Ingegneria del Software (A.A. 2024/2025)

- *Docente*:

  Prof. Andrea D'Ambrogio

- *Obiettivi*:
  - fornire i metodi e le tecnologie per inquadrare la produzione del software all'interno di una disciplina ingegneristica
  - presentare il processo software e le più moderne tecniche di produzione
- *Esami*:
  - 2 appelli a fine di ogni semestre
  - 2 appelli a settembre
- *Testo consigliato*:
  - I. Sommerville, *Software Engineering*, Addison-Wesley (anche in italiano)
- *Materiale didattico*: distribuito su piattaforma **MS Teams**

SwEng: Unconsummated Marriage

- Software Engineering
  - disciplina per la produzione del software secondo i principi dell'ingegneria (progettazione e validazione)
  - essenziale per fare del sw un prodotto industriale

- Se manca si incorre in
  - scarsa qualità del prodotto
  - scarsa competitività
    - cost overrun
    - time overrun

SwEng: Unconsummated Marriage

- Sw Eng disciplina giovane …..
  - per anni i costruttori di Hw hanno visto la produzione di sw come attività banale, simile a USO del calcolatore, che richiede principalmente **abilità**
  - per anni l'abilità programmativa, la conoscenza delle ultime novità su linguaggi, interfacce etc., è stata considerata sufficiente a fare un ingegnere del sw
  - per anni la Sw Eng è stata considerata una branca della teoria della programmazione (o informatica teorica)

SwEng: Unconsummated Marriage

- Matrimonio non consumato……
  - quello tra la teoria della programmazione e i principi dell'ingegneria (progettazione e validazione) (D.L.Parnas, CACM, Sept. 1997)

- Cose da far sposare
  - ingegneri conoscano bene la teoria della programmazione
  - informatici teorici conoscano bene i principi dell'ingegneria

SwEng: Unconsummated Marriage

- Esempio: ingegneria chimica
  - matrimonio tra chimica e ingegneria (termodinamica, meccanica, dinamica fluidi etc.)
  - nessuno considera più l'ingegneria chimica come branca della chimica

- SwEng, termine coniato oltre 50 anni fa
  - conferenza NATO, Garmisch, Germania 1968
  - per testimoniare l'esigenza che il software fosse inquadrato all'interno di una disciplina ingegneristica.

SwEng: Unconsummated Marriage

- Risultati del ’68
  - l’attività della programmazione non è né una scienza né una matematica. Ciò perché il programmatore non aggiunge conoscenza a conoscenza, bensì costruisce un PRODOTTO
  - gli ingegneri devono basare sulla teoria della programmazione i loro principi di progettazione e convalida dei prodotti software
  - i problemi e i rischi connessi alla produzione e all’uso del software (bassa qualità, time e cost overrun) sono tipici dei prodotti costruiti da persone NON QUALIFICATE o, meglio, EDUCATE PER ALTRE PROFESSIONI

Aspetti tipici dell'Ingegneria del Sw (1)

- ACCIDENTALI del prodotto sw
  (superabili col progresso della tecnologia)

