Ingegneria del software
Introduzione

A. Gianoli

Univ. Ferrara - Corso di laurea in Informatica

Dove?

- Pressman, Cap. 1
- Sommerville, Cap. 1

Software engineering?

- Software engineering ≠ programming
- Software engineering ≠ computer science
- **Software engineering:** creare e mantenere applicativi software usando tecnologie e tecniche derivate dall'Informatica, dall'Ing. gestionale, e da altri campi.
- Software engineering riguarda il lavorare come gruppo, rispettando dei vincoli, per sviluppare un prodotto software per il cliente
- Software engineering è una disciplina

Perché è importante?

Due esempi:

- Nel 1996 Ariane 5 esplose dopo 40 sec dal lancio
  - costo totale: ~500M$ di satelliti, ~8000M$ di sviluppo del razzo
  - cause del fallimento: errori di specifiche e di design nel software del sistema inerziale
    http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html
- Therac-25: macchina per la radio-terapia controllata da computer
  - tra il 1985 e il 1987 sei pazienti ricevettero una overdose massiccia
  - cause del fallimento: cattivo design, impiego di soli sistemi software per evitare incidenti, ...
    http://sunnyday.mit.edu/papers/therac.pdf

FAQ sul Software Engineering

- Che cos'è il software?
- Quali sono gli attributi di un buon software?
- Che cos'è l'ingegneria del software?
- Qual'è la differenza tra ingegneria del software e informatica?
- Qual'è la differenza tra ing. del soft. e ingegneria dei sistemi?
- Cos'è il "processo di produzione del software"?
- Cos'è un "modello di produzione del software"?
- Quali sono i costi legati alla produzione di software?
- Quali sono le principali sfide che l'ing. del soft. deve affrontare?

Che cos'è il "software"?

- Un insieme di programmi di computer e la loro documentazione
  - configuration manual, users manual, ...

- **Generic** software
  Sviluppato per essere venduto a un ampio spettro di utenti
  La maggioranza del software

- **Custom** software
  Sviluppato per un utente specifico sulla base dei suoi bisogni
  Richiede più lavoro di sviluppo

Caratteristiche del Software

- Il Software è sviluppato, o è strutturato: non è "costruito" nel senso classico del termine
  - Lo sviluppo del software e lo sviluppo dell'hardware sono due cose ben distinte e diverse
- il software non si "consuma" (però può "invecchiare")
- rispetto all'industria in generale, l'industria del software produce una grossa quantità di prodotti custom (=software)

Il dilemma dello sviluppare software

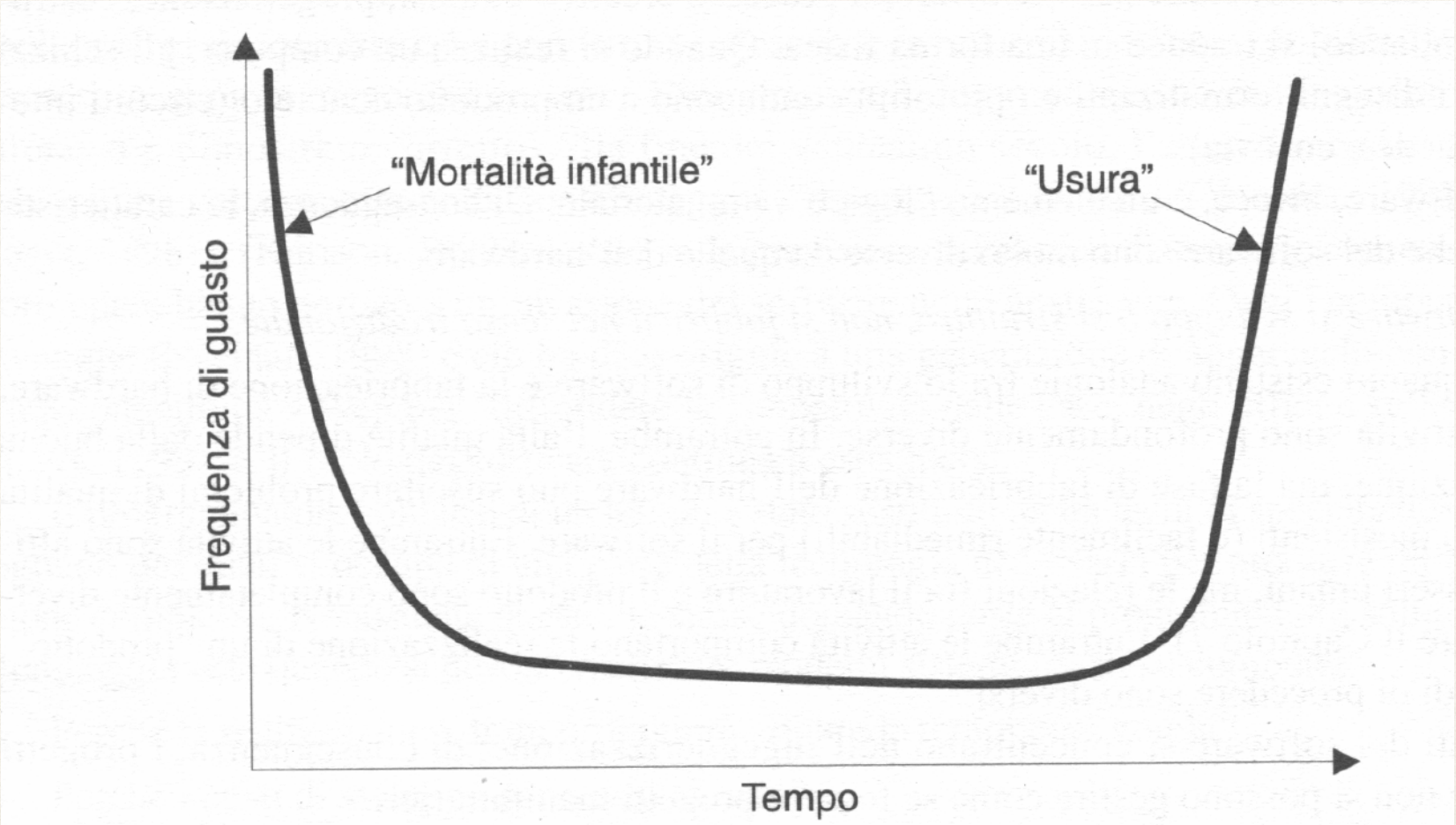
- I costi per sviluppare progetti software salgono mentre i costi dell'hardware scendono
- Il tempo necessario a sviluppare software si allunga e i costi per mantenere il software aumentano
- Gli errori del software aumentano mentre i guasti dovuti all'hardware scendono (mooolto bassi se si sceglie bene l'hardware)

