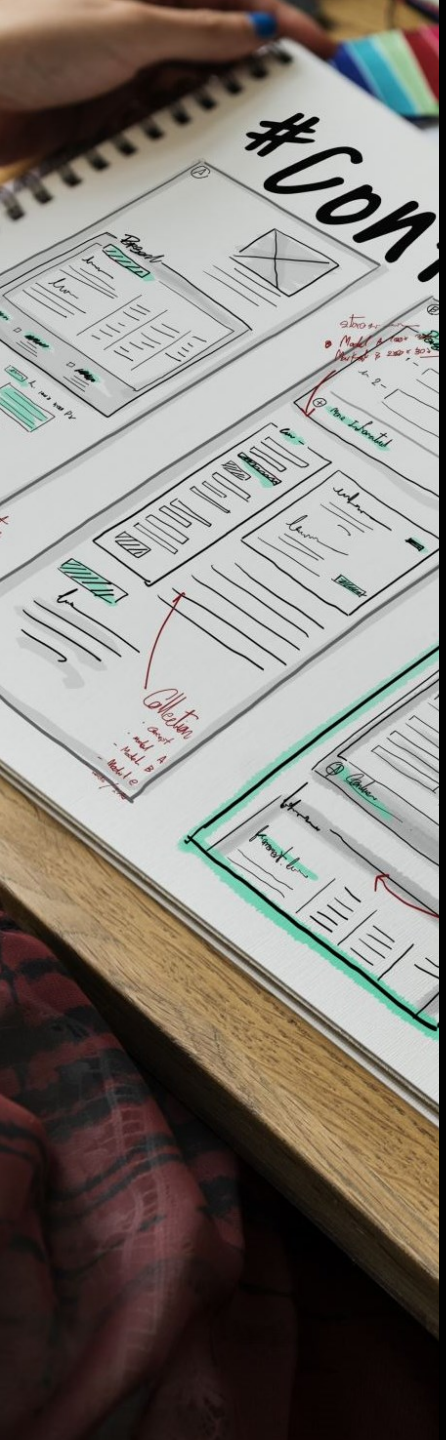


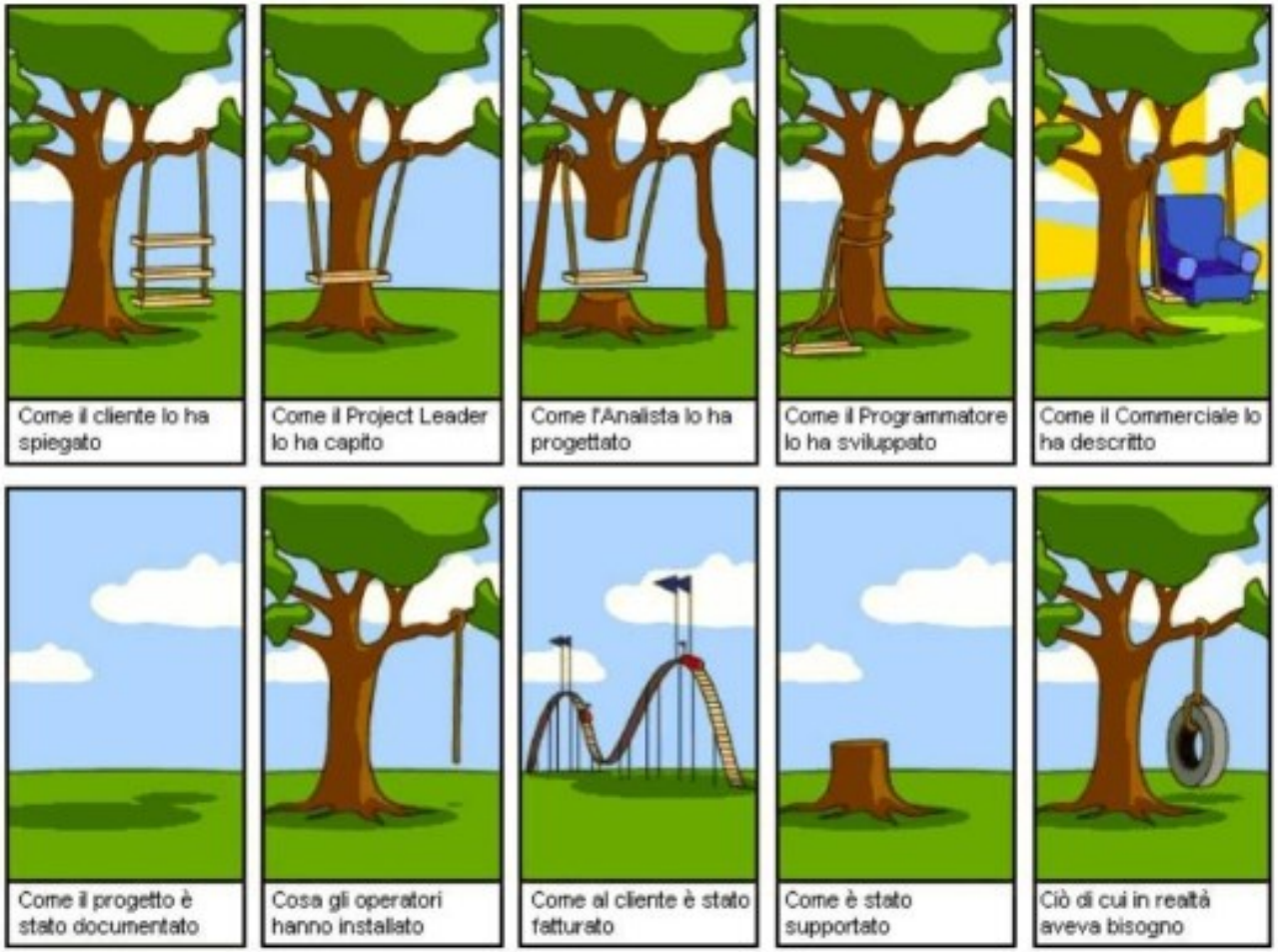
# Progettazione del Software Sicuro

Vincenzo Calabrò

# Ingegneria del Software

**Vincenzo Calabrò**

# Dai requisiti al prodotto



# Definizioni di Ing del SW

***Applicazione dell'ingegneria al software***

**Il campo della scienza informatica che si occupa di *progettare, realizzare* e *convalidare* sistemi software**

- di grande dimensione e complessità
- costruiti in team
- esistenti in molte versioni
- che durano per molti anni
- soggetti a modifiche

# Definizioni

**Applicazione di un approccio sistematico, disciplinato e quantificabile nello sviluppo, funzionamento e manutenzione del software (IEEE standard 610.12-1990)**

**Multi-person construction of multi-version software (Parnas 1978)**

# Diverse attività e strumenti


Capire chi fa cosa, perchè lo fa, quando lo fa, in che contesto lo fa, come lo fa, …

who, why, what, where, when, how

# Storia

**Il campo dell'ingegneria del software è nato nel 1968 in risposta ai fallimenti cronici di grandi progetti software che non sono riusciti a soddisfare i vincoli di pianificazione e budget**

- Il riconoscimento della "crisi del software"

**Termine è diventato popolare dopo la Conferenza NATO a Garmisch Partenkirchen (Germania) 1968**

# Ruolo dell'ingegnere del software

**La capacità di programmazione non è abbastanza**

**Fare l' ingegnere del software comporta "programmare-in-grande, ed occorre:**

- capire le esigenze (o *requisiti*) e scrivere le *specifiche*
  - ricavare *modelli* e *ragione* su di loro
- essere *master* del progetto software
- operare a vari livelli di *astrazione* (dal modello al codice)
- lavorare in *team*
- avere capacità di *comunicazione*
