ISSN 1123-3249

DE QUALITATE

SETTEMBRE 1995

RIVISTA ITALIANA DELLA QUALITA'

£ 15.000 • Pubblicazione mensile
Diretta da Roberto Scaramuzza
Tribunale di Roma 5.8.92 N° 479/92
Spedizione Abbon. Post. 50% - Roma

IL GLOBAL DESIGN APPROACH PER LA GESTIONE DELLA QUALITÀ
Frédéric Jallat

QUALITÀ ED INNOVAZIONE: IL RUOLO DEI SISTEMI TERRITORIALI
Maria Silano e Giuseppe Rossetto

VERSO IL ROBUST DESIGN
Sergio Di Veroli

L'ATTIVITÀ NORMATIVA NEL SETTORE DELL'INFORMATICA E DELLE TELECOMUNICAZIONI
Roberto Bottio e Francesco Serracchioli

IL SOFTWARE: STRUMENTO ED OBIETTIVO DELLA QUALITÀ
Luca Matteini

L'APPLICAZIONE DELLE ISO 9000 NELLE AZIENDE DI INFORMATICA
Ioanis Tsiouras

LA MISURA DELL'EFFICIENZA DELL'IMPRESA
Giovanna Prina e Ranieri Tesi

LA NUOVA QUALITÀ
Marco Maiocchi

QUALITÀ DEL SOFTWARE
Alberto De Toni, Renato Granzotto, Stefano Tonchia

NST
NUOVO STUDIO TECNA

SOMMARIO

- **Editoriale,** *William J. Glynn* 6

SEZIONE FORMAZIONE

- **Il global design approach per la gestione della qualità,** *Frédéric Jallat* 7
- **La voce del processo,** *Domenico Lepore e Claudia Franceschelli* 15
- **Qualità ed innovazione: il ruolo dei sistemi territoriali,** *Maria Silano e Giuseppe Rossetto* 21
- **Verso il robust design,** *Sergio Di Veroli* 27

SEZIONE ORIENTAMENTO

- **L'attività normativa nel settore dell'informatica e delle telecomunicazioni,** *Roberto Bottio e Francesco Serracchioli* 35
- **Il software: strumento ed obiettivo della qualità,** *Luca Matteini* 41
- **L'applicazione delle ISO 9000 nelle aziende di informatica,** *Ioanis Tsiouras* 49
- **La misura dell'efficienza dell'impresa,** *Giovanna Prina e Ranieri Tesi,* 55
- **Quel magico legame fra sistema qualità e sistema informativo,** *Gian Paolo Rossi* 61

SEZIONE TESTIMONIANZE

- **La nuova qualità,** *Marco Maiocchi* 67
- **Qualità del software,** *Alberto De Toni, Renato Granzotto, Stefano Tonchia* 75

■ Modelli di misurazione, normative internazionali ed esperienze aziendali ■

QUALITÀ DEL SOFTWARE

Alberto De Toni *, Renato Granzotto ** Stefano Tonchia ***

EVIDENZA

CORSIVO

Molti degli strumenti e delle procedure di Qualità Totale possono essere utilmente applicate anche allo sviluppo del software. Il software è un prodotto complesso che richiede una gestione attenta e formalizzata. L'adozione di una metodologia di ingegneria del software strutturata può fornire le basi per un Sistema Qualità, il quale, a sua volta, assicura che le responsabilità delle procedure siano chiaramente stabilite e che le procedure vengano condotte secondo gli standard richiesti

La norma IEEE-STD-729 [1983] definisce la qualità del software come: «Il grado al quale il software possiede una combinazione desiderata di attributi».

La misurazione della qualità del software

Definire degli obiettivi misurabili nello sviluppo del software è quindi uno degli aspetti più sentiti e richiesti. Infatti, solo attraverso la misurazione è possibile concretizzare degli obiettivi di qualità, tempo e costo e valutare l'adeguatezza di nuove procedure e metodi di sviluppo in relazione ai risultati conseguiti.

Obiettivi misurabili

Fenton [1991] propone un modello per la misurazione del software (Tab.1), con criteri riguardanti le "entità":

Il modello di Fenton per la misurazione della Qualità del software:
- **processi**
- **prodotti**
- **risorse**

1. *processi:* sono i compiti d'ingegneria del software (analisi, progettazione e testing) richiesti per la produzione del software;
2. *prodotti:* sono gli oggetti (compresi la documentazione di progetto, il codice sorgente e la documentazione di test) risultanti dall'attività di ingegnerizzazione del software;
3. *risorse:* sono i componenti (personale, programmi software, elaboratori elettronici e risorse finanziarie) necessari allo svolgimento dei processi e che costituiscono l'input dei processi stessi.

Per ognuna di queste "entità", Fenton illustra gli attributi che possono essere utilizzati per la misurazione; tali attributi sono di due tipi:

* Professore Associato di Organizzazione della Produzione e dei Sistemi Logistici - Facoltà di Ingegneria - Università di Udine.
** Ingegnere gestionale.
*** Dottore di ricerca in Scienza dell'Innovazione Industriale.

Entità	Attributi interni	Attributi esterni
PROCESSI		
Specifiche di costruz.	tempo, impegno, numero di variazioni de requisiti	qualità, stabilità, costo
Progettaz. dettagliata	tempo, impegno, numero di specifiche errate trovate	costo, efficacia
Testing	tempo, impegno, numero di errori rilevati	stabilità, costo, efficacia
PRODOTTI		
Specifiche	dimensione, riusabilità, modularità, ridondanza, funzionalità, correttezza sintattica	comprensibilità, manutenibilità
Progetti	dimensione, riusabilità, modularità, accoppiamento, coesione, funzionalità	qualità, complessità, manutenibilità
Codifica	dimensione, riusabilità, modularità, accoppiamento, funzionalità, complessità algoritmica, strutturazione del controllo di flusso	affidabilità, usabilità, manutenibilità
Dati di test	dimensione, livello di copertura	qualità
RISORSE		
Personale	costo, età	produttività, capacità, esperienza
Team	dimensione, livello di comunicazione, strutturazione	produttività, qualità
Software	dimensione, prezzo	usabilità, affidabilità
Hardware	memoria, velocità, prezzo	affidabilità
Office	dimensione, ambiente, ecc.	comfort, qualità

Tabella 1 - Modello di Fenton (1991) per la misurazione del software

Attributi interni

Misure interne

• *attributi interni:* sono quelli misurati in relazione all'entità stessa (sono anche denominati *misure interne* all'entità). Ad esempio, una misura interna del codice sorgente è la dimensione, la quale risulta definita dalle linee di codice: il numero di linee di codice può essere ricavato direttamente dall'esame del codice sorgente (si devono solo stabilire delle convenzioni per il conteggio);

• *attributi esterni* sono invece quegli attributi di un'entità misurabili solamente in rapporto con l'ambiente *(misure esterne).* Ad esempio, la manutenibilità del codice sorgente misura quanto una specifica entità di prodotto (il codice sorgente) soddisfa il requisito di poter essere corretta o variata facilmente nel tempo, ed è un attributo esterno in quanto dipendente dall'abilità e dagli strumenti a disposizione delle persone che effettuano la manutenzione.

Le misure di Pressman

Per iniziare un programma di misurazione, Pressman [1993] suggerisce il seguente insieme di misure dirette:

• *metriche per il processo*
- costo per ogni compito di un progetto
- impegno in giorni-uomo per ogni compito di un progetto
- tempo di calendario per completare il progetto

• *metriche per il prodotto*
- linee di codice prodotte (Lines of Code - LOC)
- "punti-funzione" (Function Points - FP)
- pagine di documentazione
- velocità di esecuzione del programma
- occupazione di memoria
- difetti riscontrati in un definito periodo di tempo

• *risorse*
- numero di persone coinvolte
- dimensione del team di sviluppo e livelli di esperienza

I function-points [Albrecht, 1983] vengono derivati utilizzando relazioni empiriche basate su proprietà misurabili «nel dominio delle informazioni del software» e su valutazioni della complessità del software. I valori del dominio delle informazioni sono i seguenti: numero di ingressi dell'utente; numero di uscite dell'utente; numero di interrogazioni dell'utente; numero di files; numero di interfacce leggibili dalla macchina. Mediante i FP si ottiene una misura della dimensione di un programma che è alternativa a quella ricavata dal numero di LOC. Tale misura è anche più significativa, in quanto non dipende dal linguaggio di programmazione utilizzato per la codifica.

Dalle misure alle metriche

- costo degli strumenti
- disponibilità degli strumenti

Da queste misure è poi possibile ottenere un insieme di metriche, fra cui:

- produttività = LOC/persone-mese, oppure FP/persone-mese;
- qualità = n°difetti/LOC, oppure n°difetti/FP;
- costi = costo/LOC prodotte, oppure costo/FP prodotti;
- documentazione = n°pagine di documentazione/LOC, oppure n° pagine di documentazione/FP.

