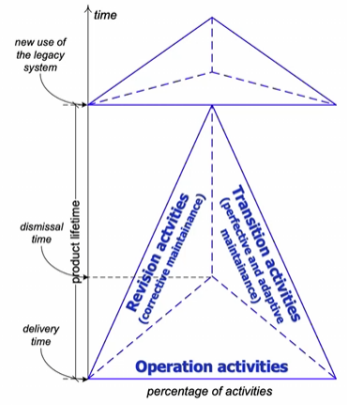
- ***Lezione 33***

  ➔ *Tredicesimo Blocco*

Passiamo ora all'argomento successivo direttamente correlato a quanto appena visto: la Qualità del Software.
Essa rappresenta per definizione il grado per cui il software possiede una serie di attributi "desiderabili".
La qualità del software può essere analizzata da vari punti di vista:

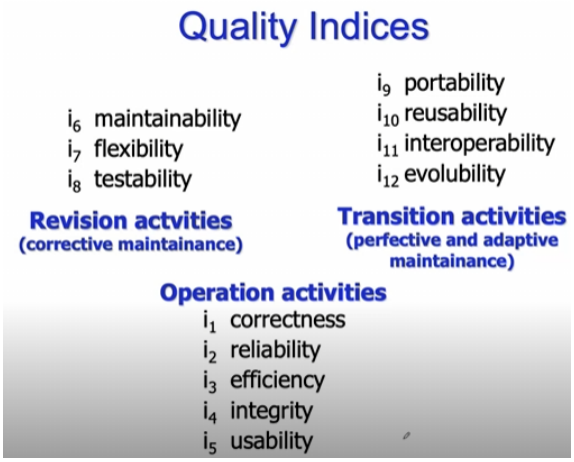
- Trascendentale: prodotto eccellente. Questo punto di vista l'abbiamo escluso fin dall'inizio, dal momento che l'obiettivo nello sviluppo software non deve essere tanto sviluppare un prodotto che sia eccellente quanto un prodotto che faccia correttamente quanto deve fare
- Utente: quanto il prodotto software contribuisca a raggiungere gli obiettivi dell'utente
- Prodotto: si valutano le caratteristiche del prodotto software (correttezza, affidabilità etc..) nonché la conformità con i requisiti, qui si fa riferimento alla qualità del software con la definizione introdotta poco fa dove la si vede come combinazione di queste caratteristiche "desiderabili", di qualità.
- Organizzazione: ci si focalizza sui benefici aziendali → costi e profitti (se il prodotto è facile da mettere sul mercato, quanto il prodotto permetta di incrementare i profitti/ridurre i costi etc…).

Vediamo ora come definire in modo quantitativo questa combinazione di attributi, nel tempo infatti per cercare di dare una valutazione oggettiva a questi attributi sono stati introdotti dei modelli di qualità che definiscono in modo preciso cosa si intende per qualità del software.
Importante introdurre questi modelli perché il concetto di qualità è un insieme di caratteristiche facilmente inquadrabili in modo soggettivo, ma si devono valutare in modo oggettivo per poter essere misurate e dare valutazione quantitativa.
In particolare ci soffermiamo sul Quality Model introdotto da McCall: il Quality Triangle.



Sulle ascisse la percentuale di attività che vengono svolte sul prodotto a partire dal momento in cui questo è immesso sul mercato. Sulle ordinate il tempo, che parte dal momento di

delivery (rilascio, momento in cui il software passa dallo stadio di sviluppo a quello di uso/manutenzione).
Dal momento del rilascio il software sarà usato da diversi attori in modi diversi: vi saranno gli utenti che svolgeranno le così dette Operation Activities. Mentre gli utenti usano il software gli sviluppatori avvieranno delle attività di monitoraggio e manutenzione pressoché continue per molti motivi (non solo correttivi!).
Infatti tra le attività di manutenzione le Revision Activities (manutenzione correttiva) e Transition Activities (manutenzione perfettiva e adattiva).
Ad un certo punto, quando gli utenti terminano di usare il software (Operation Activities), il software entra nello stadio di dismissione.
Come sappiamo questo stadio non è istantaneo e non rappresenta necessariamente la totale cestinazione del software, ma magari una sua rielaborazione (quindi ulteriori attività di sviluppo e manutenzione) per arrivare eventualmente a un nuovo utilizzo del sistema legacy (triangolo in alto).
Ciò che fa McCall, per ogni possibile attività di Revision, Transition o Operation, è definire degli attributi di qualità del software (detti nel caso di McCall Indici di Qualità).
Quindi qualità come combinazione di attributi desiderabili, nel caso di McCall come combinazione di 12 indici differenziati in base al tipo di attività.



Operation:

- i_1 Correttezza: rappresenta il grado con cui il prodotto soddisfa le specifiche e gli obiettivi degli utenti (quanto il software fa quello che l'utente vuole?)
- i_2 Affidabilità: grado con cui il software esegue le sue funzioni con la precisione richiesta (diverso dall'affidabilità classica che avevamo definito come la probabilità che il software funzioni correttamente in un certo intervallo di tempo)
- i_3 Efficienza: fa riferimento alle risorse di calcolo, quanto me serve potente il pc per eseguire il software
- i_4 Integrità: il software è sicuro? Si fa riferimento alla sicurezza del software in termini di accesso esterno indesiderato
- i_5 Usabilita: impegno richiesto per riuscire ad utilizzare il software da parte dell'utente