- avere abilità *manageriali*

# Ingegneria del SW *vs* programmazione

- Un programmatore scrive un programma completo
- Un ingegnere del sw progetta un componente sw che sarà poi combinato con altri per creare un sistema complesso
- Programmazione è un'attività individuale
- Ingegneria del sw è attività di gruppo
- Programmazione è implementazione
- Ingegneria del sw è progettazione, sviluppo, convalida, evoluzione

# Proprietà del software sicuro

## Prodotto e processo

**Prodotto**: ***cosa*** **realizziamo (software)**

**Processo**: ***come*** **realizziamo il prodotto**

- Esistono diversi modelli in cui organizzare le fasi di sviluppo

**Entrambi sono estremamente importanti**

**Entrambi hanno delle qualità**

# Le caratteristiche del prodotto SW

## Differente dai tipi tradizionali di prodotti

- Intangibile:
  - difficile da descrivere e valutare
- Malleabile:
  - può essere trasformato e dotato di nuove funzionalità
- *Human intensive*:
  - non comporta nessun processo manifatturiero tradizionale

# Qualità di prodotto e processo

- **Qualità del processo**:
  - riguardano i metodi utilizzati per lo sviluppo del SW
- **Qualità del prodotto**:
  - riguardano il SW stesso e sono sempre valutabili

**Le due categorie sono strettamente legate:**

- *le qualità del processo influisco su quelle del prodotto*

# Qualità del processo software

- Produttività:
  - misura l'efficienza del processo di produzione del SW in termini di *velocità di consegna* del SW
- Tempestività:
  - misura la capacità del processo di produzione del SW di *valutare e rispettare i tempi di consegna* del prodotto
- Trasparenza:
  - misura la capacità del processo di produzione del SW di *capire il suo stato attuale e tutti i suoi passi*