La dura verità

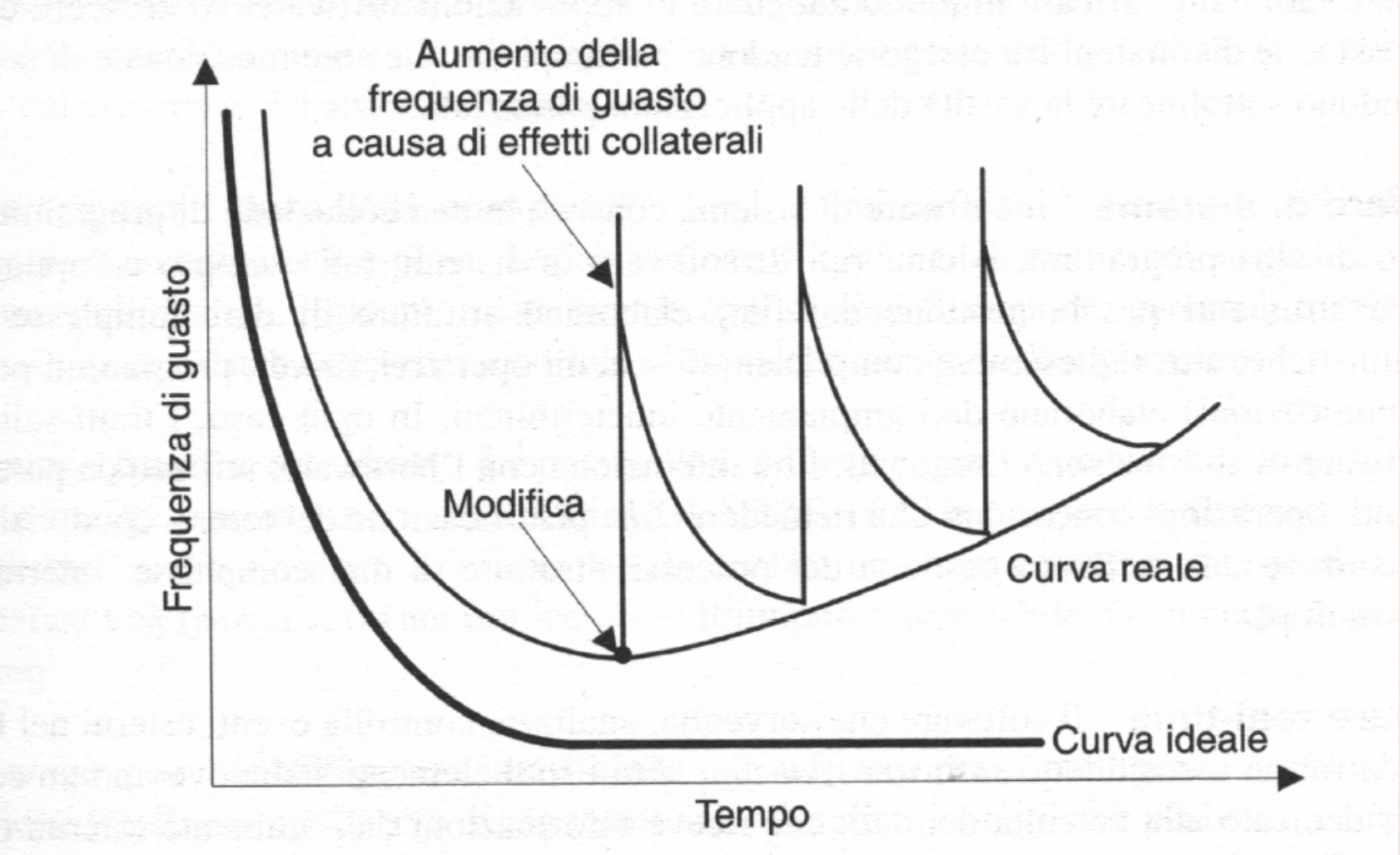
Q: How many programmers does it take to change a lightbulb?