Le due alternative sono dovute alla scelta tra le linee di codice ed il numero di "function-points" quale fattore di normalizzazione; l'utilizzo dei function-points dà in genere delle misure più significative, anche se queste sono più complesse da ottenere. Infatti, è chiaro che la funzionalità di una singola linea di codice dipende molto dal linguaggio di programmazione utilizzato (ad esempio, da studi empirici risulta che per ottenere una funzionalità equivalente ad un "function-point" occorrono all'incirca 300 linee di codice Assembler, mentre ne bastano approssimativamente 70 di codice Ada [Albrecht, 1983]).

Il modello di Mc Call

Nell'ambito della misurazione del software, che come abbiamo visto concerne le tre classi "processi", "prodotti" e "risorse", assume particolare rilevanza per l'utilizzatore finale la misurazione della qualità del prodotto software che egli acquista. Gli attributi in relazione alla qualità del prodotto software vengono anche chiamati "caratteristiche di qualità del software".

I due modelli più noti riguardanti la qualità del prodotto software sono quello di McCall [1979] (Fig.1 a sinistra) e quello di Boehm [1982] (Fig.1 a destra).

Nel modello di McCall vengono individuate tre aree corrispondenti a caratteristiche di prodotto:

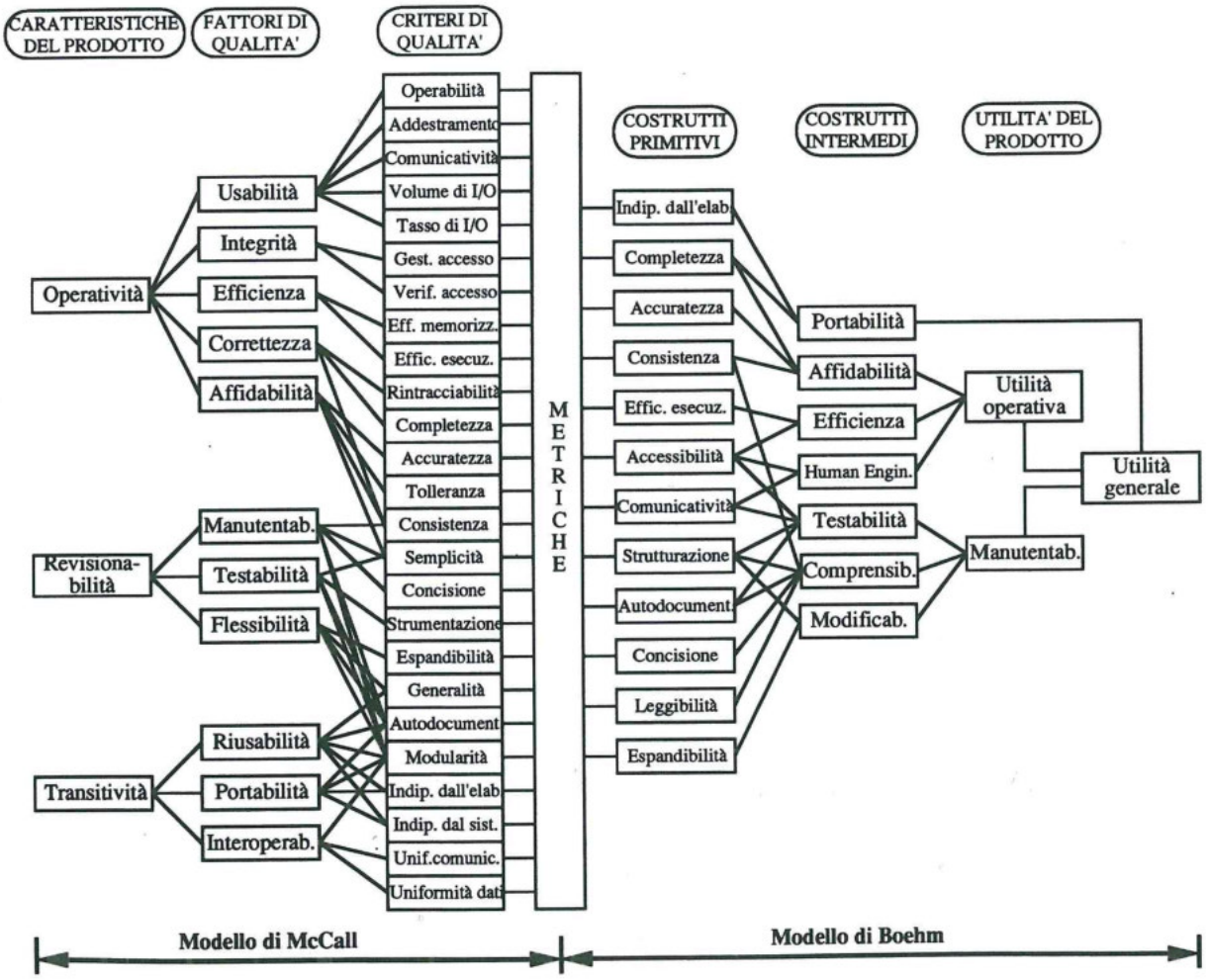

Figura 1 - Modelli inerenti la qualità del software

1. *operatività del prodotto:* richiede che il software possa essere appreso facilmente, operare in maniera efficiente e che i risultati siano quelli richiesti dall'acquirente;

2. *revisionabilità del prodotto:* concerne la correzione e l'adattamento successivo del software, attività che generalmente assorbono costi rilevanti;

3. *transitività del prodotto:* si riferisce alla compatibilità con gli elaboratori e le reti.

Per ognuna di queste aree vengono definiti alcuni importanti fattori di qualità.

L'operatività del prodotto

I fattori di qualità definiti da McCall per l'area dell'*operatività* del prodotto sono:

- *correttezza:* è il grado con il quale un programma soddisfa le specifiche e gli obiettivi prestabiliti;
- *affidabilità:* misura la bontà con cui un programma svolge le proprie funzioni nel tempo secondo gli standard desiderati;
- *efficienza:* misura l'uso efficiente di risorse quali il tempo di CPU, la quantità di memoria, ecc.;
- *usabilità:* è la facilità con la quale l'utente utilizza il prodotto software;
- *integrità:* misura il grado di protezione del programma da accessi non autorizzati.

La revisionabilità del prodotto

Per l'area della *revisionabilità* del prodotto vengono invece definiti i seguenti fattori di qualità:

- *manutenibilità:* misura lo sforzo richiesto per localizzare e correggere un malfunzionamento in un programma mentre questo si trova nel suo ambiente operativo;
- *flessibilità:* individua la facilità di effettuare modifiche richieste da cambiamenti nell'ambiente operativo;
- *testabilità:* misura la facilità di testing del programma, per assicurarsi che sia privo di errori e soddisfi alle specifiche.

La transitività del prodotto

Infine, i fattori di qualità definiti per l'area della *transitività* del prodotto sono:

- *portabilità:* valuta lo sforzo richiesto per trasferire il programma da un ambiente a un altro;
- *riusabilità:* misura la facilità nel riusare il programma in applicazioni differenti (si riferisce alla costruzione di pacchetti software);
- *interoperabilità:* valuta lo sforzo richiesto per accoppiare il sistema ad un altro sistema.

Fattori di qualità e criteri di qualità

Gli undici fattori di qualità proposti vengono a loro volta determinati in base a venticinque criteri di qualità, definiti in Tab.2; le associazioni tra i fattori ed i criteri di qualità sono desumibili dalla Fig.2.

I criteri di qualità sono infine applicati mediante associazione a delle opportune metriche di qualità.

Le metriche utilizzate per la misura della qualità

Fra le metriche maggiormente utilizzate per la misura della qualità [Gillies, 1992]:

- *tempo medio di riparazione* (MTTR - Mean Time To Repair): è il tempo medio di intervento per ripristinare la funzionalità di un sistema a seguito di un avvenuto malfunzionamento (definito come lo scostamento del programma dal comportamento stabilito nelle specifiche dei requisiti). Il MTTR fornisce di fatto una misura indiretta della manutenibilità del codice sorgente;

• *tempo medio tra due malfunzionamenti successivi* (MTBF - Mean Time Between Failure): è definito come MTBF = MTTF+MTTR dove MTTR è il tempo medio di riparazione visto sopra, mentre MTTF (Mean Time To Failure) è il tempo medio al malfunzionamento (corrispondente al periodo di buon funzionamento prima del guasto); il MTBF può essere utilizzato come misura indiretta dell'affidabilità del codice sorgente [Musa et al., 1983];

Numero ciclomatico

• *numero ciclomatico:* è una delle possibili misure della complessità del codice sorgente. Il numero ciclomatico è stato proposto da McCabe [1976] ed è basato sulla teoria dei grafi, con lo scopo di stimare il numero di cammini linearmente indipendenti del programma. La complessità fornisce una misura indiretta dell'affidabilità e della manutenibilità, entrambe in relazione inversa;