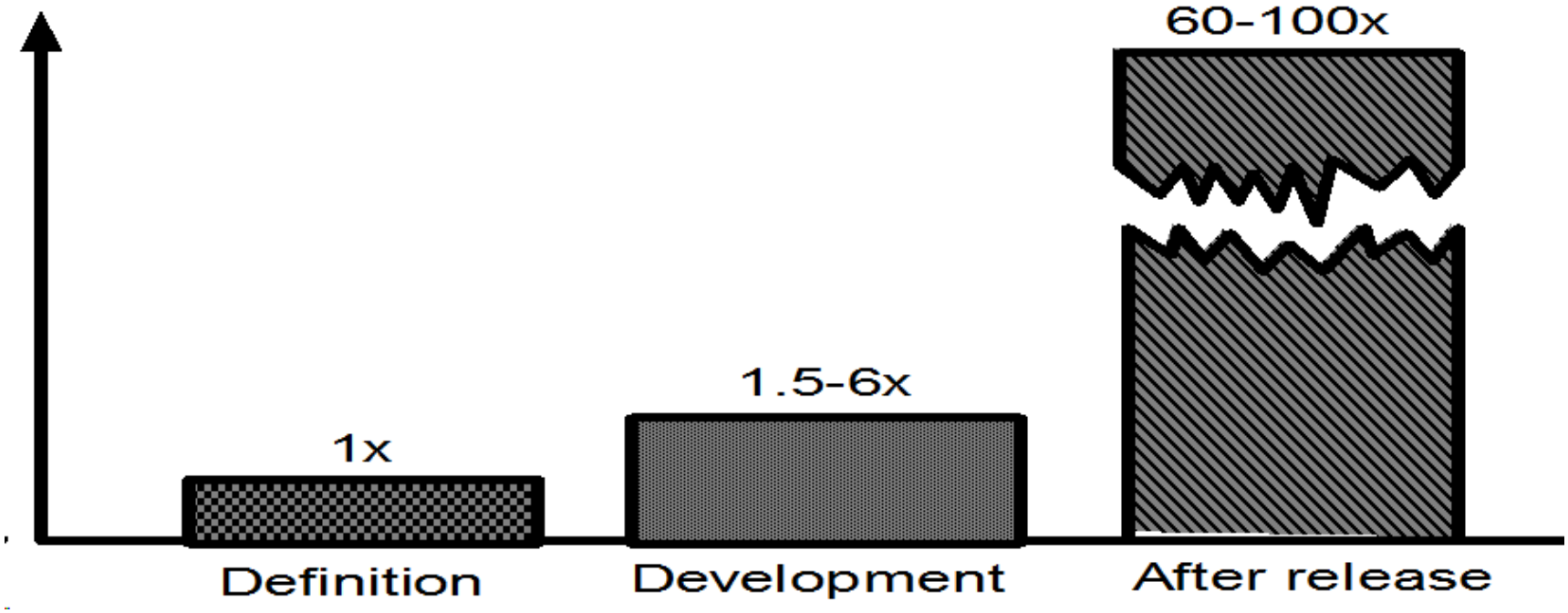
- di attitudine
- di manutenzione
- di specifica e progetto
- di teaming

Ciclo di vita del Sw = 3 Stadi, 6 Fasi

- Produzione Sw = sviluppo + manutenzione
- Sviluppo (stadio1) = 6 fasi
  1. Requisiti
  2. Specifiche (o analisi dei requisiti)
  3. Pianificazione
  4. Progetto (preliminare e dettagliato)
  5. Codifica
  6. Integrazione
- Manutenzione (stadio2)
  - copre circa il 60% dei costi del ciclo di vita
- Dismissione (stadio3)

L'effetto delle modifiche

- L'effetto delle modifiche varia secondo la fase in cui vengono introdotte.
- In fasi avanzate, una modifica può comportare rivolgimenti che richiedono nuove risorse o correzioni importanti al progetto, cioè costi supplementari



Dov'è il Testing?

- Non esplicitamente menzionato tra le 6 fasi
- Non è una fase separata
- E' un'attività che ha luogo durante l'intero sviluppo
- In due modi:
  - **Verifica (alla fine di ogni fase)**
  - **Validazione (alla fine dello sviluppo)**
- Verifica = la fase è stata ben svolta? (*are we building the product right?*)
- Validazione = il prodotto finale è buono? (*are we building the right product?*)

Defect Removal Efficiency (DRE)

- Fa riferimento alla percentuale di difetti trovati prima del rilascio del prodotto software
- Se il team di sviluppo trova 900 difetti prima del rilascio e gli utenti trovano 100 difetti in un intervallo temporale standard a partire dalla data di rilascio (tipicamente 90 giorni) allora il valore di DRE è pari al 90%
- In base a statistiche aggiornate al 2016, il *DRE medio* negli Stati Uniti è pari al 92% (i valori cambiano in base al modello di ciclo di vita)

Aspetti tipici dell'Ingegneria del Sw (2)