A: We looked at the light fixture and decided there's no point trying to maintain it. We're going to rewrite it from scratch. Could you wait two months?

“Guasti” Hardware



“Guasti” Software



Attributi essenziali di un software di qualità

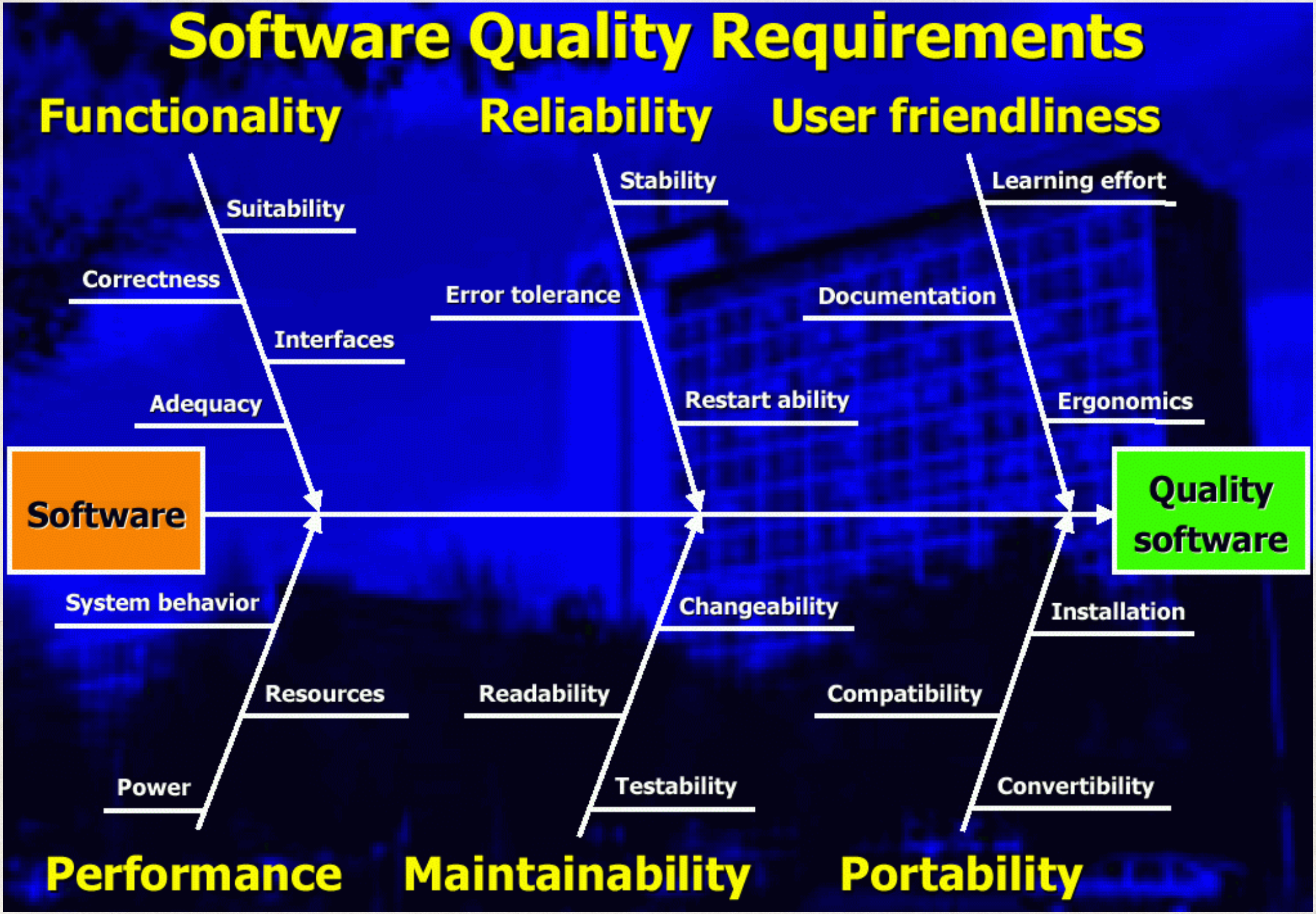
Le qualità su cui si basa la valutazione di un software possono essere

