni comprese dall'utente dopo aver letto la descrizione del prodotto (es. Manuale utente, Specifiche funzionali)	M = A/B A = Numero di funzioni comprese B = Numero totale di funzioni disponibili	Rapporto esito test utente Manuale utente e/o Specifiche funzionali
2. Accessibilità delle dimostrazioni	Misurare la percentuale di demo/tutorial disponibile agli utenti. Può essere misurata sia in fase di revisione statica della documentazione che in fase di reale utilizzo del prodotto	M = A/B A = Numero di demo/tutorial disponibili agli utenti B = Numero totale di demo/tutorial realizzate	Rapporto esito test utente
3. Efficacia delle dimostrazioni	Misura la percentuale di demo/tutorial seguite con successo	M = A/B A = Numero di demo/tutorial seguite con successo B = Numero totale di demo/tutorial seguite	Rapporto esito test utente
4. Evidenza delle funzioni	Misura se l'utente è in grado di riconoscere le funzioni presenti sulle interfacce del prodotto	M = A/B A = Numero di funzioni riconosciute B = Numero totale di funzioni disponibili	Rapporto esito test utente
5. Comprensione delle funzioni disponibili	Misura se l'utente è in grado di comprendere come utilizzare le funzioni presenti sulle interfacce del prodotto	M = A/B A = Numero di funzioni comprese dall'utente B = Numero totale di funzioni presenti	Rapporto esito test utente
6. Comprensione degli input/output	Misura se l'utente è in grado di comprendere quali input siano richiesti e quali output siano prodotti	M = A/B A = Numero di input e output compresi dall'utente B = Numero totale di input e output che l'utente verifica	Rapporto esito test utente

Apprendibilità

Una "metrica esterna dell'apprendibilità" misura la facilità ed il tempo necessario agli utenti per imparare ad utilizzare le funzioni del prodotto, e misura anche l'efficacia dell'aiuto in linea (help on-line).

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Tempo di apprendimento	Misura il tempo necessario ad un utente per comprendere l'utilizzo corretto di una funzione	T = Tempo di apprendimento di una funzione	Rapporto esito test utente
2. Facilità di apprendimento all'uso di un task	Misura il tempo complessivo necessario ad un utente per imparare a svolgere un task correttamente ed in maniera efficiente (vedi Nota)	M = Sum(T) T = Tempo speso a svolgere il task in un singolo tentativo	Rapporto esito test utente
3. Accessibilità dell'aiuto in linea	Misura la capacità dell'utente ad accedere con successo l'aiuto in linea durante lo svolgimento di un task	M = A/B A = Numero di volte in cui l'utente accede con successo all'aiuto in linea B = Numero totale di tentativi di accesso all'aiuto in linea	Rapporto esito test utente
4. Efficacia della documentazione e dell'aiuto in linea	Misura la capacità dell'utente ad utilizzare con successo la documentazione e l'aiuto in linea durante lo svolgimento di un task	M = A/B A = Numero di funzioni completate con successo accedendo alla documentazione ed all'aiuto in linea B = Numero totale di funzioni eseguite accedendo alla documentazione ed all'aiuto in linea	Rapporto esito test utente
Nota: Per imparare a svolgere un compito (task) in maniera efficiente, l'utente ripete lo svolgimento del task più volte finché non impara a svolgerlo nel modo migliore.			

Operabilità

Una "metrica esterna dell'operabilità" misurare se gli utenti sono in grado di operare con il prodotto e tenerlo sotto controllo. Le metriche di questo tipo appartengono a categorie che seguono i principi dei dialoghi definiti nelle norme ISO 9241-10:

- adeguatezza del software a completare i compiti (task) richiesti;
- auto-descrizione del software;
- controllabilità del software;
- conformità del software alle aspettative degli utenti;
- tolleranza ai guati;
- adeguatezza del software alla personalizzazione.

La scelta delle funzioni da testare dipende dalla loro frequenza di utilizzo, dalla criticità per il business, da eventuali problemi di usabilità evidenziati.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Consistenza dell'operatività nell'uso del prodotto	Misurare la facilità con cui l'utente comunica al sistema cosa vuole fare e riceve come risultato dal sistema cosa si aspetta	M = A/B[6] A = Numero di interazioni svolte come l'utente si aspetta che avvengano B = Numero totale di interazioni svolte	Report esito test operativo (di usabilità)
2. Correzione degli errori operativi	Misurare il tempo speso a correggere un errore operativo commesso dall'utente nello svolgere un task	M = TER/TOT TER = Tempo speso per correggere gli errori TOT = Tempo totale per completare il task	Report esito test operativo (di usabilità)
	Misurare la facilità con cui l'utente riesce a risolvere gli errori commessi durante l'esecuzione di un task	M = A/B A = Numero di task in cui l'utente è stato in grado di risolvere gli errori commessi B = Numero totale di task eseguiti con rilevamento di errori	Report esito test operativo (di usabilità)

[6] La metrica relativa a quanto realmente avviene rispetto a quanto ci si attende che avvenga è definita in letteratura come: WYSIWYG = What You See Is What You Get, letteralemente "Ciò che vedi è ciò che ottieni".