# Qualità del Software

**Le qualità su cui si basa la valutazione del SW sono classificate in:**

- **esterne** (black box view)
- **interne** (white box view)

**Le due categorie sono strettamente connesse:**

- *non è possibile ottenere le qualità esterne se il SW non gode delle qualità interne*

# Qualità esterne del SW

**Sono quelle percepite da un osservatore esterno che esamina il prodotto come se fosse una scatola nera visibile agli utenti (black box view)**

- Riguardano soprattutto le funzionalità del prodotto

**Alcune qualità esterne**

- Correttezza
- Affidabilità
- Efficienza
- Usabilità
- Portabilità
- Interoperabilità
- Robustezza

# Qualità interne del SW

**Sono quelle percepite da un osservatore che esamina il prodotto come se fosse una scatola trasparente (white box view)**

- Non sono visibili agli utenti
- Riguardano soprattutto le caratteristiche legate allo sviluppo del SW

**Alcune qualità interne**

- Riusabilità
- Verificabilità
- Facilità di manutenzione

# Qualità esterne del SW (1)

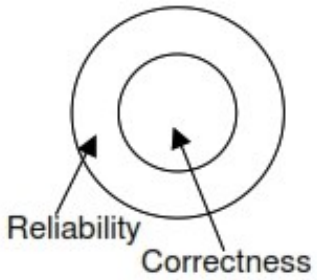
**Correttezza (o funzionalità):**

- un SW è corretto se rispetta le specifiche *funzionali* di progetto (assumendo che la specifica esista!)
  - E' una proprietà matematica che stabilisce l'equivalenza tra il SW e la sua specifica
    - Specifiche sbagliate possono dipendere da requisiti incorretti o da errori nella conoscenza del dominio di applicazione
  - Se le specifiche sono espresse formalmente, la correttezza può essere definita formalmente
    - Provata come un teorema (*verifica*), oppure valutata attraverso dei contro-esempi (*testing*)

# Qualità esterne del SW (2)

**Affidabilità (o reliability):**

- un SW è affidabile se si comporta *"come previsto"*
  - Può essere valutata matematicamente come la *"probabilità di assenza di fallimenti in un dato intervallo di tempo"*
  - Se le specifiche sono corrette, tutto il software corretto è affidabile, ma non vale il viceversa



# Qualità esterne del SW (3)

**Efficienza:**

- un SW è efficiente se usa intelligentemente le risorse di calcolo (ad es. memoria, tempo di processamento, metodi di comunicazione)
  - Può essere valutata attraverso analisi di complessità o la simulazione di scenari critici
  - Può influenzare la scalabilità:
    - una soluzione che funziona in piccolo può non funzionare in grande
  - Può cambiare con la tecnologia
  - Può influenzare la facilità d'uso

# Qualità esterne del SW (4)

**Usabilità:**

- facilità d'uso da parte dell'utente
  - Qualità difficile da valutare; molto soggettiva
  - Comporta la definizione di *expected user*
  - Influisce molto sulle interfacce utente:
    - visuale vs testuale

# Qualità esterne del SW (5)

**Portabilità:**

- un SW è portabile se può funzionare su più piattaforme
  - Esempio: programmi in Java

**Interoperabilità:**

- fa riferimento all'abilità di un sistema di coesistere e cooperare con altri sistemi
  - Esempio: un word processor in cui possono essere creati grafici

# Qualità esterne del SW (6)

**Robustezza:**

- un SW è robusto se si comporta in modo ragionevole anche in circostanze non previste dalle specifiche di progetto (es. input incorretti, rotture di dischi, ecc.)
  - N.B. La valutazione della correttezza e dell'affidabilità è basata sulle specifiche di progetto, mentre la robustezza riguarda tutti i casi non trattati
  - L'analisi delle conseguenze delle circostanze non previste può portare ad un ampliamento dei requisiti

## Qualità interne del SW (1)

**Riusabilità:**

- un SW è riusabile se può essere usato, in tutto o in parte, per costruire nuovi sistemi

**Verificabilità:**

- possibilità di *dimostrare* a posteriori la correttezza o altre caratteristiche del software
  - Verifica formale
  - Testing
  - Verifica a runtime

# Qualità interne del SW (2)

**Facilità di manutenzione:**

- facilità nel realizzare adattamenti o evoluzioni; agio nel correggere gli errori

**Un SW è facile da manutenere se:**

- è strutturato in modo tale da facilitare la ricerca degli errori (*modifiche correttive*)
- la sua struttura permette di aggiungere nuove funzionalità al sistema (*modifiche perfettive*)
- la sua struttura permette di adattarlo ai cambiamenti del dominio applicativo (*modifiche adattative*)