- ESSENZIALI del prodotto sw

  (non superabili col progresso dei mezzi e conoscenze)

- complessità
- conformità
- cambiabilità
- invisibilità

Aspetti tipici dell'Ingegneria del Sw (3)

DI COSTO del prodotto sw

- costo verso dimensione (size)
- costo verso repliche
- costo verso ampiezza di mercato

Aspetti di Costo

- *Costo* proporzionale al quadrato del *size* ($C=aS^2$)
  - fare due prodotti di size S/2 costa meno che farne uno di size S
- Produrre una replica non costa niente
- Vendere un *prodotto* di size doppio *per il mercato*
  - richiede un *prezzo* 4 volte superiore a parità di (ampiezza di) mercato
  - richiede un mercato (di ampiezza) 4 volte maggiore a parità di prezzo

Definizioni (1)

- *Prodotto Sw* (o brevemente *Sw*) =
  = Codice + Documentazione
- *Artefatto*= prodotto Sw *intermedio*
  - documento requisiti
  - documento di specifica
  - documento di progetto
- *Codice* = prodotto Sw *finale*
- *Sistema Sw* = insieme organizzato di prodotti Sw

Definizioni (2)

- *Cliente* = soggetto che ordina il prodotto Sw
- *Sviluppatore* = soggetto che lo produce
- *Utente* = soggetto che lo usa

- Sw *interno* = cliente e sviluppatore coincidono
- Sw *a contratto* = cliente e sviluppatore sono soggetti differenti

Aspetti di Affidabilità (Sw Reliability)

- Informalmente
  - credibilità del prodotto software
- Formalmente
  - probabilità che il prodotto software lavori "correttamente" in un determinato intervallo temporale

Difetto, Guasto, Errore

- **Difetto (defect)**
  - anomalia presente in un prodotto Sw
- **Guasto (failure)**
  - comportamento anomalo del prodotto Sw dovuto alla presenza di un difetto
- **Errore**
  - azione errata di chi (per ignoranza, distrazione, etc) introduce un difetto nel prodotto Sw

Affidabilità Sw

- Intuitivamente:
  - Un prodotto software con molti difetti è poco affidabile.
- E' chiaro che:
  - L'affidabilità del prodotto migliora via via che si riduce il numero di difetti

Caratteristiche dell'affidabilità Sw (1)

- Relazione non-semplice tra:
  - affidabilità osservata
  - e numero di difetti latenti
- L'eliminare difetti dalle parti del prodotto raramente usate
  - Ha piccoli effetti sull'affidabilità osservata.

La regola 10-90

- Esperimenti condotti su programmi di notevoli dimensioni mostrano che:
  - Il 90% del tempo di esecuzione totale è speso eseguendo il solo 10% delle istruzioni
- Detto 10% è chiamato :
  - core (nucleo) del programma

Caratteristiche dell'affidabilità Sw (2)

- Il miglioramento dell'affidabilità per l'eliminazione di un difetto:
  - dipende dalla localizzazione del difetto (ovvero se appartiene o meno al nucleo del programma)

Caratteristiche dell'affidabilità Sw (3)

- Dunque, l'affidabilità osservata dipende da:
  - come è usato il prodotto
  - in termini tecnici, dal suo profilo operativo (operational profile)

Caratteristiche dell'affidabilità Sw (4)

- Dunque, poiché utenti differenti usano il software secondo profili operativi diversi:
  - I difetti che si manifestano per un utente
    - potrebbero non manifestarsi per l'altro
- Dunque, l'affidabilità di un prodotto Sw:
  - Dipende dall'utente

Confronto tra affidabilità Hw e Sw (1)

- **I guasti Sw:**
  - sono dovuti alla presenza di difetti nei programmi
  - il software non si consuma

- **I guasti Hw son quasi sempre dovuti a:**
  - consumo/deterioramento dei componenti
  - qualche componente non si comporta più come specificato
  - qualche componente si rompe

Confronto tra affidabilità Hw e Sw (2)