3. Disponibilità di valori di base predefiniti (default)	Misurare la facilità dell'utente nel selezionare valori dei parametri nell'utilizzo delle funzioni	M = 1- A/B A = Numero di volte in cui l'utente fallisce nel selezionare i valori dei parametri B = Numero totale di volte in cui l'utente deve selezionare valori dei parametri	Report esito test operativo (di usabilità)
4. Comprensibilità dei messaggi nell'uso del prodotto	Misura la facilità con cui l'utente è in grado di comprendere i messaggi emessi dal sistema e di proseguire nelle operazioni	M = 1 - A/B A = Tempo speso dall'utente per comprendere i messaggi (e per ripetere un'operazione a causa di un messaggio non ben interpretato) B = Tempo totale di osservazione dell'operazione	Report esito test operativo (di usabilità)
5. Autoesplicazione dei messaggi nell'uso del prodotto	Misura la percentuale di condizioni di errore che l'utente è in grado di risolvere seguendo le indicazioni fornite dai messaggi	M = A/B A = Numero di volte in cui l'utente risolve una condizione di errore con l'aiuto dei messaggi B = Numero totale di condizioni di errore testate	Report esito test operativo (di usabilità)
6. Recupero delle condizioni di errore nell'uso del prodotto	Misura la percentuale di condizioni di errore che l'utente è in grado di risolvere da solo in quanto il sistema non emette alcun messaggio a proposito	M = A/B A = Numero di volte in cui l'utente risolve una condizione di errore da solo B = Numero totale di condizioni di errore testate	Report esito test operativo (di usabilità)
7. Tempo utile di utilizzo del prodotto tra due errori consecutivi commessi dall'utente	Misurare il tempo medio di utilizzo continuativo del prodotto tra due errori consecutivi commessi dall'utente	M = tempo medio = Sum(T)/N T = Tempo di utilizzo continuativo del prodotto tra due errori consecutivi commessi dall'utente N = Numero di intervalli operativi misurati	Report esito test operativo (di usabilità)

8. Possibilità di tornare indietro da operazioni eseguite	Misurare la frequenza con cui l'utente è in grado di correggere errori negli input forniti	M = A/B A = Numero di volte in cui la correzione dell'input ha successo B = Numero totale di correzioni eseguite	Report esito test operativo (di usabilità)
	Misurare il numero di campi di input che l'utente è in gradi di correggere con successo a fronte di errori commessi	M = A/B N = Numero di input corretti con successo dall'utente B = Numero totale di input corretti	Report esito test operativo (di usabilità)
9. Personalizzazione	Misurare la possibilità dell'utente di personalizzare le procedure a seconda delle proprie necessità	M = A/B A = Numero di funzioni che permettono la personalizzazione come richiesto B = Numero totale di funzioni per cui è richiesta la personalizzazione	Report esito test operativo (di usabilità)
	Misurare la correttezza delle personalizzazioni consentite ed eseguite	M = A/B A = Numero di personalizzazioni eseguite con successo B = Numero totale di personalizzazioni eseguite	Report esito test operativo (di usabilità)
10. Riduzione delle procedure operative	Misura la riduzione delle operazioni dell'utente nell'utilizzo di una funzione. La misura paragona il prodotto attuale con una sua versione precedente o con un altro prodotto.	M = 1 – A/B A = Numero di funzioni in cui si è ridotto il numero di operazioni richieste B = Numero totale di funzioni eseguite	Report esito test operativo (di usabilità)
11. Accesso fisico alle funzioni da parte di personale portatore di handicap	Misura il numero di funzioni disponili dal prodotto ed accessibili da parte di portatori di handicap	M = A/B A = Numero di funzioni accessibili B = Numero totale di funzioni disponibili	Report esito test operativo (di usabilità)

Attrattività

Una "metrica esterna dell'attrattività" misura "come appare" il software; essa può essere influenzata dalle scelte grafiche e progettuali delle interfacce utente (colori, immagini, formato dei testi, ecc.). Queste caratteristiche sono particolarmente importanti per il prodotto software di consumo.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Attrattività delle interfacce	Misurare quanto attrattiva risulti un'interfaccia all'utente	M = Sum(Q)/N Q = questionario riempito da un utente che utilizza l'interfaccia N = numero di utenti. Il valore medio è calcolato sul risultato di almeno 5 utenti.	Questionario
2. Possibilità di personalizzare le interfacce	Misurare il grado di personalizzazione che un utente può fare di un'interfaccia	M = A/B A = numero di interfacce che l'utente può personalizzare B = numero di interfacce che l'utente vorrebbe poter personalizzare	Rapporto esito test di operatività (o di usabilità)

Aderenza all'usabilità

Una "metrica esterna dell'aderenza" misura il grado di aderenza a standard, convenzioni e regolamenti, leggi, ecc. relative all'usabilità.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alle norme i usabilità	Misurare il livello di aderenza a standard, normative e regolamentazioni applicabili all'usabilità	M = 1 - A/B A = numero di elementi che risultano non aderenti B = numero totale di elementi che devono risultare aderenti	Rapporto esito revisione tecnica (di usabilità)

4.6.4 Efficienza

Le "metriche esterne dell'efficienza" devono poter misurare le prestazioni del software in termini di comportamento rispetto al tempo (tempo di risposta di un enquiry, tempo di completamento di una transazione, ecc.), di utilizzo di risorse (memoria occupata, spazio disco necessario, ecc.) ed aderenza a standard stabiliti al riguardo.