OPERABILITÀ	Facilità di operazione di un programma.
ADDESTRAMENTO	Grado di assistenza offerto dal software ai nuovi utenti.
COMUNICATIVITÀ	Grado di facilità di assimilazione da parte dell'utente degli inputs e outputs forniti dal software.
VOLUME DI I/O	Volume di dati trattabile in operazioni di input/output.
TASSO DI I/O	Velocità alla quale è possibile effettuare operazioni di I/O.
GESTIONE DELL'ACCESSO	Caratteristiche di gestione dell'accesso ai programmi e ai dati.
VERIFICA DELL'ACCESSO	Caratteristiche di verifica dell'accesso ai programmi e ai dati.
EFFICIENZA DI MEMORIZZAZIONE	Caratteristiche del programma che consentono i minimi requisiti di memorizzazione nel corso delle operazioni.
EFFICIENZA DI ESECUZIONE	Caratteristiche del programma che consentono il minimo tempo d'esecuzione.
RINTRACCIABILITÀ	Possibilità di collegare rapidamente con i requisiti originali un'entità progettuale o un componente effettivo del programma.
COMPLETEZZA	Livello raggiunto di realizzazione della funzione richiesta.
ACCURATEZZA	Precisione dei calcoli e dei controlli.
TOLLERANZA AGLI ERRORI	Danno arrecato nel caso che il programma incontri un errore.
CONSISTENZA	Uso di tecniche di progettazione e di documentazione uniformi in tutto il progetto di sviluppo del software.
SEMPLICITÀ	Livello di difficoltà relativo alla comprensione del programma.
CONCISIONE	Compattezza del programma in termini di linee di codice.
STRUMENTAZIONE	Grado con cui il programma controlla le sue stesse operazioni e identifica eventuali errori.
ESPANDIBILITÀ	Grado di estensione possibile dell'architettura, della struttura dei dati o delle procedure del programma.
GENERALITÀ	Ampiezza dell'applicazione potenziale dei componenti del programma.
AUTO-DOCUMENTAZIONE	Quantità di documentazione significativa fornita dal codice sorgente.
MODULARITÀ	Indipendenza funzionale dei componenti (moduli) che costituiscono la struttura del programma.
INDIPENDENZA DAL-L'ELABORATORE	Grado di disaccoppiamento del software dall'hardware su cui opera.
INDIPENDENZA DAL SISTEMA SOFTWARE	Grado di indipendenza del programma rispetto alle caratteristiche non standard del linguaggio di programmazione, del sistema operativo e altri vincoli.
UNIFORMITÀ DI COMUNICAZIONE	Grado con cui vengono utilizzati protocolli d'interfaccia.
UNIFORMITÀ DEI DATI	Uso di strutture e tipi di dati uniformi all'interno del programma.

Tabella 2 - Criteri di misurazione della qualità del software (Mc Call, 1979)

Leggibilità

• *leggibilità:* i due indici di leggibilità più utilizzati sono quello di Flesch-Kincaid e quello di Fog [Gunning, 1968]: operando a livello di frasi, parole e sillabe forniscono un indice di facilità della comprensione di un testo. Possono essere utilizzati per misurare la leggibilità della documentazione di un programma e quindi, in maniera indiretta, anche la sua usabilità.

Il modello di Boehm: utilità del prodotto, costrutti intermedi e primitivi

Il secondo modello citato, proposto da Boehm, presenta tre livelli gerarchici (si veda la parte destra di Fig.1). La prima suddivisione viene effettuata in accordo alle cosiddette "utilità" del prodotto; vengono distinte: *utilità generali* ed *utilità operative* del prodotto, dove le utilità operative sono un sottoinsieme di quelle generali e sono approssimativamente corrispondenti all'area dell'*operatività del prodotto* incontrata nel modello di McCall. La seconda suddivisione genera il livello dei *costrutti intermedi* della qualità, i quali vengono a loro volta definiti mediante dei *costrutti primitivi* della qualità del software (terzo livello); questi costrutti primitivi vengono infine associati alle metriche di qualità le quali ne consentono la misura.

Il limite principale

Il limite principale di entrambi i modelli consiste nella loro non facile applicabilità a casi concreti. Infatti, le relazioni tra le metriche, i criteri di qualità ed i fattori di qualità sono in genere complesse: infatti, ad uno stesso criterio di qualità possono

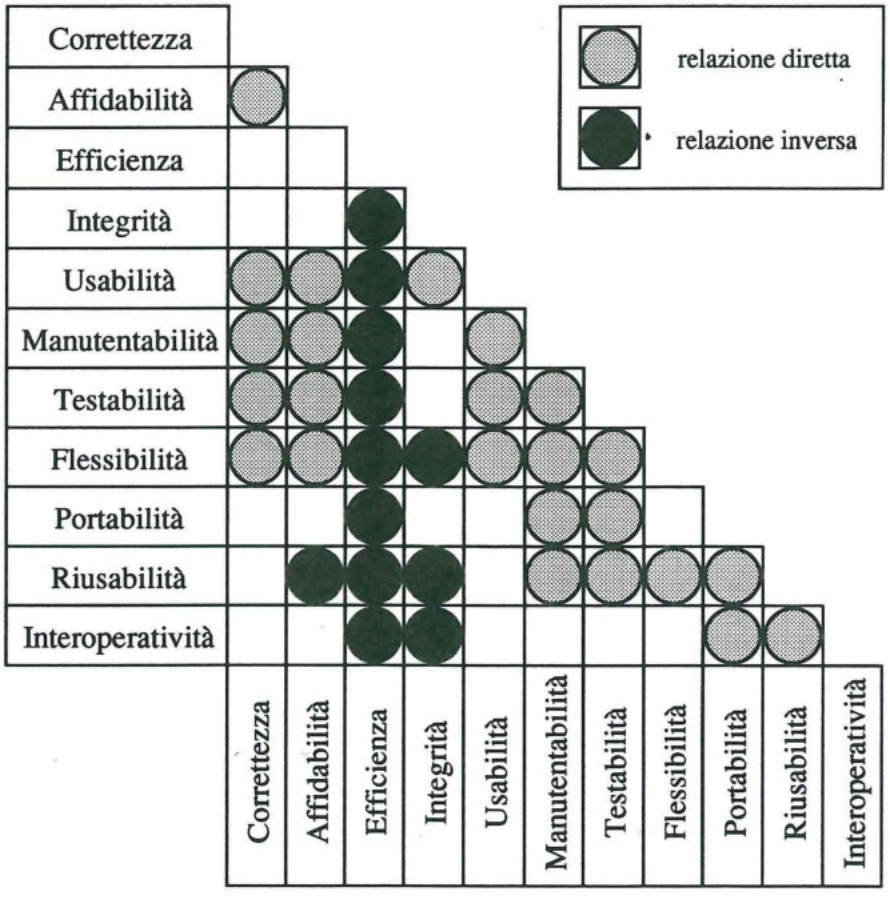

Figura 2 - Relazioni dirette e inverse tra i fattori della qualità (Perry, 1981)

essere associate più metriche e, viceversa, una metrica può essere associata a più criteri di qualità.

Inoltre, non tutti i fattori di qualità possono essere massimizzati contemporaneamente: infatti tra i fattori di qualità è possibile individuare una relazione che può essere inversa, diretta o neutra.

Relazioni tra i fattori di qualità

Perry [1981] ha proposto uno schema (Fig.2) nel quale sono esposte le relazioni che legano tra loro i fattori di qualità del modello di McCall.

Ad esempio, l'usabilità e l'efficienza sono due fattori tra i quali esiste "trade-off", in quanto un codice con una migliore interfaccia tra l'uomo e l'elaboratore può incrementare in maniera significativa la quantità di codice e la potenza di calcolo richieste; ciò significa che la necessità di aumentare l'usabilità di un programma può comportare una minore efficienza dello stesso. Nel caso di presenza di "trade-off" bisogna specificare un sistema di priorità.
La portabilità e la riusabilità, invece, sono in relazione diretta, in quanto un codice portabile è più facilmente privo di istruzioni specifiche dell'ambiente per il quale è stato sviluppato ed è anche solitamente ben strutturato; queste due caratteristiche lo rendono perciò facilmente riusabile.
Altri fattori di qualità sono tra loro in relazione neutra. È questo il caso della correttezza e dell'efficienza: un codice può essere più o meno corretto senza che per questo la sua efficienza ne sia influenzata.

Misure di qualità del sfotware indirette

Un altro genere di misure della qualità del software è costituito da tutta *la serie di misure indirette* che tendono a valutare la soddisfazione dei clienti. Il principio che sta alla base di questo tipo di misure è semplice: mediante apposite indagini, si individuano le percentuali dei clienti soddisfatti sui non soddisfatti e su quelli indifferenti, per ognuno degli attributi considerati e per il livello di soddisfazione globale.

Misure della soddisfazione del cliente per alcuni attributi del software vengono già utilizzate da alcune aziende. Ad esempio, IBM monitora i livelli di soddisfazione "CUPRIMDSO" (acronimo di: Capability, Usability, Performance, Reliability, Installability, Maintainability, Documentation, Servicing, Overall Satisfation) dei clienti, vale a dire riguardo a: funzionalità, usabilità, prestazioni, affidabilità, installabilità, servizio, soddisfazione globale del prodotto software. Hewlett-Packard invece focalizza l'attenzione sui valori "FURPS" (acronimo di: Functionality, Usability, Reliability, Performance, Supportability), ovvero: funzionalità, usabilità, affidabilità, prestazioni e supportabilità (indipendenza dall'hardware e dal sistema operativo) del prodotto software [Basili et al., 1994].