- interne: riguardano caratteristiche legate alle scelte implementative e non sono visibili agli utenti finali;
- esterne: riguardano funzionalità fornite dal sistema e sono visibili agli utenti finali

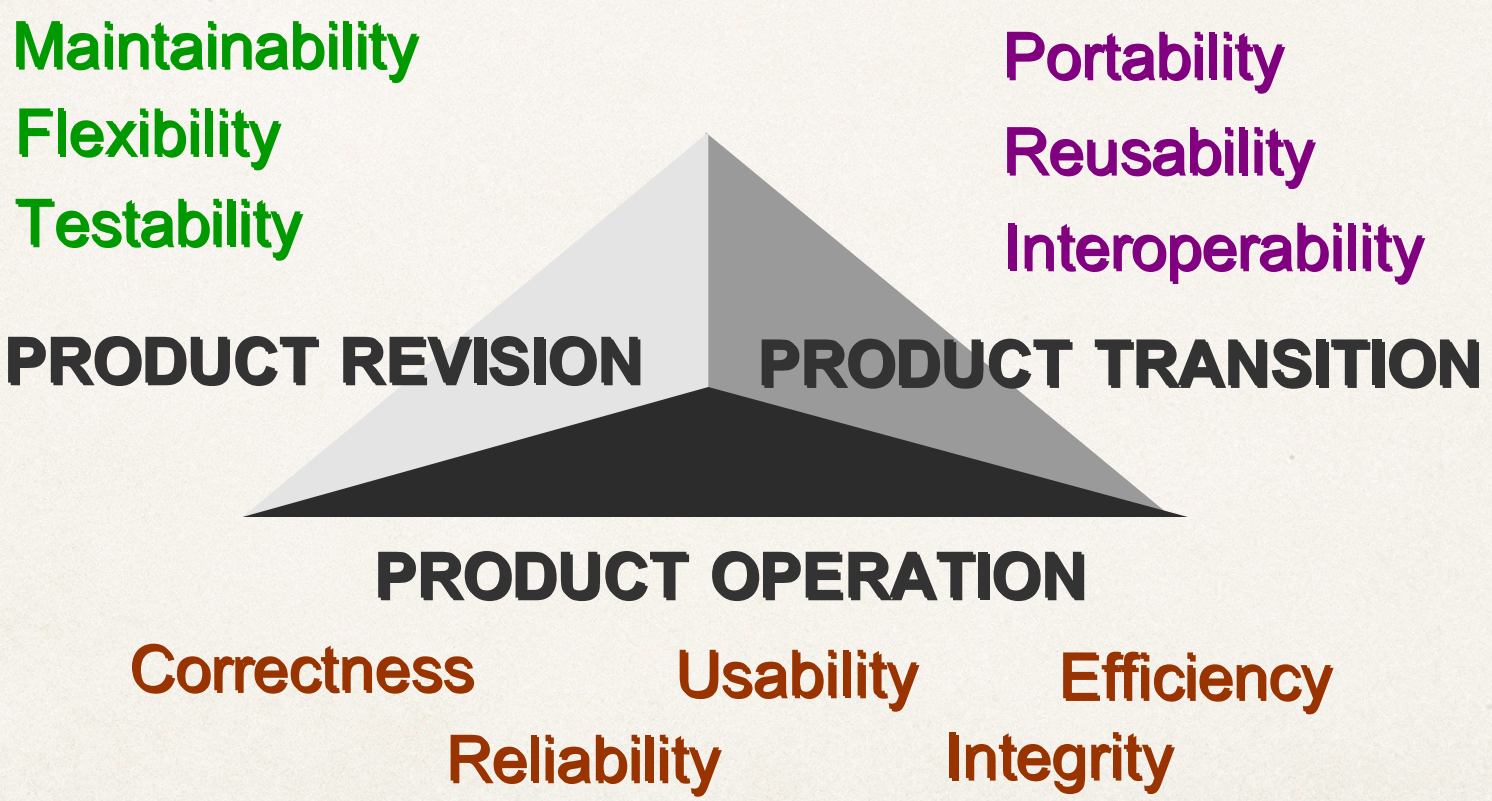
Le due categorie sono collegate: in generale non è possibile ottenere qualità esterne se il software non gode di qualità interne