In particolare, per ciascuna sottocaratteristica, si possono applicare metriche specifiche.

Comportamento rispetto al tempo

Una "metrica esterna del comportamento del prodotto rispetto al tempo" misura:

a) il tempo medio necessario al software per eseguire una funzione (esempio, tempo medio di risposta di una transazione);

b) quantità di lavoro (throughput) eseguita nell'unità di tempo (esempio, numero di record elaborati, numero di righe di un tabulato stampate).

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Tempo di completamento di un task	Misurare il tempo necessario a completare un task in determinate condizioni di carico del sistema	M = T T = tempo necessario al sistema per completare il task (vedi Nota 1)	Rapporto esito test
2. Quantità di lavoro svolto in un intervallo di tempo	Misurare la quantità di lavoro (numero di transazioni) eseguito in un determinato intervallo di tempo	M = A/B A = numero di transazioni eseguite nell'intervallo di tempo B = intervallo di tempo	Rapporto esito test
3. Transazioni concorrenti svolte in un intervallo di tempo	Misurare il numero di transazioni concorrenti che si possono svolgere nell'intervallo di tempo stabilito (Vedi Nota 2)	M = A/B A = numero di transazioni eseguite nell'intervallo di tempo B = intervallo di tempo	Rapporto esito test
4. Tempo di risposta	Misurare il tempo medio con cui il sistema risponde ad un comando immesso dall'utente (vedi Nota 3)	M = tempo che intercorre tra l'immissione del comando da parte dell'operatore e la presentazione della risposta del sistema	Rapporto esito test

Nota 1: Per task particolarmente critici oppure usati molto di frequente dagli utenti, si effettuano molte misure e si calcolano il "valore medio" dei tempi di risposta misurati ed il valore "peggiore" (tempo massimo).

Nota 2: Per effettuare misure relative ad operazioni concorrenti si utilizza un ambiente con più utenti che operano in contemporanea, oppure si utilizzano strumenti particolari di simulazione.

Nota 3: Per transazioni particolarmente critiche oppure usate molto di frequente dagli utenti si calcola il valore medio di più prove eseguite per una stessa transazione ed il valore peggiore registrato.

Utilizzo delle risorse

Una "metrica esterna dell'utilizzo di risorse" misura quante risorse sono necessarie per eseguire le funzioni richieste. Si misura:

a) l'utilizzo di risorse di input e output in termini di:

- numero di risorse di input/output utilizzate,
- numero di errori causati dal loro utilizzo,
- limiti di carico sopportati dalle unità.

b) l'utilizzo di risorse di memoria in termini di:

- quantità media e massima di memoria utilizzata,
- numero di errori causati dall'utilizzo di memoria,
- in assoluto e rispetto ad un intervallo di tempo.

c) l'utilizzo di risorse di trasmissione in termini di:

- utilizzo medio e massimo delle risorse trasmissive,
- numero di errori rilevati durante le operazioni di trasmissione, in assoluto e rispetto ad un intervallo di tempo;
- bilanciamento delle risorse utilizzate.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Utilizzo I/O	Misurare il livello di utilizzo di I/O per determinare eventuali inefficienze	M = A/B A = tempo di utilizzo dell'I/O misurato B = tempo di utilizzo di I/O previsto dal design	- Rapporto esito test - Specifiche - Design
2. Correttezza dell'utilizzo dell'I/O	Misurare il numero di errori rilevati durante l'utilizzo dell'I/O da parte del prodotto 1) in un determinato intervallo di tempo; 2) rispetto ad un numero di task eseguiti	M = 1 – A/B A = numero di errori rilevati B = intervallo di tempo di misurazione (1), oppure numero di task eseguiti (2)	- Rapporto esito test