Definito cosa è necessario considerare quando si parla della qualità del software, affronteremo ora il tema delle normative internazionali per la certificazione della qualità e dei Sistemi Qualità del software.

Normative internazionali

Allo scopo di garantire lo sviluppo di software rispondente ai requisiti prefissati, fin dagli anni '70 sono state sviluppate alcune normative per l'*assicurazione della qualità del software* (SQA, Software Quality Assurance).

I principali enti, nazionali ed internazionali, promotori di norme per l'SQA sono:

- NBS *(National Bureau of Standard* americano*)*
- IEEE *(Institute of Electronic and Electrical Engineers* americano*)*
- DOD/JLC *(Join Logistic Command of the Department of Defense americano),* US Navy e US Army con le normative DOD e MIL *(Military)*
- FAA *(Federal Aviation Administration* americano*)*
- NASA *(National Aeronautics and Space Administration* americano)
- ESA *(European Space Agency)*
- NATO *(North Atlantic Treaty Organization)*
- ISO *(International Organization for Standardization)*
- IAEA *(International Atomic Energy Agency)*
- IEC *(International Electronic Committee).*

Standard principali

Molte delle normative per l'SQA vengono utilizzate in settori particolari: ad esempio le normative NASA ed ESA per il settore aerospaziale, le normative IAEA e IEC per il settore nucleare e le normative DOD e MIL per il il settore militare.

Normative di utilizzo più generale sono quelle IEEE e, recentemente, la norma ISO 9000/3 che tratteremo in seguito.

Il primo standard per l'SQA (MIL-S-52779 "Requisiti di un programma per la qualità del software" dell'US Army) fu approvato nel 1974. Questo standard fondamentale ha fortemente influenzato tutti gli altri successivi [Schulmeyer, 1992].

Approccio strutturato e "top-down"

Altro standard importante - a ragione del suo largo impiego nelle forniture di carattere militare - è DOD-2167-A "Sviluppo di software di sistema per la difesa" del 1988. La sua caratteristica è di far riferimento a un ben preciso *modello di ingegneria del software* che comprende sia lo sviluppo del software che quello dell'hardware. Lo standard prescrive un approccio strutturato e "top-down" alla progettazione di sistema e allo sviluppo. Un'enfasi particolare viene posta sulle fasi di analisi dei requisiti e di formulazione delle specifiche di progettazione. Ad ogni passo, tale standard richiede una documentazione in grado di dimostrare il soddisfacimento dei requisiti, scritti in un prestabilito formato.

A differenza di DOD-2167A, lo standard IEEE STD 730, "Standard IEEE di prova-uso per i piani di assicurazione della qualità del software" del 1980, è di maggiore interesse per tutti i tipi di applicazione. L'IEEE, con questo ed altri standard correlati (IEEE STD 828 per i piani di gestione della configurazione, IEEE STD 1012 per i piani di verifica e validazione, ecc.), ha inteso fornire alcuni strumenti di validità universale, adattabili alle singole situazioni di sviluppo. Fra le categorie di software che possono essere interessate dalla normativa IEEE citiamo: il software di sistema, il software "real time", il software per Electronic Data Processing (EDP), il software per personal computer, il software ingegneristico e scientifico, il software per AI (intelligenza artificiale).

L'uniformazione delle norme per i Sistemi Qualità

La normativa ISO, mediante la serie ISO 9000, promette però qualcosa in più rispetto alle normative per l'SQA viste sopra: l'uniformazione delle norme per i Sistemi Qualità e per l'assicurazione della qualità, in base a principi comuni a tutti i settori merceologici, non solo al software.

La norma ISO 9000/3

La norma ISO 9001:1987, dal titolo "Sistemi Qualità - Criteri per l'assicurazione della qualità nella progettazione, sviluppo, fabbricazione, installazione ed assistenza", è la norma della serie che più si adatta al caso dello sviluppo del software, in particolare su commessa.

Tuttavia, l'applicazione diretta della norma ISO 9001 ai Sistemi Qualità del software può essere difficoltosa a causa della natura particolare di questo tipo di prodotto. Per evitare ciò, è stata sviluppata la norma ISO 9000/3:1991, che descriveremo di seguito. È importante sottolineare il fatto che questa norma si propone esclusivamente come guida per l'interpretazione della ISO 9001 applicata allo sviluppo del software, e non intende perciò modificare i contenuti della ISO 9001.

Certificazione del sistema qualità per il software

Realizzato il Sistema Qualità conforme alla norma ISO 9001 (con la guida di ISO 9000/3), il passo successivo è la certificazione del Sistema Qualità. La certificazione coivolge un ente e-

sterno accreditato. In Italia, l'unico ente attualmente abilitato alla certificazione nel settore dell'informatica è l'IMQ (Istituto Marchio di Qualità). Tale certificazione permette di aderire all'ITQS ("recognition arrangements for assessment and certification of Quality Systems in the Information Technology sector").

Infine, per ultimi ma non meno importanti, ecco alcuni miglioramenti di processo e di prodotto che dovrebbero essere consentiti dall'implementazione di un Sistema Qualità per il software [cfr. Price Waterhouse, 1988]:

Benefici attesi

- *riduzione dei costi* - Una pianificazione più dettagliata e uniforme del progetto dovrebbe consentire un monitoraggio del processo con correzione anticipata degli errori e minor impegno nel controllo-qualità e nel testing. I minori costi sono tuttavia in parte controbilanciati da quelli dovuti all'implementazione del Sistema Qualità;
- *risparmio di tempo* - La più accurata documentazione dei vecchi progetti dovrebbe consentire una riduzione dei tempi di completamento di un nuovo progetto, se questo utilizza parte dei progetti già realizzati;
- *affidabilità* - Il Sistema Qualità dovrebbe ridurre gli errori che arrivano agli utenti e quindi accrescere l'affidabilità del software.
- *funzionalità* - L'uso di un Sistema Qualità, attraverso i riesami di contratto, di progetto e la prova di accettazione dovrebbero facilitare il soddisfacimento dei requisiti del committente;
- *manutenibilità.* La migliore documentazione e lo sviluppo effettuato anche tenendo in considerazione le future esigenze di manutenzione dovrebbero ridurre la necessità di cambiamenti del prodotto software e facilitarli quando necessari.

Analizziamo ora in maggior dettaglio la norma ISO 9000/3.

La norma ISO 9000/3

Origine della norma ISO 9000/3

La norma UNI ISO 9000/3:1991 (dal titolo "Regole riguardanti la conduzione aziendale per la qualità e l'assicurazione della qualità - Guida per l'applicazione della ISO 9001 - UNI EN 29001 allo sviluppo, alla fornitura ed alla manutenzione del software") è stata concepita specificatamente per essere utilizzata dagli sviluppatori di software. Tuttavia, l'ente preposto alla certificazione deve verificare la conformità del Sistema Qualità aziendale alla norma ISO 9001 e non alla ISO 9000/3. È perciò comprensibile come possano insorgere delle perplessità e l'utente si possa domandare quale sia la norma alla quale fare davvero riferimento.

Sviluppatori di software

Rapporti tra ISO 9000/3 e ISO 9001

Un suggerimento potrebbe essere quello di servirsi prima della norma ISO 9000/3, più vicina alla terminologia ed alle metodologie di sviluppo dell'ingegneria del software, allo scopo di verificare lo stato del proprio processo di sviluppo. Successivamente, per l'implementazione effettiva del Sistema Qualità è opportuno utilizzare la norma ISO 9001. Le tabelle di correlazione tra ISO 9001 e ISO 9000/3 (si veda: ISO 9000/3, Appendici A e B) potranno poi fornire un ulteriore supporto.

Contenuti della norma ISO 9000/3

In Tab.3 sono riportati gli indici di correlazione tra ISO 9000/3 e ISO 9001 desunti dalla suddetta Appendice A. Nell'ultima colonna viene riportato anche il tipo di contributo che la norma per il software dà rispetto alla norma di carattere generale; questo è stato classificato da Gillies [1992] in: nullo (vd. tabella), di poco conto, significativo e determinante.

Lo sviluppo del software nell'ambito di un sistema qualità

Come anticipato, la struttura della norma ISO 9000/3 differisce da quella della norma ISO 9001, e questo per adattarsi meglio alle caratteristiche peculiari del prodotto software. Vediamo brevemente il contenuto delle sei sezioni che compongono la norma.

• *Sezioni 1, 2 e 3: Materiale introduttivo.* Le prime tre sezioni della norma ISO 9000/3 riguardano la definizione dei suoi obiettivi e i riferimenti alle altre norme.

• *Sezione 4: Sistema Qualità* - Struttura. Questa parte contiene quattro sottosezioni: *responsabilità della direzione, Sistema Qualità, verifiche ispettive interne del Sistema Qualità e azioni correttive.* Gli argomenti trattati in questi punti sono molto simili a quelli corrispondenti della norma ISO 9001.

• *Sezione 5: Sistema Qualità - Attività relative al ciclo di vita.* Questa sezione contiene nove sottosezioni, le quali trattano le attività relative a una o più parti del ciclo di vita del software. Alcune delle corrispondenti sezioni della ISO 9001 non appaiono sostanziali quando applicate al software.