Attributi essenziali di un software di qualità

- Maintainability:
  sw deve essere scritto in modo da poter evolvere seguendo i cambiamenti dei requisiti. È un punto fondamentale: la possibilità di modificare il sw è requisito inevitabile in un business in evoluzione
- Dependability (e security):
  insieme di caratteristiche, tra cui reliability, security, safety. Dependable sw deve non causare danni fisici o economici nell'eventualità di un system failure. Malicious users non devono poter accedere o danneggiare il sistema
- Efficiency:
  sw non deve sprecare risorse (memoria, cicli cpu,..). Stiamo parlando di responsiveness, processing time, memory utilization, ...
- Acceptability:
  sw deve essere accettabile dagli utenti per cui è stato sviluppato: deve essere comprensibile, usabile e compatibile con gli altri sistemi che usano


Software Quality Requirements
Functionality
Reliability
User friendliness
Suitability
Correctness
Interfaces
Adequacy
Stability
Error tolerance
Restart ability
Learning effort
Documentation
Ergonomics
Software
Quality software
System behavior
Resources
Power
Changeability
Readability
Testability
Installation
Compatibility
Convertibility
Performance
Maintainability
Portability

Il triangolo di McCall



Cos'è l'ingegneria del software?

Definizione IEEE

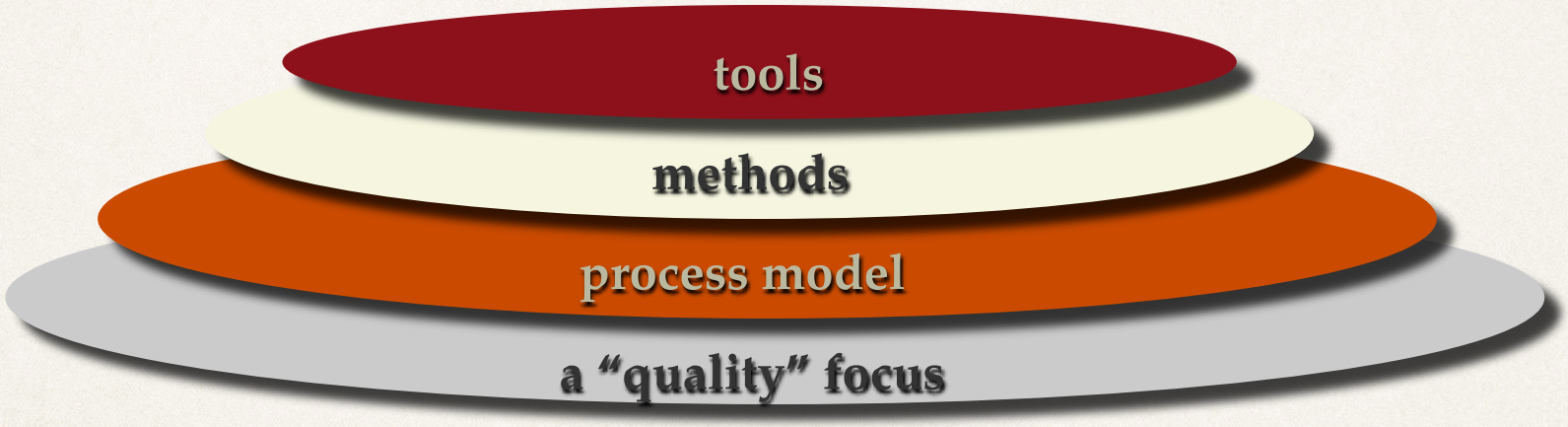
"The application of systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software;
that is,
the application of engineering to software."

Cos'è l'ingegneria del software?

- Software engineering è una disciplina ingegneristica che riguarda tutti gli aspetti della produzione di software, dai primissimi stadi delle specifiche di sistema fino alla manutenzione del sistema una volta che è in uso

- ha collegamenti con vari campi:
  - computer science (algoritmi, struture dati, linguaggi, tools)
  - business management (project mgmt, scheduling)
  - comunicazioni (gestire le relazioni con i "stakeholders": clienti, management, sviluppatori, testers, venditori)
  - psicologia/sociologia (personalità, stili, usabilità,...)
  - arte (GUI design, cosa è "appealing" per l'utente finale)

Vista a layers dell'ingegneria del software



Definizione dei layer

- Il layer dei processi è la base per la gestione di ogni progetto software.

"Un processo definisce chi fa cosa,
e quando e come si raggiunge un certo obiettivo."

- Il layer dei metodi descrive come all'interno di un singolo processo si realizzano i singoli passi che lo compongono.

- Il layer dei tools descrive strumenti che supportano lo sviluppo delle attività.