3. Tempo di attesa per l'I/O	Misurare il tempo di attesa per il completamento di un'operazione di I/O	M = T = tempo speso dal sistema per completare un'operazione di I/O. E' opportuno misurare il tempo medio ripetendo più volte il task e tener conto anche del valore "massimo" rilevato per task particolarmente critici o molto frequenti	- Rapporto esito test
4. Carico massimo di operazioni di I/O	Misurare il limite massimo di operazioni di I/O che il sistema è in grado di supportare durante l'esecuzione di un determinato task (mantenendo le prestazioni richieste)	M = Max Max = valore massimo di operazioni di I/O che il sistema è in grado di supportare. Le misure sono effettuate utilizzando opportuni strumenti di simulazione e misurazione dei carichi.	- Rapporto esito test - Rapporto strumento di simulazione
5. Utilizzo di memoria	Misurare: 1) la quantità di memoria utilizzata nell'esecuzione di uno o più task; 2) il limite massimo di memoria utilizzata prima che il sistema vada in errore per insufficienza di memoria	1) M = A/B A = dimensione della memoria utilizzata da uno o più task; 2) M = Max = limite massimo di memoria utilizzabile	- Rapporto esito test - Rapporto strumento di simulazione
6. Correttezza dell'utilizzo della memoria	Misurare il numero di errori rilevati a causa dell'utilizzo della memoria con un determinato carico di lavoro: 1) in un determinato intervallo di tempo; 2) rispetto ad un numero di task eseguiti	M = 1 – A/B A = numero di errori rilevati B = intervallo di tempo di misurazione (1), B = numero di task eseguiti (2)	- Rapporto esito test
7. Utilizzo della rete	Misurare: 1) l'efficienza delle trasmissioni durante l'esecuzione di uno o più task; 2) la massima capacità di trasmissione. Le misure sono effettuate utilizzando strumenti di simulazione e monitoraggio.	1) M = A/B A = capacità di trasmissione misurata B = capacità di trasmissione prevista dal disegno 2) M = Max = valore massimo di capacità trasmissiva.	- Rapporto esito test - Design - Rapporto strumento di simulazione e monitoraggio delle trasmissioni

8. Correttezza dell'utilizzo della rete di trasmissione	Misurare il numero di errori rilevati durante le operazioni di trasmissione in un determinato periodo di tempo e con un determinato carico di lavoro	M = 1 - A/B A = numero di errori di trasmissione rilevati B = durata dell'intervallo di tempo	- Rapporto esito test

Aderenza all'efficienza

Una "metrica esterna dell'efficienza" misura il grado di aderenza a standard, convenzioni e regolamenti, leggi, ecc. applicabili.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza all'efficienza	Misurare l'aderenza a standard, regolamentazione e normative applicabili all'efficienza	M = 1 - A/B A = numero di elementi che risultano non aderenti B = numero totale di elementi che devono risultare aderenti	Rapporto esito revisioni tecniche

4.6.5 Manutenibilità

Una "metrica esterna della manutenibilità" deve poter misurare attributi relativi al comportamento dei manutentori (le persone responsabili della manutenzione), degli utenti e del software stesso quando il prodotto è in manutenzione o modificato durante i test o le attività di manutenzione.

In particolare, per ciascuna sottocaratteristica, si possono applicare metriche specifiche.

Analizzabilità

Una "metrica esterna dell'analizzabilità" misura la quantità di tempo e le risorse necessarie ai manutentori ed agli utenti quanto sia necessario diagnosticare un problema, identificare le parti coinvolte in una modifica.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Supporto alla funzione di diagnosi	Misurare quanto le funzioni di diagnosi disponibili siano in grado di supportare la manutenzione; ovvero misurare quanto gli utenti o il personale addetto alla manutenzione siano in grado di diagnosticare le cause dei problemi riscontrati	M = A/B A = numero di volte in cui l'utente o il manutentore è in grado di diagnosticare la causa del problema B = numero di condizioni di errori riscontrate	- Rapporto risoluzione problemi - Rapporto esito operazioni
2. Registrazione dei dati di errore	Misurare la capacità di registrazione del sistema delle informazioni relative alle condizioni di errore	M = A/B A = numero di informazioni registrate B = numero di informazioni di registrazione previste dal design	- Rapporto risoluzione problemi - Design
3. Efficacia dell'analisi dei problemi	Misurare quanto sia facile per l'utente o per il manutentore effettuare una corretta analisi dei problemi	M = A/B A = numero di analisi condotte con successo e nei tempi determinati B = numero di analisi svolte	- Rapporto risoluzione problemi - Design/Specifiche
4. Efficacia della ricerca della causa di un problema	Misurare quanto sia facile per l'utente o per il manutentore risalire alla causa di un problema	M = A/B A = numero di volte in cui si trova la causa di un problema nei tempi determinati B = numero di analisi svolte	- Rapporto risoluzione problemi - Design/Specifiche