• *Sezione 6: Sistema Qualità - Attività di supporto (trasversali a tutte le fasi).* Le attività di questa sezione, che vengono trattate in maniera molto più completa rispetto a quanto fatto nella norma ISO 9001, includono: la "gestione della configurazione", la "misurazione", le "regole, pratiche e convenzioni" e gli "strumenti e tecniche".

Dalle sezioni 5 e 6, abbiamo tratto in forma sintetica (a volte

Vengono definiti sette termini fondamentali:
- software: creazione intellettuale che comprende i programmi, le procedure, le regole e la relativa documentazione, pertinenti al funzionamento di un sistema per l'elaborazione di dati;
- prodotto software: insieme completo dei programmi per elaboratore, delle procedure, dei dati e della relativa documentazione, designati per la consegna ad un utente;
- elemento software: parte identificabile di un prodotto software, ad uno stadio intermedio od allo stadio finale dello sviluppo;
- sviluppo: tutte le attività da svolgere per creare un prodotto software;
- fase: segmento definito di lavoro;
- verifica (per il software): processo di valutazione dei

TITOLO SEZIONE	SOTTO-SEZ.	TITOLO SOTTO-SEZIONE	Correlaz. con ISO 9001	Contributo alla ISO 9001
4. Sistema Qualità - - Struttura.	4.1	Responsabilità della direzione	4.1	■ ■
	4.2	Sistema Qualità (SQ)	4.2	■ ■
	4.3	Verifiche ispettive interne del SQ	4.17	■
	4.4	Azioni correttive	4.14	■
5. Sistema Qualità - - Attività del ciclo di vita.	5.1	Generalità	-	-
	5.2	Riesame del contratto	4.3	■ ■
	5.3	Specifica dei requisiti del committente	4.3, 4.4	■ ■
	5.4	Pianificazione dello sviluppo	4.4	■ ■
	5.5	Pianificazione della qualità	4.2, 4.4	■ ■
	5.6	Progettazione e realizzazione	4.4-9-13	■ ■
	5.7	Prova (test) e validazione	4.4-10-11-13	■ ■
	5.8	Accettazione	4.10, 4.15	■ ■
	5.9	Duplicazione, consegna e installazione	4.10-13-15	■ ■
	5.10	Manutenzione	4.13, 4.19	■ ■ ■
6. Sistema Qualità - - Attività di supporto.	6.1	Gestione della configurazione	4.4-5-8-12-13	■ ■ ■
	6.2	Controllo della documentazione	4.5	■ ■
	6.3	Documenti di registrazione della qualità	4.16	-
	6.4	Misure	4.20	■ ■ ■
	6.5	Regole, pratiche e convenzioni	4.9, 4.11	■ ■
	6.6	Strumenti e tecniche	4.9, 4.11	■ ■
	6.7	Approvvigionamenti	4.6	■
	6.8	Prodotti software inclusi	4.7	■ ■
	6.9	Addestramento	4.18	■

Contributo:
- nullo
■ di poco conto
■ ■ significativo
■ ■ ■ determinante

Tabella 3 - Tabella di correlazione tra ISO 9000/3 e ISO 9001 (cfr. App. A di ISO 9000/3) e contributo offerto dalla normativa ISO 9000/3 ad integrazione della normativa ISO 9001 (Gillies, 1992)

integrandole con ulteriori elementi) alcune fra le più importanti attività relative allo sviluppo del software da svolgere nell'ambito di un Sistema Qualità. Esse sono suddivise in:

- attività relative al ciclo di vita;
- attività di supporto.

Attività relative al ciclo di vita

ATTIVITÀ RELATIVE AL CICLO DI VITA (sono le attività da svolgere sequenzialmente nel corso del processo di sviluppo del prodotto software).

La sequenza di attività riprende parzialmente i titoli delle sottosezioni delle sezioni 5 e 6 della norma ISO 9000/3.

1) Formulazione del documento di specifica dei requisiti.

Specifica dei requisiti

Consiste in un insieme completo e non ambiguo di requisiti funzionali. I requisiti devono essere espressi in termini comprensibili al committente, al quale dev'essere fornita tutta l'assistenza necessaria per evitare eventuali incomprensioni sulla terminologia adottata dal fornitore. Inoltre, i requisiti devono essere sufficientemente dettagliati da permetterne la validazione in fase di accettazione del prodotto software.

Riesame del contratto

2) Riesame del contratto.

I requisiti funzionali devono essere definiti e documentati. Eventuali scostamenti dei medesimi dall'offerta iniziale devono essere risolti in questa fase. Solo successivamente si può procedere con le fasi seguenti dello sviluppo, e la progettazione può così aver luogo.

Piano di sviluppo

3) Preparazione del piano di sviluppo.

Il piano di sviluppo, assieme alla specifica dei requisiti funzionali, è il documento fondamentale di ogni progetto. In esso devono comparire i seguenti elementi:

- definizione del progetto (suoi obiettivi e descrizione dei progetti correlati);
- organizzazione delle risorse necessarie al compimento del progetto, incluse la struttura del gruppo di lavoro e le responsabilità;
- fasi del progetto (descrizione delle fasi del progetto, dei dati e requisiti necessari in ingresso ad ogni fase e dei risultati attesi in uscita da ogni fase);
- programma temporale del progetto (descrizione di tutte le attività, delle loro interrelazioni e delle risorse richieste in termini di tempo, strumenti e personale);
- identificazione dei piani correlati (piano della qualità - vedi attività 4; piano di prova - vedi attività 5; piano di gestione della configurazione - vedi attività 10).

Piano della qualità

4) Preparazione del piano della qualità.

È questo un documento molto importante in un Sistema Qualità. Gli obiettivi di qualità devono essere identificati ed espressi in termini misurabili ogni qualvolta ciò sia possibile. Devono essere definiti criteri di ingresso/uscita per ciascuna fase dello sviluppo.

Devono essere definite le responsabilità specifiche per l'esecuzione di attività inerenti la qualità, quali riesami e test, gestione della configurazione e controllo delle modifiche, controllo dei difetti ed azioni correttive.

Le non-conformità devono essere gestite tramite procedure standard, e le azioni correttive intraprese devono essere registrate.

prodotti di una fase per assicurarne la correttezza e la coerenza rispetto ai prodotti ed alle norme richieste dalle fasi successive; - validazione (per il software): processo di valutazione del software per assicurarne la conformità ai requisiti specificati [ISO 9000/3, punti da 3.1 a 3.7].

Piano di prova metodologie

5) Preparazione del piano di test (o piano di prova).

Il piano di test deve essere preparato prima di intraprendere il test stesso, e deve contenere i seguenti elementi: casi di test, dati di test e risultati attesi, criteri per giudicare il superamento della prova.

6) Metodologie e pratiche di progettazione e realizzazione.

La progettazione e la realizzazione del prodotto software deve avvenire mediante metodologie stabilite. Ad esempio, per l'analisi dei requisiti si possono utilizzare le metodologie SAM ("Structured Analysis Methodology" [De Marco, 1979]) e SSADM ("Structured System Analysis and Design Methodology", utilizzata come standard dal governo britannico); un'altra tecnica strutturata di analisi e progettazione è SADT ("Structured Analysis and Design Technique", marchio della Softech Inc., distribuito in Italia da Finsiel). Spesso tali metodologie sono associate a strumenti automatizzati o CASE ("Computer Aided Software Engineering").

Inoltre, i dati di progetto devono essere registrati, prevedendone l'utilizzo in esperienze di progetto future. La progettazione stessa deve avvenire in maniera da facilitare le attività successive, quali il test, la manutenzione e l'uso.

Riesami di progettazione

7) Riesami di progettazione.

I riesami consistono nella verifica del soddisfacimento dei requisiti specificati e nell'applicazione delle metodologie di progettazione stabilite. Se un riesame rivela delle deficienze, queste devono essere registrate e risolte prima di procedere con la realizzazione del prodotto.

Accettazione

8) Accettazione.

Con l'accettazione il committente valida il prodotto, dichiarando che questo è conforme ai requisiti specificati e concordati con il fornitore all'atto della sottoscrizione del contratto. Anche l'accettazione ed il trattamento degli eventuali problemi insorti al momento della stessa devono avvenire tramite procedure precedentemente concordate.

Piano di manutenzione

9) Piano di manutenzione.

La manutenzione, quando è prevista dalle clausole contrattuali, deve avvenire secondo procedure pianificate e concordate con il committente. Sono quattro i tipi di manutenzione che si incontrano [Pressman, 1993]:

- manutenzione correttiva: si riferisce all'attività di correzione degli errori individuati successivamente alla consegna del software al cliente;
- manutenzione adattativa: è l'attività di adattamento del software a seguito dell'evoluzione dell'hardware o dei sistemi operativi in uso;
- manutenzione perfettiva: avviene con riferimento ad un package di successo. Infatti, se il pacchetto s'impone sul mercato, allora è più probabile che il cliente senta la necessità di richiedere nuove funzionalità o di potenziare quelle esistenti;
- manutenzione preventiva (o reingegnerizzazione): è dovuta alla necessità di modificare il software al fine di migliorarne la manutenibilità o l'affidabilità future, o di fornire una piattaforma migliore per future evoluzioni. È un'attività ancora poco frequente nel mondo del software.