Ingegneria del software e informatica

- L'informatica è una scienza: il "cuore" sono i fondamenti teorici: linguaggi – algoritmi – complessità – formalismi ecc.
- L'ingegneria del software ha a che fare con aspetti più "pratici": come pianificare e sviluppare la produzione di software di qualità.
- Ad un ingegnere del software le conoscenze di base dell'informatica servono quanto la fisica ad un ingegnere elettrico

Ingegneria del software e ingegneria di sistema

- L'ingegneria di sistema ha come oggetto tutti gli aspetti dello sviluppo di un sistema basato su computer, inclusi gli aspetti hardware, software e di processo.
- L'ingegneria del software può essere vista come una parte dell'ingegneria di sistema.
- Gli ingegneri del software collaborano
  - alla specifica del sistema,
  - alla progettazione architetturale
  - all'integrazione con le altre componenti.

Il Processo di produzione del software

- Il processo di produzione software è un insieme di attività il cui fine è lo sviluppo oppure la modifica di un prodotto software

- Attività generiche di tutti i processi di produzione del software:
  - Specifica: cosa deve fare il sistema e quali sono i vincoli per la progettazione
  - Sviluppo: produzione del sistema software
  - Validazione: verifica che il software faccia ciò che il cliente richiede
  - Evoluzione: modificare il software in base alla modifica delle esigenze

Problemi nel processo di sviluppo software

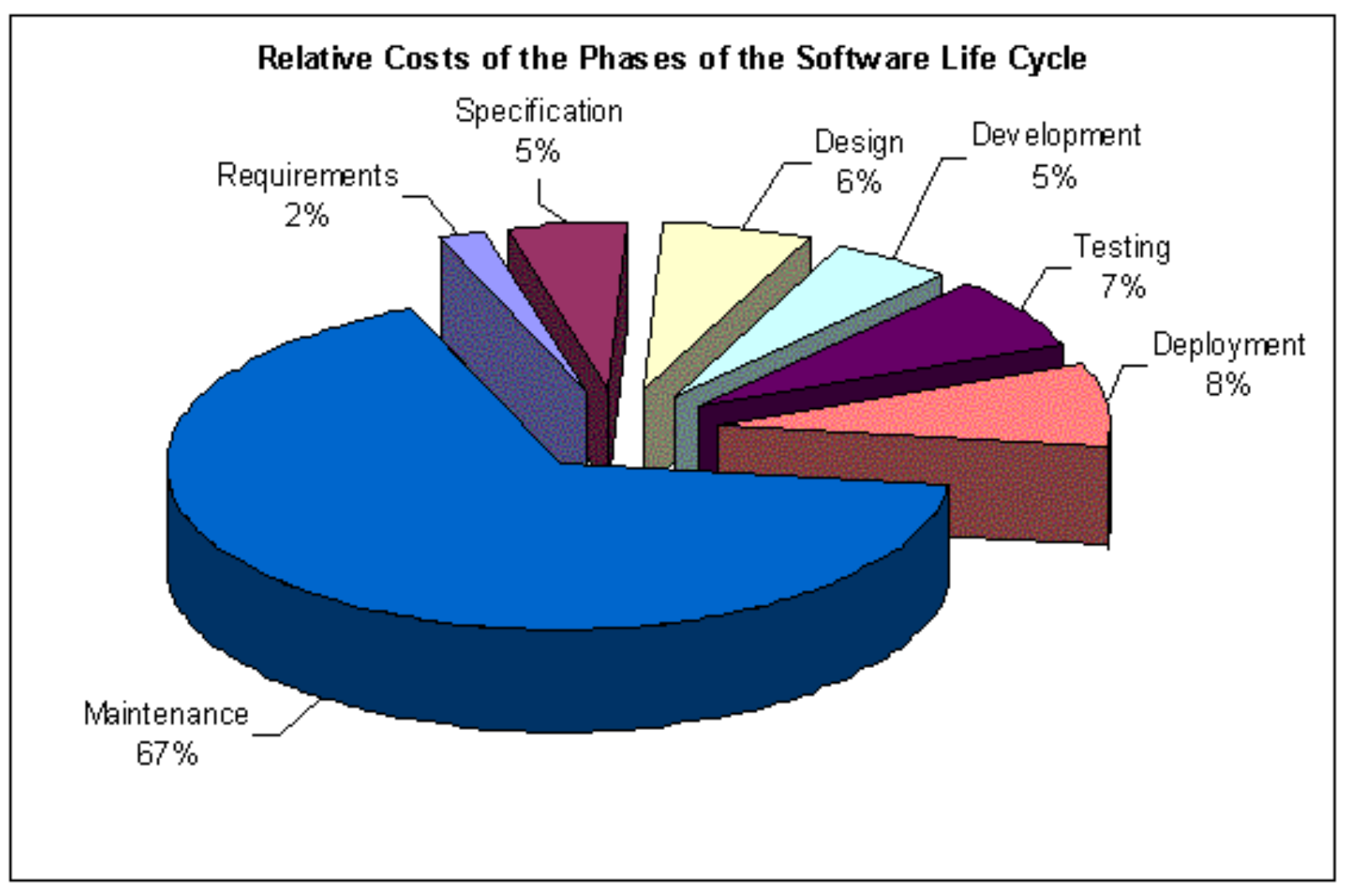
- Specifiche incomplete/incoerenti
- Mancanza di distinzione tra specifica, progettazione e implementazione
- Assenza di un sistema di validazione
- Il software non si consuma: la manutenzione non significa riparare alcune componenti “rotte”, ma modificare il prodotto rispetto a nuove esigenze