Modificabilità

Una "metrica esterna della modificabilità" misura la quantità di tempo e le risorse necessarie ai manutentori ed agli utenti quanto sia necessario eseguire le modifiche specificate.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Capacità di controllo delle modifiche del software	Misurare quanto sia facile per l'utente identificare una versione del software modificata. Misurare quanto sia facile per il manutentore controllare le modifiche del software eseguite per risolvere un problema	M = A/B A = numero di modifiche registrate nel sistema B = numero di modifiche apportate	- Rapporto risoluzione problemi - Registro modifiche (Change Log)
2. Parametrizzazione delle modifiche al software	Misurare la possibilità per un utente o per un manutentore di cambiare alcuni parametri del software per risolvere un problema	M = 1 – A/B A = numero di volte in cui la modifica dei parametri non ha permesso di risolvere il problema B = numero di volte in cui si è tentato di cambiare i parametri	- Rapporto risoluzione problemi
3. Facilità di modifica del software	Misurare quanto sia facile per un manutentore modificar il software	M = Sum(A/B) / N A = tempo speso per modificare il software (in ore) B = dimensione del software modificato (in Klocs o FP) N = numero di modifiche effettuate	- Rapporto risoluzione problemi
4. Efficienza del ciclo di manutenzione	Misurare quanto sia facile per un manutentore risolvere i problemi in tempi ragionevoli con soddisfazione degli utenti	M = Sum(T)/N T = tempo effettivo di realizzazione di una modifica (vedi Nota 1) N = numero di modifiche effettuate	- Rapporto di manutenzione
Nota 1: Il tempo effettivo di modifica è misurato dal momento in cui il manutentore riceve la notifica del problema e lo accetta come problema reale al momento in cui l'utente riceve la modifica con la soluzione del problema.			

Stabilità

Una "metrica esterna della stabilità" misura ogni tipo di comportamento del prodotto software "non previsto" a seguito di un test o di una modifica eseguita.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Efficacia delle modifiche	Misurare se l'utente può utilizzare il prodotto dopo le modifiche senza rilevare ulteriori errori. Misurare se il manutentore può modificare facilmente il software riducendo/annullando i possibili effetti collaterali negativi procurati dalle modifiche apportate.	M = 1 - A/B A = numero di modifiche che generano ulteriori errori B = numero totale di modifiche effettuate	Rapporto di manutenzione

Provabilità (testabilità)

Una "metrica esterna della provabilità" misura la quantità di tempo e le risorse necessarie ai manutentori ed agli unteti quanto sia necessario eseguire dei test sul software modificato oppure no (esempio, test di regressione).

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Efficacia del re-testing	Misurare la facilità con cui l'utente o il manutentore effettua il collaudo di una modifica al software	M = Sum(T)/N T = tempo speso in ore per effettuare il collaudo delle modifiche N = numero di modifiche effettuate	Rapporto risoluzione problemi
2. Disponibilità di ambiente di test	Misurare se sia possibile effettuare test esaustivi di una modifica senza dover aggiungere elementi all'ambiente di test. Per elementi di test si intendono: casi di test, strumenti di test, base dati di prova, ecc.	M = A/B A = numero di casi in cui si effettuano i test senza l'aggiunta di ulteriori elementi B = numero di test eseguiti	Rapporto risoluzione problemi
3. Ripartenza dei test	Misurare la capacità di fermare un test in esecuzione e farlo ripartire dal punto in cui si è interrotto senza incominciare da capo. Si tratta di test progettati per essere eseguiti in passi successivi, ciascuno con registrazione del passo appena completato e possibilità di ripartenza dal passo successivo.	M = A/B A = numero di test che permettono l'interruzione e la ripartenza B = numero totale di test eseguiti	Rapporto risoluzione problemi

Aderenza alla manutenibilità

Una "metrica esterna della manutenibilità" misura il grado di aderenza a standard, convenzioni e regolamenti applicabili alla manutenzione.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alla manutenibilità	Misurare il livello di aderenza a standard, regolamentazione e normative sulla manutenibilità	M = 1 – A/B A = numero di elementi che risultano aderenti B = numero totale di elementi che devono risultare aderenti	- Specifiche di aderenza - Specifiche di test - Rapporto esito test

4.6.6 Portabilità

Le "metriche esterne della portabilità" devono poter misurare il comportamento del sistema o dell'operatore durante le attività di porting.

Adattabilità

Una "metrica esterna dell'adattabilità" misura il lavoro richiesto ad un operatore che debba adattare il software ad un ambiente diverso da quello originario. Se l'adattamento al nuovo ambiente richiede nuove procedure, le metriche devono poter misurare la quantità di lavoro (*effort*) necessario a tale attività.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Adattabilità della struttura dei dati	Misurare quanto sia facile per l'utente o per il manutentore adattare il prodotto ad un nuovo ambiente senza dover modificare la struttura dei dati esistente	M = 1 - A/B A = numero di strutture dati modificate o che presentano limitazioni B = numero di strutture dati che devono risultare invariate secondo il disegno	- Rapporto risoluzione problemi - Specifiche e disegno
2. Adattabilità all'ambiente operativo	Misurare quanto sia facile per l'utente o per il manutentore adattare il prodotto ad un nuovo ambiente operativo	M = 1 - A/B A = numero di elementi operativi modificati o che presentano limitazioni B = numero di elementi operativi che devono risultare invariati secondo il disegno	- Rapporto risoluzione problemi - Specifiche e disegno
3. Adattabilità all'ambiente hardware	Misurare quanto sia facile per l'utente o per il manutentore adattare il prodotto ad un nuovo ambiente hardware	M = 1 - A/B A = numero di funzioni non completate nel nuovo ambiente B = numero di funzioni che devono risultare invariate secondo il disegno	- Rapporto risoluzione problemi - Specifiche e disegno
4. Adattabilità all'ambiente software	Misurare quanto sia facile per l'utente o per il manutentore adattare il prodotto ad un nuovo ambiente software	M = 1 - A/B A = numero di funzioni non completate nel nuovo ambiente B = numero di funzioni che devono risultare invariate secondo il disegno	- Rapporto risoluzione problemi - Specifiche e disegno