Attività di supporto

ATTIVITÀ DI SUPPORTO (sono trasversali al ciclo di vita, non dipendono da una singola fase):

Configurazione

10) Gestione della configurazione.

Ogni elemento software dev'essere identificato, controllato, e deve poter essere rintracciato a partire da una data configurazione finale del prodotto software. Dell'elemento software devono essere indicate:

- organizzazioni coinvolte nella preparazione e loro responsabilità;
- strumenti, tecniche e metodologie adottati.

11) Controllo della documentazione.

Questa attività consiste nello stabilire e tenere aggiornate le procedure necessarie al controllo di tutti i documenti in relazione alla norma.

Documentazione

12) Documenti di registrazione della qualità.

Tali documenti registrano il conseguimento della qualità richiesta e l'efficacia del Sistema Qualità. Inoltre, devono essere correlabili ai prodotti ai quali si riferiscono.

Misurazione

13) Misurazione.

La misurazione deve avvenire anche per le caratteristiche di qualità (oltre che per quelle di produttività/costo) del processo e del prodotto, utilizzando le metriche che allo stato dell'arte più si adattano al caso in esame. Alcune delle metriche in uso sono state riportate nella prime parti di questo articolo.

Strumenti

14) Utilizzo di strumenti e tecniche.

Fra gli strumenti automatizzati che supportano l'intero ciclo di sviluppo del software: IEF (Information Engineering Facility) della Texas Instruments, basato sulla metodologia di sviluppo IEM (Information Engineering Methodology); Cohesion (della Digital Equipment Corporation); Softbench (della Hewlett-Packard); A/D Cycle (della IBM) e NSE (della Sun Microsystems).

Addestramento

15) Addestramento.

L'addestramento ("training") è una delle attività fondamentali del Sistema Qualità. Il personale coinvolto nel processo deve conoscere esattamente le proprie mansioni ed essere preparato per svolgerle. Si deve mantenere la registrazione del profilo professionale di ogni dipendente, comprendente tutte le attività di addestramento al quale è stato sottoposto e le esperienze maturate.

L'implemetazione del Sistema Qualità nel gruppo EURIS

Il gruppo EURIS, con sedi a Trieste, Padova, Milano e Fiume, conta attualmente una cinquantina di dipendenti, fattura all'incirca quattro miliardi ed è composto da tre società: EURIS S.r.l., EIDOS S.r.l. e ATIS S.r.l. Ogni società svolge delle attività specifiche relativamente allo sviluppo del software: EURIS si occupa prevalentemente dell'analisi dei requisiti, della progettazione di software e dell'assistenza al cliente; EIDOS, che opera principalmente su commesse di EURIS, effettua le fasi successive di codifica, testing e manutenzione; ATIS, infine, fornisce soluzioni in ambito manifatturiero a partire da un package MRP II.

Vantaggi attesi dal sistema qualità

A metà del 1994 il gruppo si pose come obiettivo l'istituzione di un Sistema Qualità, da attuarsi nell'arco di due anni. I vantaggi attesi dal Sistema Qualità furono così definiti:

- una migliore definizione delle interfacce fra le tre società nell'ottica cliente-fornitore. Questo comporta essenzialmente la

- sequenza e interrelazioni delle attività,
- documenti previsti in ogni fase.

Inoltre, il manuale della qualità è il primo documento che l'istituto di certificazione IMQ, che abbiamo visto essere finora l'unico in Italia abilitato alla certificazione dei Sistemi Qualità aziendali nel settore dell'informatica, esamina prima ancora di effettuare la visita all'azienda interessata.

Si è quindi stabilito che fosse proprio il leader del progetto qualità a stendere la prima bozza del manuale della qualità.

Il manuale non deve essere troppo burocratico, ma deve nello stesso tempo indicare in maniera chiara e precisa quali sono i compiti e le responsabilità di ognuno.

Visto che il manuale sarà il principale strumento di lavoro in azienda, è fondamentale che esso ottenga anche l'approvazione di tutto il management coinvolto.

Il processo di stesura del manuale è perciò iterativo. Man mano che le procedure vengono implementate e collaudate, inizialmente su progetti pilota e poi in tutta l'azienda, il manuale dev'essere rivisto ed adattato. Anche successivamente, con il Sistema Qualità operante a regime, periodicamente sarà necessario controllare la coerenza tra il manuale e gli obiettivi aziendali.

Utilizzo del CASE

Fase 4: selezione dei metodi e degli strumenti CASE (Computer Aided Software Engineering).

Questa fase comporta l'istituzione di un team per la scelta di metodi e strumenti. Le scelte fatte, dovranno essere in accordo a quanto stabilito per le scelte relative al processo, operate nel manuale della qualità. Nel caso del gruppo EURIS, il compito di guidare il team è stato assegnato al leader del progetto qualità. Il team è composto dalle persone che, in azienda, più conoscono i metodi e gli strumenti CASE.

Progetto pilota

Fase 5: avvio di un progetto pilota.

Prima di procedere con le fasi successive del Sistema Qualità, è opportuno sperimentare su un singolo progetto le scelte di processo, metodi e strumenti fatti nelle fasi precedenti.

Questo permetterà una taratura di queste scelte, e darà informazioni preziose su come estendere il nuovo modo di operare a tutta l'azienda.

Inoltre, il progetto pilota può essere affidato a poche persone e molto ben motivate, e ha effetti inizialmente limitati sul sistema esistente.

I criteri che conducono alla scelta del progetto pilota devono tener conto delle esigenze di sperimentazione; pertanto esso non dev'essere un progetto critico in termini di tempi di consegna e di risorse utilizzate. Ma, visto che è in questa fase che vengono stabiliti gli standard di analisi, progettazione, codifica e testing che saranno successivamente estesi a tutto il sistema, il progetto deve anche essere abbastanza significativo da fornire una casistica valida per tutti i progetti esistenti.

Formazione del personale

Fase 6: Formazione del personale.

La formazione è essenziale per fornire ad ogni figura professionale le conoscenze necessarie ad operare correttamente nel Sistema Qualità.

Esistono comunque una serie di benefici che gli strumenti CASE possono apportare alla qualità del software [Gillies, 1992]:
- produttività: lo scopo di questi strumenti è proprio quello di massimizzarne la produttività;
- consistenza: questi strumenti forniscono una "enciclopedia" dei dati alla quale gli sviluppatori si devono riferire. Questo permette che parecchi sviluppatori lavorino separatamente mantenendo la consistenza in termini di variabili, dati, sintassi e così via. In grandi progetti, questo fatto da solo può giustificare l'uso di strumenti CASE;
- automazione della metodologia: molti strumenti CASE sono associati a una metodologia soggiacente; lo sviluppatore si trova costretto a seguirla. Questo migliora la consistenza, anche se limita la creatività;
- incoraggiamento alle "buone pratiche": fornendo una metodologia, lo strumento assicura che vengano applicate metodologie come la strutturazione, ecc.;
- documentazione: questa è un'area dello sviluppo di sistemi notoriamente sottovalutata. Gli strumenti CASE possono fornire vari livelli di automazione nell'assistere al processo di produzione della documentazione del software;
- manutenzione: il costo della manutenzione è stato il motivo principale all'introduzione degli strumenti CASE. Gli strumenti possono aiutare nel miglioramento della qualità iniziale e rendere i cambiamenti da realizzare meno costosi.

Gestione del Sistema Qualità

Fase 7: Gestione del Sistema Qualità.

In questa fase il Sistema Qualità è reso operante ed efficace.

Il team guidato dal leader del progetto qualità viene sciolto, e si forma una struttura permanente. Il funzionamento a regime si otterrà comunque solo dopo del tempo.

Ricordiamo che il Sistema Qualità va periodicamente riesaminato. Se il sistema verrà poi certificato, l'ente certificatore stesso effettuerà dei riesami periodici.

Vediamo ora in maniera più particolareggiata quali sono le procedure e le relative responsabilità che devono operare a regime.

Responsabilità di sviluppo del software con sistema qualità a regime: procedura in EURIS

Responsabilità per lo sviluppo di software nella gestione del Sistema Qualità a regime.

Come abbiamo visto, la conformità del Sistema Qualità alla normativa ISO 9000 implica lo svolgimento di una serie di attività specifiche. È stata quindi preparata una procedura per lo svolgimento di tali attività nel gruppo EURIS.

In Fig. 3 abbiamo rappresentato in forma sintetica tale procedura. Nelle colonne compaiono le posizioni di responsabilità presenti in azienda, mentre nelle righe i tipi di documenti di output del processo di sviluppo software. Le celle della figura indicano invece le attività specifiche del processo. È così possibile determinare per ogni singola attività quale sia l'output atteso ed anche di chi sia la relativa responsabilità. La sequenza delle attività è rappresentata da una linea continua, mentre la partecipazione ad una medesima attività da parte di più persone è rappresentata da una linea tratteggiata.