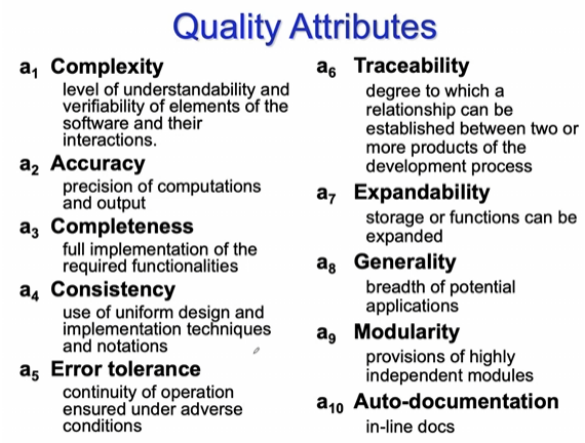
Revision:

- i_6 Manutenibilità: impegno necessario per individuare e correggere un difetto del programma
- i_7 Testabilità: impegno necessario per testare il prodotto e verificare che funzioni come desidero
- i_8 Flessibilità: impegno richiesto per modificare il prodotto (legato alla complessità del software stesso).

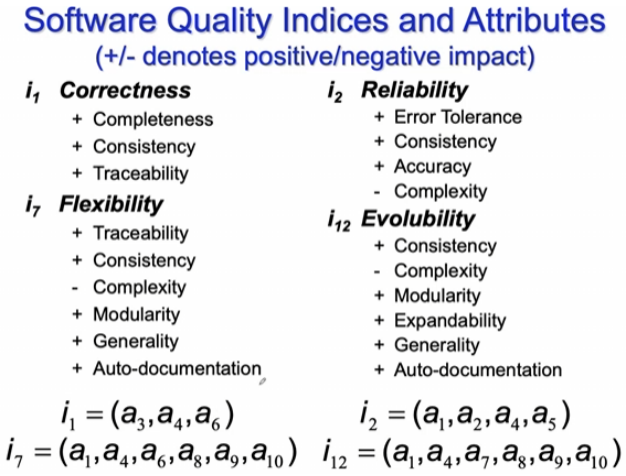
Transition:

- i_9 Portabilità: impegno richiesto per trasportare il prodotto da un ambiente operativo a un altro
- i_{10} Riusabilità: il grado con cui il prodotto o sue parti possono essere riusate in applicazioni differenti
- i_{11} Interoperabilità: quanto il software può essere accoppiato con altri prodotti
- i_{12} Evolubilità: effort richiesto per aggiornare il prodotto al fine di aggiornare eventuali nuovi requisiti (manutenzione perfettiva anche detta evolutiva)

Si ricorda che questi indici sono di alto livello, ciò che poi farà il modello di qualità è scomporli ulteriormente per arrivare a un livello tale da poter quantificare il valore dell'attributo. Per ora li stiamo definendo solo a livello astratto.
Il modello di qualità funziona come segue. Misuro questi 12 indici, che rappresentano le possibili caratteristiche che permetterebbero di definire oggettivamente la qualità del mio software. Per misurarli è necessario introdurre dei sottoindici, che mi permettano di arrivare ad un livello di dettaglio tale da poter quantificare nella pratica queste caratteristiche.
Secondo McCall in particolare la qualità del software è definita da un vettore contenente i 12 indici di qualità appena visti $q = (i_1, \dots , i_2)$.
Ciascun indice può essere misurato facendo riferimento ad un insieme di attributi di qualità (attento alla terminologia e quindi alla differenza tra Indice di Qualità e Attributo di Qualità per McCall). → $i_j = (a_1, a_2, \dots, a_n)$.
Un attributo può impattare anche più indici di qualità.
Gli attributi sono 10 :
Complessità come livello di comprensibilità, Accuratezza come precisione nel calcolo, Completezza come quanto il software ha implementato in modo completo le funzionalità richieste, Consistenza come utilizzo di approcci di design uniforme, Errror Tolerance come quanto il software riesce a "sopravvivere" ad eventuali malfunzionamenti e continuare a funzionare, Tracciabilità grado di relazione tra due o più prodotti nel processo di sviluppo (es. tra codice e documento requisiti etc..), Espandibilità come quanto il software può essere esteso in termini di storage e funzioni, Generalità come ampiezza di applicazioni potenziali, Modularità come grado di indipendenza tra moduli e Auto-documentation come quanto il software è in grado di aiutare l'utente attraverso un help in linea.



Quindi McCall definisce alcuni di questi attributi per ogni indice e misura tutti questi attributi secondo specifiche metriche. Vediamo un esempio per 4 indici



Si noti come gli attributi possono avere impatto positivo o negativo per gli indici. Ad es. per la flessibilità maggiore è la complessità, peggiore è l'indice di qualità!
In generale + se l'attributo influisce positivamente sull'indice, - altrimenti.
Per misurare gli attributi a loro volta sono necessari tipicamente dei sottoattributi per arrivare ad un'effettiva misura: es. per la modularità a_9 per misurarla si utilizzano tipicamente i sottoattributi come abbiamo visto nelle lezioni precedenti di coesione e coupling, morfologia e information flow (poi esistono anche altri modi, ogni modello usa il proprio e può far quindi riferimento a diverse metriche).
Per questi calcoli, che quindi possono avvenire usando diverse metriche, si introduce il Checklist Method, che mette a disposizione a chi deve valutare la qualità del software delle checklist che tipicamente pongono delle domande alle quali semplicemente l'utilizzatore deve porre una risposta.
Sarà a carico del metodo checklist, una volta fornite le risposte, calcolare il valore complessivo dell'attributo.
Una volta calcolati i valori dei vari attributi questi vengono raggruppati per ottenere il livello del corrispondente indice (tenendo ovviamente conto dell'impatto, se positivo o negativo).