5. Facilità di porting	Misurare quanto sia facile per l'utente e per il manutentore completare con successo il porting del prodotto nel nuovo ambiente	M = T T = tempo speso per installare, personalizzare e modificare il prodotto	- Rapporto risoluzione problemi - Rapporto esito installazione

Installabilità

Una "metrica esterna dell'installabilità" misura il lavoro richiesto ad un operatore che debba installare il software in un determinato ambiente.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Facilità di installazione	Misurare la facilità con cui l'utente o il manutentore installa e personalizza il prodotto nel nuovo ambiente senza dover ripetere più volte l'installazione e la personalizzazione a seguito del verificarsi di problemi	M = 1 – A/B A = numero di volte in cui fallisce l'operazione di l'installazione o personalizzazione B = numero di volte in cui si tenta di installare o personalizzare il prodotto nel nuovo ambiente	- Rapporto installazione - Rapporto risoluzione errori
2. Efficienza dell'installazione	Misurare il tempo necessario ad installare e personalizzare il prodotto nel nuovo ambiente	M = A/B A = numero di ore impiegate per installare e personalizzare il prodotto nel nuovo ambiente B = tempo previsto dal design e specifiche	- Rapporto installazione - Design, specifiche

Coesistenza

Una "metrica esterna della coesistenza" misura il comportamento di un operatore (quantità di lavoro, difficoltà) che debba utilizzare il software insieme ad altri prodotti con i quali condivide risorse.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Possibilità di coesistenza con altri prodotti	Misurare se l'utente possa utilizzare il prodotto senza limitazioni o errori anche in presenza di altri prodotti nello stesso ambiente	M = A/B A = numero di errori o limitazioni riscontrate B = tempo totale in cui si è operato concorrentemente con altri prodotti nello stesso ambiente	- Rapporto esito test - Rapporto risoluzione problemi

Sostituibilità

Una "metrica esterna della sostituilità" misura il comportamento di un utente (quantità di lavoro aggiuntivo, difficoltà) che debba utilizzare il software al posto di un altro prodotto in un determinato ambiente.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Continuità dei dati	Misurare se l'utente o il manutentore possa continuare ad utilizzare i dati esistenti anche dopo aver sostituito l'attuale software con il nuovo prodotto	M = 1 – A/B A = numero di errori rilevati nell'utilizzo dei dati esistenti B = numero di dati che devono essere ancora utilizzabile secondo il design e le specifiche	- Rapporto risoluzione problemi - Design specifiche
2. Efficacia della migrazione dei dati	Misurare se l'utente o il manutentore possa utilizzare i dati dopo il completamento della migrazione prevista	M = 1 – A/B A = numero di dati errati riscontrati dopo la migrazione B = numero di dati migrati	- Rapporto risoluzione problemi - Rapporto esito migrazione
3. Continuità funzionale	Misurare se l'utente o il manutentore possa utilizzare le funzioni precedenti anche con il nuovo prodotto	M = A/B A = numero di funzioni che producono un risultato simile a quello precedente B = numero di funzioni testate e che devono continuare a valere secondo il disegno e le specifiche	- Rapporto esito test - Rapporto risoluzione problemi - Design specifiche

Aderenza alla portabilità

Una "metrica esterna della portabilità" misura il numero di funzioni che presentano problemi di aderenza a standard, convenzioni e regolamenti di portabilità, ed il numero di volte che tali problemi si presentano.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Aderenza alla portabilità	Misurare il livello di aderenza a standard, regolamentazione e normative sulla portabilità	M = 1 – A/B A = numero di elementi che risultano aderenti B = numero totale di elementi che devono risultare aderenti	- Specifiche di aderenza - Specifiche di test - Rapporto esito test

4.7 Esempi di metriche in uso

Le "metriche della qualità in us"o misurano il grado con cui il prodotto software soddisfa le necessità specificate degli utenti nel loro reale contesto operativo. Il livello raggiungimento degli obiettivi da parte degli utenti è valutato in relazione all'efficacia, alla produttività, alla sicurezza ed al grado di soddisfazione. Tali metriche si adoperano solo negli ambienti reali di produzione.