- Esempi di difetti Hw
  - un resistore si altera
  - un condensatore va in corto
  - una porta logica si blocca su 1 oppure 0
- Per riparare un difetto hw:
  - si sostituisce il componente

Confronto tra affidabilità Hw e Sw (3)

- I difetti Sw sono latenti
  - il sistema Sw continua a guastarsi
    - a meno che non si effettuino le dovute correzioni

Confronto tra affidabilità Hw e Sw (4)

- A causa della differenza negli effetti dei difetti:
  - Le metriche usate per l'affidabilità Hw
    - Non sono estensibili al Sw

Confronto tra affidabilità Hw e Sw (5)

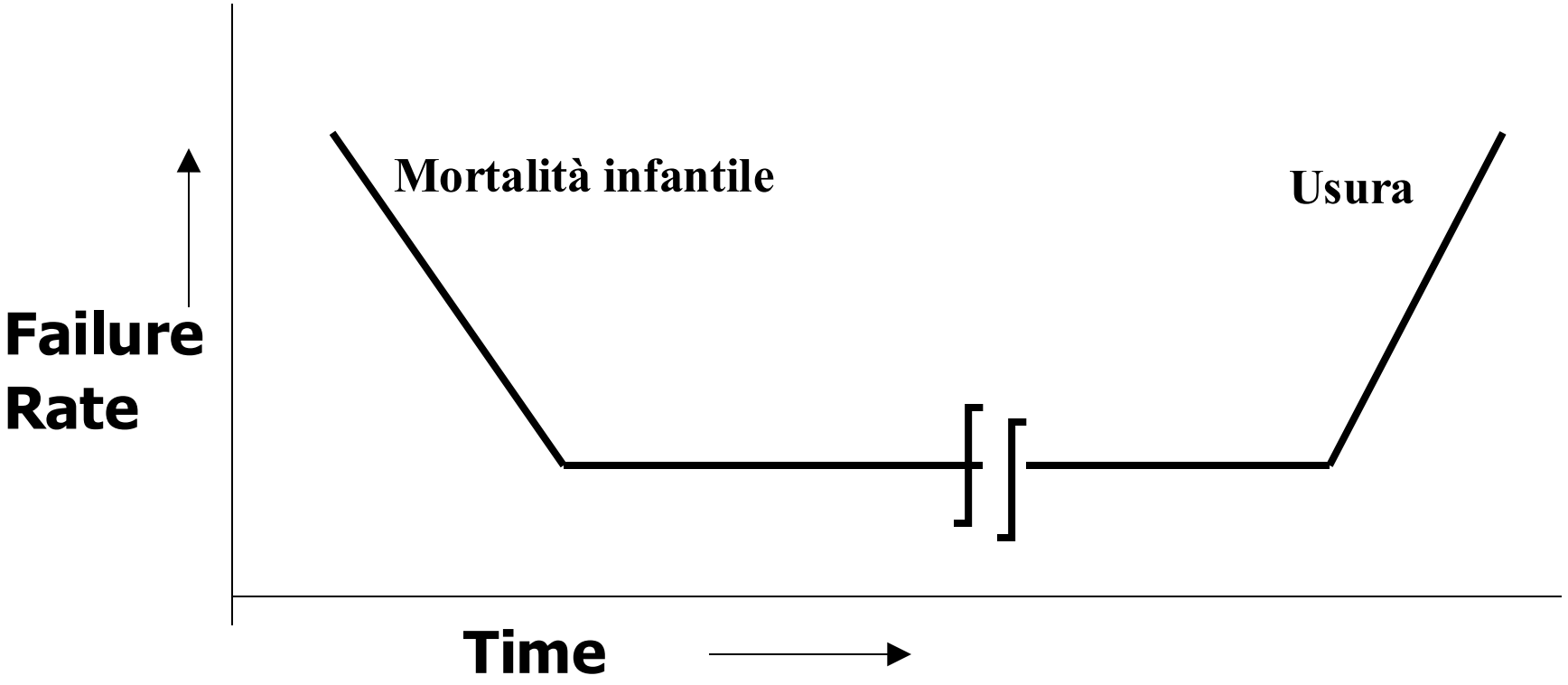
- Dopo la riparazione dell'Hw
  - la sua affidabilità torna come era
- Dopo la riparazione del Sw:
  - la sua affidabilità può aumentare o diminuire.