Costi legati alla produzione di software

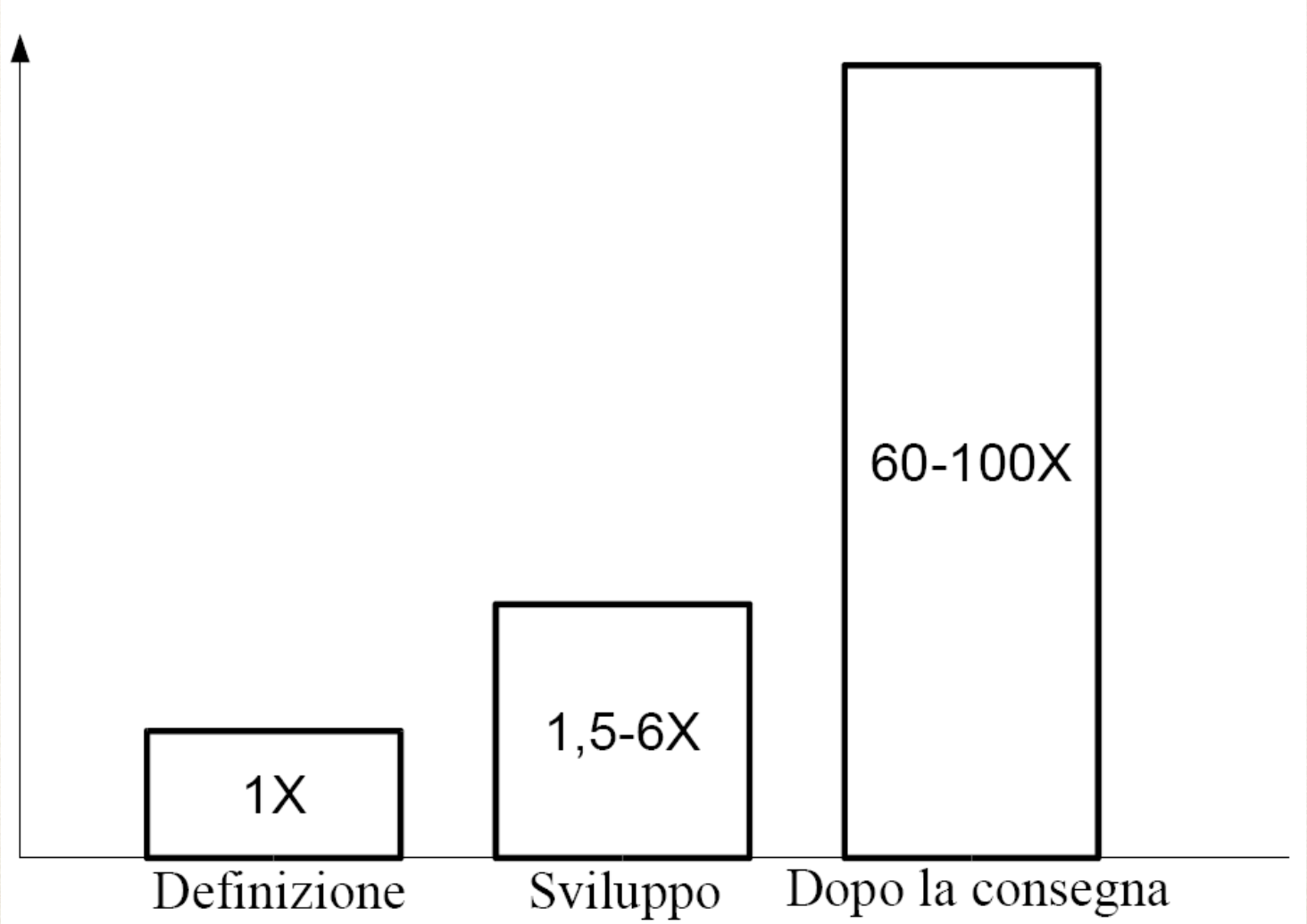
- All'incirca il 60% dei costi è legato allo sviluppo, il 40% sono costi per la verifica e validazione (testing).
- I costi variano a seconda del tipo di sistema che deve essere sviluppato e da requisiti quali la performance o l'affidabilità del sistema.
- La distribuzione di costi nelle varie fasi del processo di produzione del software dipende dal modello di processo.