Bozza di contratto e specifiche software

Redazione della bozza di contratto e delle specifiche software.

Il direttore commerciale, in collaborazione con il committente o con un suo rappresentante, formula una bozza di con-

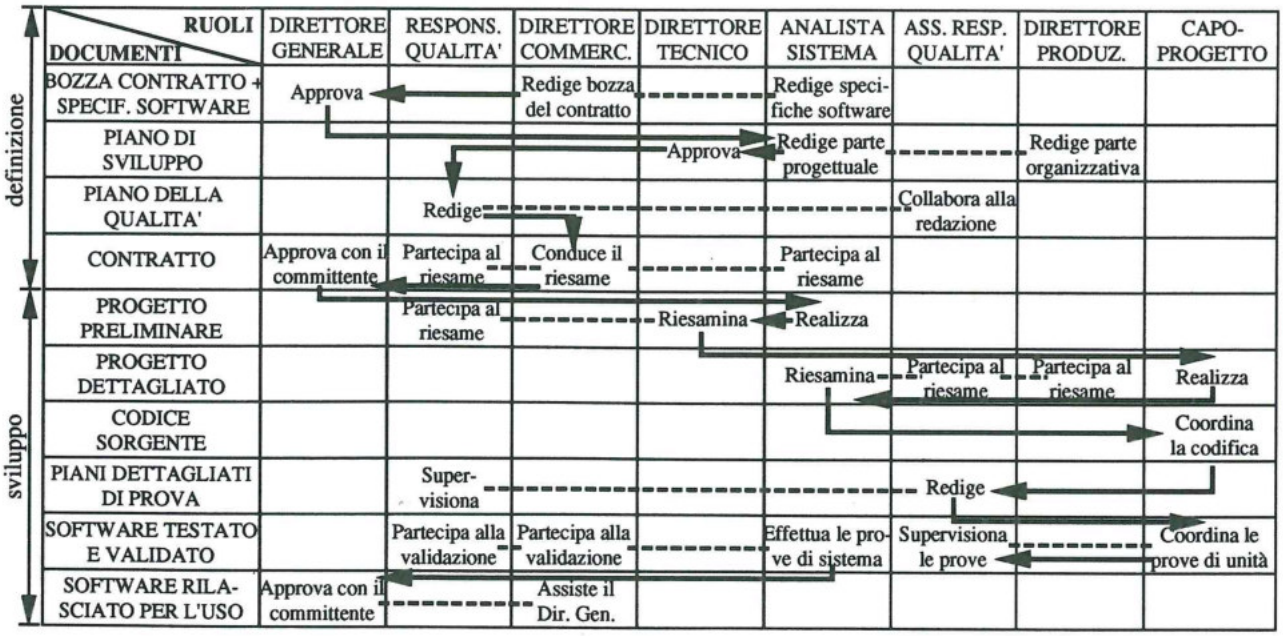

Figura 3 - Procedura per la definizione e lo sviluppo del software nel gruppo Euris

tratto, che prevede una sintesi degli obiettivi, risorse, costi, studio di fattibilità e tempi relativi al software che si vuole sviluppare. Contestualmente, l'analista di sistema prepara le specifiche funzionali del software. Il documento di specifica dei requisiti include: il dominio delle informazioni, le funzioni, le interfacce, i criteri di validazione ed i vincoli di progetto del software da sviluppare.

A questo punto il direttore generale dà l'assenso alla prosecuzione del nuovo progetto.

La bozza di contratto verrà successivamente vagliata nella fase di riesame del contratto, allo scopo di definire e risolvere tutte le ambiguità e le variazioni nelle specifiche del prodotto.

Piano di sviluppo

Redazione del piano di sviluppo.

Il piano di sviluppo è costituito da una parte *progettuale* ed una parte *organizzativa*. La prima parte, consistente nella definizione delle fasi del progetto e nell'identificazione dei piani correlati (piano della qualità, piano di gestione della configurazione, piano di prova e piano d'integrazione) viene redatta dall'*analista di sistema,* il quale svolgerà anche la successiva fase della *progettazione preliminare.*

La parte organizzativa, riguardante l'organizzazione delle risorse ed il programma temporale, è invece compito del direttore di produzione di EURIS.

A questo punto il piano di sviluppo completo viene approvato dal direttore tecnico.

Piano della qualità

Redazione del piano della qualità.

Questa attività è svolta dal *responsabile della qualità,* la nuova figura dirigenziale introdotta in EURIS che si occupa del Sitema Qualità e della sua gestione.

Ricordiamo a questo proposito che la norma ISO 9001, punto 4.1, prevede la figura del *rappresentante della direzione,* il quale è responsabile per la qualità nei confronti dell'utente finale e risponde solamente alla direzione generale.

Per adempiere a tutti i nuovi compiti relativi alla qualità, il *responsabile della qualità* necessita anche della collaborazione di almeno un assistente.

Ricordiamo che il piano della qualità deve contenere: i criteri d'ingresso uscita per ogni fase dello sviluppo; la pianificazione delle attività di prova, verifica e validazione; le responsabilità specifiche per i riesami e le prove, la gestione della configurazione, ed il controllo dei difetti e le azioni correttive.

Riesame del contratto

Riesame del contratto.

Nel processo di sviluppo, i punti di controllo sono le attività volte a determinare se sia possibile proseguire o se le fasi già svolte vadano meglio definite.

Il riesame del contratto è quindi il primo punto di controllo; avviene successivamente alle fasi di pianificazione e consiste in una o più riunioni. Il riesame viene condotto dalla direzione commerciale, in quanto riguardante le specifiche contrattuali concordate con il committente. L'analista di sistema partecipa in ragione della propria conoscenza di ulteriori dettagli progettuali. Il responsabile della qualità partecipa anch'egli, allo scopo di assicurare il rispetto delle procedure vigenti in azienda; ed infine la direzione generale ed il committente approvano la nuova stesura del contratto.

Progettazione preliminare

Progettazione preliminare.

Possiamo suddividere l'attività di progettazione in due fasi:

• *progettazione preliminare:* si concentra sulla trasformazione delle specifiche software nell'*architettura software* (la rappresentazione della struttura modulare gerarchica del sistema software) e nell'*architettura dei dati*;

• *progettazione dettagliata:* si focalizza sui raffinamenti della rappresentazione architetturale, che portano alla struttura dettagliata dei dati e delle rappresentazioni algoritmiche [Pressman, 1993].

L'attitività di *progettazione preliminare,* in EURIS, viene svolta solitamente dall'analista di sistema che ha redatto le specifiche funzionali del prodotto. Egli, inoltre, gestisce gli aspetti tecnici del progetto nel corso delle successive fasi dello sviluppo, coordinando le esigenze del committente di verifica e cambiamento dei requisiti funzionali con le esigenze di pianificazione e gestione delle risorse che invece ha il direttore della produzione. In tal modo l'analista di sistema assume anche il ruolo di *account manager;* tale ruolo sarà in futuro potenziato, includendo ad esempio compiti formalizzati di gestione della configurazione.

Riesame del progetto preliminare.

È questo il secondo punto di controllo, ed ha lo scopo di controllare la consistenza dei dettagli di progetto necessari alla produzione prima di procedere con la fase di progettazione dettagliata. Questo riesame viene condotto dalla direzione tecnica, con la partecipazione dell'analista di sistema che ha effettuato la progettazione e con la presenza del responsabile della qualità per verificare la rispondenza del progetto preliminare alle prescrizioni qualitative.

Progettazione dettagliata

Progettazione dettagliata.

In EURIS, la progettazione dettagliata viene svolta dai capi progetto della produzione. Essi hanno un doppio riferimento, l'analista di sistema per quanto riguarda gli aspetti tecnici e il direttore di produzione per quelli organizzativi. Il loro compito è quello di tradurre la progettazione preliminare in componenti software testati e nel rispetto dei tempi previsti.

Riesame del progetto dettagliato.

Il riesame viene svolto dall'analista di sistema responsabile della conduzione tecnica di quel progetto. L'assistente del responsabile della qualità presenzia per verificare il rispetto del piano di sviluppo e di quello della qualità approvati per quel progetto. Anche il direttore della produzione partecipa al riesame, in quanto responsabile dell'organizzazione e gestione del progetto stesso.

Codifica

Codifica.

La codifica e il test sono le attività che i programmatori effettuano sotto il controllo dei capi progetto. Il risultato di questa fase è un codice testato a livello di singola unità.

Inoltre, i capi progetto hanno un ruolo fondamentale nel mantenere la configurazione del software sotto controllo. Questo compito consiste nel controllo delle versioni di ogni singolo componente software e di come questi si integrano per formare una nuova versione di prodotto finale. L'approvazione delle

nuove versioni di ogni componente software (siano essi programmi o documentazione), viene fatta dall'analista di sistema responsabile della conduzione tecnica di quel progetto. Anche il direttore della produzione ha un ruolo in questa attività, ed è quello di coordinare la gestione della configurazione su più progetti.

Redazione dei piani dettagliati di prova.