Confronto tra affidabilità Hw e Sw (6)

- Obiettivo dell'affidabilità Hw :
  - stabilità (cioè tenere la frequenza di guasto costante)
- Obiettivo dell'affidabilità Sw:
  - crescita di affidabilità (cioè far decrescere la frequenza di guasto)

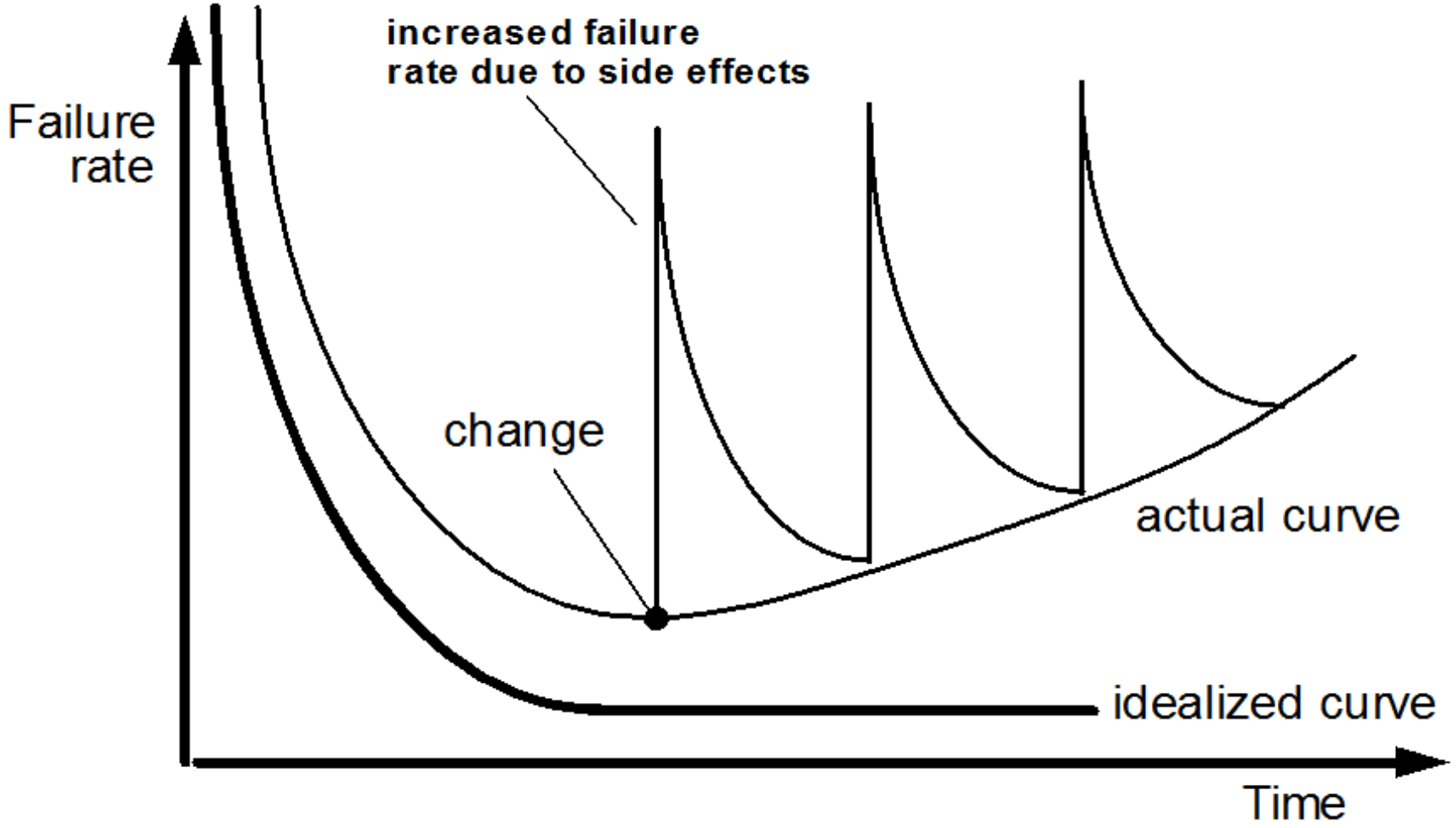
Nella realtà: andamento frequenza di guasto hardware