➡ N.B. l'80% degli errori è fatta durante la raccolta dei requisiti o la fase di design!

Costi relativi all'intero ciclo di vita del sw



Costo di una modificare



Forse ce la facciamo?

- 3 categorie di progetti software
  - success: progetto terminato in tempo, secondo i costi, e funziona
  - challenged: progetto non completamente funzionale, in ritardo e sforati i costi
  - impaired: progetto cancellato durante lo sviluppo (spesso a causa costi o tempi)

Distribuzione: success=16,2% challenged=52,7% impaired 31,1%

Q: How many programmers does it take to change a lightbulb?
A: Five. Two to write the specification program, one to screw it in, and two to explain why the project was late.

Metodi dell'Ingegneria del Software

- Approcci strutturati di sviluppo software che includono:
  - Descrizione di Modelli
    - descrizione di modelli che devono essere prodotti
  - Regole
    - Vincoli applicati ai modelli del sistema
  - Suggerimenti di design
    - suggerimenti su buone pratiche di design del codice
  - Guide al processo di sviluppo
    - Quali attività seguire

Quali sono le sfide da affrontare?

- Legacy systems
  - sistemi vecchi ma tuttora molto utilizzati che devono essere mantenuti e aggiornati
- Eterogeneità
  - sistemi distribuiti, che includono una varietà di componenti hardware e software diversi
- Tempi di consegna
  - pressione sempre maggiore per ottenere software di qualità in tempi sempre più rapidi

Miti da sfatare

- Mito del management #1

"Se sforiamo i tempi previsti, basta aggiungere programmatori e ce la faremo!"

Non basta aggiungere risorse ad un progetto per accelerarne la realizzazione.

Lettura consigliata:

Frederick Brooks, *The Mythical Man-Month*

"Adding manpower to a late project makes it later"

Miti da sfatare

1 programmatore per 3 anni

e' equivalente a

3 programmatori per 1 anno

"The bearing of a child takes 9 months, no matter how many women are assigned."

Brooks, *Mythical Man-Month*

Miti da sfatare

- Miti del management #2

"Abbiamo standard e procedure da seguire nello sviluppo. Non serve altro"

- Gli standard sono applicati? I programmatori li conoscono?
- La qualità di processo è condizione necessaria ma non sufficiente alla qualità di prodotto

Miti da sfatare

Miti del management #3

"Abbiamo i più moderni sistemi di sviluppo e i computer più recenti"

- Gli strumenti (hw e sw) sono importanti ma gli sviluppatori lo sono di più. Sviluppare non è una attività facilmente automatizzabile per cui investire negli strumenti non è sufficiente

Miti da sfatare

- Miti del cliente #1

"Se sappiamo a grandi linee cosa deve fare il sistema, possiamo già iniziare a programmare"

Avere chiari gli obiettivi è un buon inizio, ma l'attività di codifica inizia molto più tardi!

Miti da sfatare

- Miti del cliente #2

> *"I requisiti variano continuamente, ma i cambiamenti si gestiscono facilmente perché il software è flessibile"*

- La modifica delle specifiche è tanto più difficile/costosa quanto più tardi avviene nella fase di sviluppo

Miti da sfatare

Miti del programmatore #1

"Una volta messo in opera il programma, il nostro lavoro è finito"

- Scrivere programmi è una parte piccola dell'intero processo
- Una buona parte del lavoro avviene dopo la consegna della prima versione al cliente

Miti da sfatare

- Miti del programmatore #2

"Fino a quando il programma non gira
non c'è modo di valutarne la qualità"

- L'attività di revisione durante il processo di progettazione è una condizione indispensabile per garantire la qualità del prodotto

Miti da sfatare

Miti del programmatore #3

"L'ingegneria del software ci farà scrivere una inutile e voluminosa documentazione e ci rallenterà"

- Lo scopo dell'ingegneria del software è creare qualità fin dall'inizio del processo di sviluppo, per avere benefici a lungo e medio termine.
- "Good, fast, cheap..... choose two"

Responsabilità professionale

- Oltre agli aspetti tecnici ci sono anche risvolti etici, sociali e legati alla responsabilità professionale
  - confidenzialità
  - competenza
  - diritti di proprietà intellettuale
  - uso inappropriato dei computer
- potete cercare “ACM/IEEE Code of Ethics”

Concludendo

- Ingegneria del software == disciplina metodologica che studia principi, tecniche, metodi e strumenti che supportano il processo di produzione del software