Seguendo lo schema del piano di prova già presente nel piano della qualità, l'assistente del responsabile della qualità prepara i piani dettagliati di prova, che comprendono le prove di unità, d'integrazione e di sistema. La documentazione include anche i casi di prova, i dati di prova, i risultati attesi, gli ambienti e strumenti software di prova, ed i criteri per considerare le prove superate. L'assistente agisce con la supervisione del responsabile della qualità.

Prova e validazione

Prova e validazione.

La prova e validazione è un'attività di assicurazione della qualità essenziale. Questa, in EURIS, viene svolta in due fasi. Nella prima fase, il capo progetto, con la supervisione dell'assistente del responsabile della qualità, effettua le prove d'unità e d'integrazione prescritte nel piano di prova.

Successivamente, l'analista di sistema che ha redatto le specifiche funzionali e svolto la progettazione preliminare di quel prodotto effettua anche le prove di sistema e la validazione. La validazione ha lo scopo di verificare che il prodotto software nella sua versione completa sia conforme ai requisiti specificati, e quindi coinvolge anche il direttore commerciale in qualità di rappresentante del punto di vista del committente. Il compito del responsabile della qualità è quello di fornire una verifica indipendente della rispondenza del prodotto ai requisiti specificati.

Accettazione

Accettazione del prodotto software.

È questa in pratica un'attività di validazione, ma svolta dal committente e quindi con valore contrattuale. Le procedure per l'accettazione devono essere previste nel contratto, e devono venir seguite. L'eventuale introduzione di modifiche dei termini di accettazione rende consigliabile l'approvazione di questa fase da parte della direzione generale.

Misurazione

Misurazione.

La misurazione è un'attività importante prescritta dalla norma ISO 9000/3 (la sua esclusione dalla Fig. 3 è dovuta solamente alla necessità di renderne più agevole la lettura).

Essa avviene in corrispondenza delle attività di progettazione preliminare e dettagliata (numero di specifiche prodotte, tempo), codifica (LOC/tempo e documentazione/LOC), prova (difetti/LOC). La misurazione avviene anche dopo il rilascio (MTTF, MTTR, difetti/LOC).

In EURIS, per la raccolta dei dati indicati dal piano della qualità, sono stati incaricati i capi progetto. Successivamente, l'assistente del responsabile della qualità opera una sintesi di tali dati, allo scopo di renderli più significativi. Il compito del responsabile della qualità è infine quello di predisporre le procedure e gli strumenti per la misurazione, e quindi controllare che tutto proceda nella maniera stabilita.

Le metriche scelte sono per ora quelle più semplici di produttività, costo e qualità indicate al paragrafo 1. La normalizzazione avviene mediante l'utilizzo delle linee di codice (LOC), anche se si sta prendendo in considerazione il passaggio ai function points (FP).

Controllo della struttura del sistema qualità

Tutte le procedure viste sopra sono relative al ciclo di vita del prodotto. Oltre a queste vi sono altre procedure, relative invece al controllo della *struttura* del sistema qualità, e sono descritte anch'esse nel *manuale della qualità* già incontrato al paragrafo precedente.

Tra queste ultime, importanti sono quelle per le *verifiche ispettive interne del sistema qualità*. Tali ispezioni servono a verificare, su base periodica, l'efficacia di tutte le attività che influenzano la qualità. Le ispezioni sono svolte dal responsabile della qualità in collaborazione con un consulente esterno. L'approvazione comunque giunge dalla direzione generale.

Tre approcci all'assicurazione qualità nel software

In questo articolo abbiamo visto tre diversi tipi di approccio al problema dell'assicurazione della qualità del software: l'approccio teorico; l'approccio delle normative; l'esperienza concreta in azienda.

Approccio teorico

La trattazione teorica è incentrata soprattutto sulla qualità del prodotto software preso a sé stante. I modelli di qualità sono importanti per capire quali sono le caratteristiche importanti di un prodotto software, ma abbiamo visto che, dal lato pratico, mediante le metriche oggi a disposizione non è possibile ottenere delle misure univoche e complete, questo soprattutto per quanto riguarda le caratteristiche di qualità.

Approccio normativo

L'approccio delle normative si incentra principalmente sul processo di sviluppo del software. La qualità viene costruita attraverso il controllo del processo di sviluppo. A questo punto c'è un'ulteriore suddivisione: le normative di tipo "classico", nelle quali questo controllo è rigido e spinto fino a prescrivere, in alcuni casi, i formati della documentazione (ricordiamo che si tratta di norme tipicamente militari, ne è un esempio la normativa DOD); e la normativa ISO, che invece utilizza un tipo di controllo più flessibile. La normativa ISO, infatti, risulta applicabile indipendentemente dal modello di ciclo di vita utilizzato e pertanto non prescrive un dato processo di sviluppo.

Approccio progmatico

Infine, c'è l'esperienza concreta in azienda. Qui l'approccio è senz'altro più pragmatico. Ogni normativa, per quanto elastica come la ISO, introduce sempre degli elementi di rigidità che mal si sopportano in azienda. La necessità è perciò quella di rendere le procedure aziendali descritte nel manuale della qualità le più flessibili e nel contempo più rigorose possibile; non dimentichiamo che, in caso di certificazione del Sistema Qualità, ogni variazione delle procedure stesse deve essere notificata ed approvata dall'ente certificatore.

Il sistema qualità come strumento per il miglioramento continuo

Una variazione fisiologica delle procedure è comunque non solo ammessa, ma anche auspicabile, in quanto il Sistema Qualità non va concepito come un progetto a termine, ma come un processo evolutivo. Il miglioramento continuo richiede infatti avanzamenti di tipo sia metodologico che tecnologico.

Se il Sistema Qualità è ben concepito, il miglior controllo del processo di sviluppo non potrà che giovare alla qualità del prodotto, al servizio reso al cliente ed anche alla produttività dell'azienda stessa. □

Ringraziamenti
Si ringraziano Egisto Bressan (Amministratore Delegato del gruppo EURIS) ed il Sandro Vidmar (Direttore di produzione EURIS) per la disponibilità dimostrata e per il proficuo clima di confronto.

RIFERIMENTI BIBILOGRAFICI

Albrecht A.J., Gaffney J.E., «*Software Function, Source Lines of Code and Development Effort Prediction: A Software Science Validation*», IEEE Trans. Software Engineering, november 1983.

Battistella G., *La qualità nel software*, Franco Angeli, Milano, 1990.

Boehm B., *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

De Marco T., *Structured Analysis and System Specification*, Prentice-Hall, Englewood Cliffs, NJ, 1979.

Fenton N.E., *Software Metrics - A Rigorous Approach*, Chapman & Hall, London, 1991.

Gillies A.C., *Software Quality - Theory and Management*, Chapman & Hall, London, 1992.

Gunning R., *The Technique of Clear Writing*, McGraw-Hill, New York, 1968.

Kan S.H., Basili V.R., Shapiro L.N., «*Software Quality: An Overview From the Perspective of Total Quality Management*», IBM System Journal, vol.33, n°1, 1994.

Maiocchi M., *Il controllo di qualità del software*, Franco Angeli, Milano, 1988.

McCabe T.J., «*A Complexity Measure*», IEEE Trans. Software Engineering, april 1976.

McCall J.A., «*An Introduction to Software Quality Metrics*», in Cooper J.D., Fisher M.J. (eds.), Software Quality Management, Petrocelli Books, New York, 1979.

Musa J.D., Iannino A., Okumoto K., *Engineering and Managing Software with Reliability Measures*, McGraw-Hill, New York, 1983.

Nocentini S., *Il sistema di qualità del software*, Etas libri, Milano, 1993.

Perry W.E., *Effective Methods of EDP Quality Assurance*, Wellesley, Q.E.D. Information Sciences, 1981.

Pressman R.S., *A Manager's Guide to Software Engineering*, McGraw-Hill - System Design & Implementation Series, New York, 1993.

Price Waterhouse Consultants, *Software Quality Standards: The Costs & Benefits*, 1988.

Schulmeyer G.G., «*Software Quality Assurance: Coming to Terms*», in Schulmeyer G.G., McManus J.I. (eds.), Handbook of Software Quality Assurance, VanNostrand-Reinhold, New York, 1992.

Software Engineering Technical Committee of the IEEE Computer Society, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE-STD-729-1983, New York, 1983.

UNI EN 29000:1987, «*Regole riguardanti la conduzione aziendale per la qualità e l'assicurazione (o garanzia) della qualità - Criteri di scelta e di utilizzazione*», UNI - Ente Nazionale Italiano di Unificazione, dicembre 1987.

UNI EN 29001:1987, «*Sistemi Qualità - Criteri per l'assicurazione (o garanzia) della qualità nella progettazione, sviluppo, fabbricazione, installazione ed assistenza*», UNI - Ente Nazionale Italiano di Unificazione, dicembre 1987.

UNI ISO 9000/3:1991, «*Regole riguardanti la conduzione aziendale per la qualità e l'assicurazione della qualità - Guida per l'applicazione della ISO 9001 (UNI EN 29001) allo sviluppo, alla fornitura ed alla manutenzione del software*», UNI - Ente Nazionale Italiano di Unificazione, giugno 1991.