4.7.1 Efficacia

Le "metriche in uso dell'efficacia" valutano se le attività specifiche (task) eseguite dagli utenti raggiungono gli obiettivi prefissati con cura e completezza in uno specifico contesto di utilizzo. Esse non prendono in considerazione "come" tali obiettivi sono raggiunti, ma solo se essi sono raggiunti.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Frequenza di errori nel completare un task	Misurare il numero di errori rilevati dall'utente nel completare un determinato task	M = media = A/B A = numero di errori rilevati da un singolo utente B = numero di utenti che hanno ripetuto l'operazione (almeno 5)	Rapporto esito test
2. Numero di obiettivi raggiunti	Misurare il numero di obiettivi raggiunti da un utente nell'eseguire uno o più task	M = media = A/B A = numero di obiettivi raggiunti da un singolo utente B = numero di utenti che hanno ripetuto l'operazione (almeno 5)	Rapporto esito test
3. Numero di task eseguiti senza interruzione	Misurare il numero di task che un utente può eseguire senza interruzione, cioè senza che si verifichi un malfunzionamento che gli impedisca di proseguire il lavoro	M = media = A/B A = numero di task eseguiti da un singolo utente B = numero di utenti che hanno ripetuto l'operazione (almeno 5)	Rapporto esito test

4.7.2 Produttività

Le "metriche in uso della produttività" valutano le risorse utilizzate dagli utenti in relazione all'efficacia raggiunta in uno specifico contesto di utilizzo. Le risorse più comuni prese in considerazione sono il tempo necessario a completare un task; altre risorse includono l'impegno (*effort*) degli utenti, i materiali o il costo finanziario del loro utilizzo.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Produttività media di un utente	Misurare il valore medio della produttività di un utente in un determinato intervallo di tempo (vedi Nota 1)	M = media = A/B A = numero di operazioni eseguite da un singolo utente per completare un compito B = tempo speso da un singolo utente per completare le operazioni	Rapporto esito test
Nota 1: Il valore medio della produttività è calcolato ripetendo le misure con almeno 5 utenti. Il tempo impiegato per completare le operazioni include i tempi morti, spesi dall'utente per capire cosa fare, interpretare i messaggi, consultare la documentazione o l'aiuto in linea, risolvere possibili problemi, ecc.			

4.7.3 Sicurezza

Le "metriche in uso della sicurezza" valutano il livello di rischio relativo ai danni a persone, business, software, proprietà o ambienti in uno specifico contesto di utilizzo. Includono la salute e la sicurezza fisica di chi utilizza il sistema e di chi è coinvolto nell'utilizzo, così come pure le conseguenze fisiche ed economiche.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Livello di sicurezza	Misurare il numero di incidenti di sicurezza causati dall'utilizzo del prodotto in un determinato intervallo di tempo	M = A/B A = numero di incidenti di sicurezza rilevati B = intervallo di tempo monitorato	- Rapporto esito test - Registrazioni incidenti di sicurezza (Security Log)
2. Danni subiti da problemi di sicurezza	Misurare il valore del danno economico causato dai problemi di sicurezza	M = Sum(D) D = danno economico a persone, materiale e di immagine, incluso il costo per la risoluzione dei problemi	- Rapporto esito test - Rapporti audit di sicurezza - Rapporti vari

4.7.4 Soddisfazione

Le "metriche in uso della soddisfazione" valutano l'attitudine degli utenti ad utilizzare il prodotto software in un determinato contesto d'uso.

Nota: La soddisfazione è influenzata dalla percezione che gli utenti hanno delle proprietà del prodotto software (come per esempio quelle misurate dalle metriche esterne) e dalla percezione dell'efficacia, della produttività e della sicurezza.

Esempio:

Metrica	Scopo	Metodo di calcolo	Origine dei dati
1. Livello di soddisfazione	Misurare il livello di soddisfazione degli utenti che utilizzano il prodotto	M = media(Q) Q = questionario compilato da almeno 5 utenti del prodotto. Valutazione espressa su di una scala a 4 valori (vedi Nota 1)	Questionario
Nota 1: Il valore della soddisfazione è espressa dagli utenti sul questionario utilizzando una scala a 4 valori: Molto soddisfatto, Soddisfatto, Poco soddisfatto, Insoddisfatto. Il questionario è predisposto in precedenza con una serie di domande attinenti il prodotto ed il suo utilizzo. Il questionario compilato dagli utenti è sempre anonimo.			

Bibliografia

Di seguito è riportata la bibliografia di base cui l'autore fa riferimento per le sue attività. Essa è distinta per argomento: Qualità, ISO9000, CMMI, Sviluppo professionale, Ingegneria del software ecc. Essa non pretende di essere esaustiva né definitiva; rappresenta invece una buona base di partenza per approfondire i temi trattati nel libro.