(effetto dell'eliminazione dei componenti difettosi prima, e dell'usura poi)



Andamento frequenza di guasto software

(effetto dell'eliminazione dei difetti prima, e dell'invecchiamento per manutenzione poi)



Disponibilità (Sw Availability)

- % del tempo che il Sw è risultato usabile nel corso della sua vita
- Dipende
  - dal numero di guasti che si verificano
  - dal tempo necessario a ripararli

Importanza di Sw Reliability/Availability

- Metriche importanti per sistemi in cui
  - la caduta del servizio crea cadute di efficienza e sicurezza (perdite economiche e sociali)
    - sistemi di trasporto
    - di governo del traffico aereo
    - di governo del volo
    - di produzione e distribuzione di energia
    - di comunicazione
    - etc

Conclusioni (1)

- Nel corso degli anni la produzione del software ha seguito varie fasi:
  - fase **di abilità**, nella quale prevalgono gli aspetti di lavoro individuale e creativo
  - fase **artigianale**, nella quale il software viene prodotto da piccoli gruppi specializzati, spesso di alto livello di professionalità
  - fase **industriale**, nella quale l'attività di sviluppo e manutenzione del software viene pianificata e coordinata, ed il lavoro del progettista viene sempre più supportato da strumenti automatici.

Conclusioni (2)

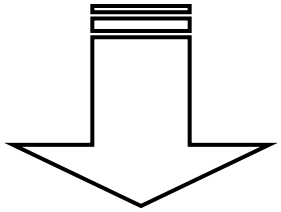
- Il termine «ingegneria del software» viene coniato per la prima volta nel 1968 in una conferenza NATO a Garmisch (Germania) per testimoniare l'esigenza che il software fosse inquadrato all'interno di una disciplina ingegneristica.
- Lo standard IEEE Std. 610.12 (1990) ha formulato una definizione più completa:
  1. Applicazione di un approccio sistematico, disciplinato e misurabile allo sviluppo, esercizio e manutenzione del software, cioè applicazione di principi ingegneristici al software
  2. Studio degli approcci di cui al punto 1

Conclusioni (3)

- Il software può essere considerato come un insieme di elementi che formano una "configurazione" che include:
  - programmi
  - documenti
  - dati multimediali
- Viene realizzato dall'ingegnere del software applicando un processo che conduca a risultati di qualità elevata
- Come per ogni altro prodotto di successo, si applica al software un approccio ingegneristico
- Caratteristiche del software:
  - il software va "ingegnerizzato"
  - il software non si consuma
  - il software è complesso, invisibile, si conforma, si cambia

Conclusioni (4)

- Come assicurare la qualità del software che si produce?
- Come bilanciare la "domanda" crescente pur mantenendo il controllo del budget a disposizione?
- Come aggiornare applicazioni vecchie (*legacy*) ma ancora necessarie?
- Come evitare tempi di consegna più lunghi di quelli pianificati?
- Come applicare con successo le nuove tecnologie software?



I metodi e le tecniche di **Ingegneria del Software** hanno lo scopo di fornire le risposte a tali problemi, al fine di realizzare software con le desiderate caratteristiche di qualità.

I miti (da sfatare) del software

- In caso di ritardo, basta aumentare il numero di programmatori
- Una descrizione generica è sufficiente a scrivere i programmi. Eventuali modifiche si possono facilmente effettuare in seguito
- Una volta messo in opera il programma, il lavoro è finito
- Non c'è modo di valutare la qualità fino a quando non si ha a disposizione il prodotto finale
- L'ingegneria del software è costosa e rallenta la produzione