# Diversi tipi di sistemi software

- **… quindi ulteriori proprietà specifiche dei sistemi**
- **Sistemi informativi**
  - Per gestire informazioni
    - Sistemi bancari, sistemi bibliotecari, sistemi amministrativi
- **Sistemi in tempo reale**
  - Il sw risponde ad eventi esterni entro un periodo di tempo predefinito (e limitato)
    - Monitoraggio di impianti, sistemi di sorveglianza, di automazione
- **Sistemi distribuiti**
  - Macchine indipendenti collegate in rete (IoT)
- **Sistemi embedded**
  - Componenti sw che controllano componenti hardware
    - Usato in aerei, robot, elettrodomestici, automobili, cellulari, dispositivi medici, ecc

# Sicurezza

**Un nuovo aspetto da gestire: la sicurezza**

**Un nome per due termini:**

**safety**

- il sistema non causa danno all'esterno
- cose cattive non accadono *dal* sistema

**security**

- il sistema non subisce danni dall'esterno
- cose cattive non accadono *sul* sistema

## Sicurezza: una proprietà relativa

**Sicurezza non è una *proprietà assoluta* del sistema**

- Il suo significato è *relativo* al contesto di applicazione
- Può essere definita rispondendo alla domanda: sicurezza contro *cosa* e da *chi*?
- Implica stabilire una *policy* di sicurezza

## Qualità del SW sicuro

**Oltre le qualità comuni a tutto il SW, un SW sicuro deve godere delle seguenti qualità (*security goals*) :**

- prevenzione
- tracciabilità e controllo (o *auditing*)
- monitoraggio
- privatezza e confidenzialità
- sicurezza a diversi livelli
- anonimato
- autenticazione
- integrità

## Prevenzione

- Anticipare possibili attacchi
  - A livello di progettazione
  - A livello di implementazione
  - A livello d'uso

## Tracciabilità

- E' il meccanismo che consente di stabilire in modo inequivocabile la relazione di causa/effetto tra elementi, eventi, o processi

- Utile per riparare ad un attacco

## Controllo (o *auditing*)

- Per auditing si intende il processo di controllo di un sistema, effettuato sulla base del confronto tra le attività svolte sul sistema con le politiche e le procedure stabilite al fine di determinare la loro conformità, suggerendo eventualmente l'opportunità di introdurre delle migliorie

- Non è una tecnica di prevenzione, ma è utile per dissuadere da potenziali attacchi

# Monitoraggio

- Monitoraggio è auditing in tempo reale
- Sistemi per il monitoraggio possono causare un elevato numero di falsi allarmi
- È possibile monitorare un programma a diversi livelli
  - Approcci semplici:
    - controllo di segnature note che identificano un attacco in corso
  - Approcci complessi:
    - monitoraggio del codice mediante *asserzioni*

# Privatezza e confidenzialità

## Privatezza

- E' il diritto di un individuo di stabilire se, come, quando e a chi l'informazione che lo riguarda può essere rilasciata

## Confidenzialità

- Assicura che certi servizi e informazioni siano accessibili solo ad utenti autorizzati

## Sicurezza a più livelli

- Alcuni tipi di dati e informazioni sono più segrete di altre
  - Informazioni classificate in: pubbliche, confidenziali, top-secret, ecc.
- È necessario disporre di sistemi in grado di gestire la segretezza di dati e informazioni a più livelli

## Anonimato

- Va intesa come la proprietà di mantenere segreta (o non accessibile pubblicamente) l'origine di certi dati
- E' una proprietà con doppio risvolto
- Il SW prende spesso decisioni non anticipate e programmate sull'anonimato
  - Il Global Identifier di Microsoft
  - Il sistema Carnivore dell' FBI
- Architetti e sviluppatori debbono pensare attentamente a cosa avviene dei dati collezionati dai propri programmi
  - Dati potrebbero essere usati in cattivo modo? E come?

# Autenticazione

- Va intesa come la proprietà di conoscere l'identità di *chi* accede ad un servizio
- E' una proprietà critica per la sicurezza
  - Un SW sicuro quasi sempre include elementi di autenticazione
- Scarsa è l'autenticazione su WEB
  - La tecnologia SSL (Secure Socket Layer) assicura la protezione dei dati ma non garantisce l'identità del server a cui si è connessi
- Esistono parecchi modi per garantirla in relazione al SW di applicazione

## Integrità

- Nel contesto della sicurezza, integrità va intesa come "*rimanere lo stesso*", ossia non modificato da quando creato

- In sistemi sicuri, le informazioni e le risorse non devono essere modificate, cancellate o distrutte in modo non autorizzato o *improprio*