Qualità

[LEGA06] Riccardo Lega - 2006
La Patente Europea della Qualità (EQDL)
FrancoAngeli

[LEONARDI00] Erika Leonardi - 2000
Capire la qualità – ISO9000 – Tutto quello che occorre capire per applicare con profitto le nuove norme
Il Sole 24 Ore

[BARBARINO98] Filippo C. Barbarino, Erika Leonardi - 1998
ISO9000 Sistema Qualità e Certificazione
Il Sole 24 Ore

[PINDER96] Mark Pinder & Stuart McAdam - 1996
La consulenza interna
Jackson Libri

[MUNRO94] Lesly Munro-Faure, Malcom Munro-Faure - 1994
Qualità totale: tecniche di attuazione
Jackson Libri

[PALA94] Giorgio Pala - 1994
La qualità. Perché, per chi e come farla
Franco Angeli

[BANCI93] Alessandro Banci, Giuseppe Iacono - 1993
La qualità nei progetti software
Franco Angeli

[NOCENTINI93] Stefano Nocentini - 1993
Il sistema di qualità del software
ETASLIBRI

[WHEELER93] Donald J. Wheeler - 1993
Understanding Variation – The Key To Managing Chaos
SPC Press

[ZEITHAML91] Valerie A. Zeithaml, A. Parasuraman, Leonard L. Berry - 1991
Servire qualità
McGraw-Hill

[RUMMLER90] Geary A. Rummler and Alan P. Brache - 1990
Improving performance
Jossey-Bass Publishers

[CROSBY86] Philip B. Crosby - 1986
La qualità non costa
McGraw-Hill

ISO9000

[ISO9001:2000] ISO/IEC 9001:2000
Quality Management Systems – Requirements

[ISO9000:2000] ISO/IEC 9000:2000
Quality Management Systems – Fundamentals and Vocabulary

[ISO9004:2000] ISO/IEC 9004:2000
Quality Management Systems – Guidelines for performance improvement

[ISO9003:2004] ISO/IEC 9003:2004
Software and Systems Engineering – Guidelines for the application of ISO 9001:2000 to computer software

ISO9126-1:2001] ISO/IEC 9126-1:2001
Software Engineering – Product Quality Part 1: Quality Model

ISO9126-2:2002] ISO/IEC 9126-2:2002
Software Engineering – Product Quality Part 2: External Metrics

ISO9126-3:2002] ISO/IEC 9126-3:2002
Software Engineering – Product Quality Part 3: Internal Metrics

ISO9126-4:2002] ISO/IEC 9126-4:2002
Software Engineering – Product Quality Part 4: Quality In Use Metrics

[ISO12207:95] ISO/IEC 12207:2000
Information Technology – Life Cycle Management – Life Cycle Processes

[ISO12207:02-1] ISO/IEC 12207:1995/Amd.1:2002
Information Technology – Software Lifecycle Processes-Amendment 1

[ISO15271:98] ISO/IEC TR 15271:1998
Information Technology – Guide for ISO/IEC 12207 Software Lifecycle Processes)

[ISO16326:99] ISO/IEC TR 16326:1999
Software Engineering – Guide for the application of ISO/IEC 12207 in Project Management

Il concetto di qualità del software si è evoluto nel tempo incorporando di volta in volta diverse esigenze divenute importanti ai fini del corretto utilizzo e funzionamento del prodotto. La diffusione sempre più larga di software nelle diverse aree applicative, con rilevanza e criticità sempre maggiore per il business e per la sicurezza, pone il problema della sua adeguatezza alle necessità di utilizzo. Sviluppare o selezionare prodotti software di qualità adeguata all'utilizzo che se ne dovrà fare, è quindi un'attività di grande importanza per il business. Per raggiungere l'obiettivo, occorre definire caratteristiche che permettano di valutare in maniera oggettiva e precisa il livello di qualità richiesto al prodotto. Per essere realmente efficaci, tali caratteristiche devono tener conto dello scopo per il quale si utilizza il prodotto. Gli elementi da tener presente nella definizione delle caratteristiche di qualità sono: profilo degli utenti, obiettivi da raggiungere con l'utilizzo del prodotto, modalità operative abitualmente seguite dagli utenti nello svolgimento dei propri compiti. Ogni caratteristica rilevante dovrà essere misurata da metriche adeguate ed accettate.

Il documento descrive il modello per la qualità del software e le relative metriche emesse dall'Organizzazione Internazionale per la Standardizzazione (ISO) con il titolo ISO/IEC 9126-1, 2, 3 - Software Engineering – Part 1, 2, 3: Quality Model, Internal Metrics, External Metrics, Metrics In Use.

Ercole Franco Colonese svolge la sua attività di consulente essenzialmente nell'area dello sviluppo applicativo. La sua esperienza è maturata in moltissimi anni di lavoro presso i laboratori internazionali IBM di sviluppo di prodotti software dove ha ricoperto ruoli tecnici e manageriali. Ha realizzato numerosi sistemi qualità aziendali certificati ISO9000 presso piccole, medie e grandi aziende, pubbliche e private. Ha implementato modelli di eccellenza (EFQM, Malcom Baldridge Six-Sigma, ecc.). Ha condotto diversi progetti di reingegnerizzazione dei processi di sviluppo software in ottica di miglioramento delle performance e della qualità. Ha applicato con successo i modelli di maturità dei processi come SEI-CMM e CMMI. Come docente, tiene corsi sulla qualità del software, le metodologie di sviluppo e l'ingegneria del software presso Clienti ed Università.

e-mail: e.colonese@virgilio.it www.colonese.it