Ingegneria del Software (e Prova Finale)

Luca Mottola
luca.mottola@polimi.it

(credits: Luciano Baresi)

Organizzazione dei corsi

- Ingegneria del software (7 crediti)
  - Primo semestre
  - Lezioni: 42 ore
  - Esercitazioni: 28 ore
- Prova finale (3 crediti)
  - Secondo semestre
  - Esercitazioni: 12 ore
  - Laboratorio: 32 ore

La squadra

- Docente:
  - Luca Mottola
- Esercitazioni
  - Rolando Brondolin

Programma

- Programmazione nei linguaggi orientati agli oggetti
  - Linguaggio Java
- Progettazione orientata agli oggetti
  - Unified Modeling Language
  - Design pattern
- Specifica di metodi e classi
- Principi di programmazione di rete e distribuita
- Principi di programmazione delle interfacce utente
- Principi del test funzionale e strutturale

Materiale didattico e organizzazione

- Non esiste un libro di testo unico
- Unico sito di riferimento: Corso Beep
  - Materiale lezione, esercitazione, lab
  - Puntatori ad ulteriore materiale e letteratura
  - Software e tools
  - Forum e avvisi… attivate le notifiche!
- Inizio “alla mezz’ora”
- Non seguiremo la suddivisione ufficiale lezioni/esercitazioni/lab
- Calendario ufficiale su goo.gl/LLZNTS

Prova finale

- Obiettivo: applicare concretamente i concetti appresi nel corso di Ingegneria del Software
- Progetto
  - Lo stesso per tutti
  - Gruppi da 2 o 3 persone
- Inizio nel secondo semestre

Iniziamo?

Ingegneria del Software

- Settore dell'informatica che studia sistemi
  - **Complessi** e di grandi dimensioni
  - Nati dal lavoro di gruppo
- Questi sistemi
  - Esistono in diverse **versioni**
  - Hanno una **durata** di anni
  - Sono soggetti a frequenti **modifiche**

Possibili definizioni

- Approccio sistematico allo sviluppo, alla messa in opera e alla manutenzione del software
- Metodi tecnici e manageriali per prevedere e tenere sotto controllo i costi per tutta la vita ("lifecycle") dei prodotti software
- Come tutte le ingegnerie:
  - Fornisce una guida per applicare la conoscenza scientifica allo sviluppo di soluzioni (software) "cost-effective" per risolvere problemi pratici

Processo e prodotto

- Processo
  - Come avviene lo sviluppo industriale del software
- Prodotto
  - Che cosa viene prodotto?

Studiare i metodi da usare perché il processo porti allo sviluppo di prodotti di qualità

Ingegneria

- Progetto normale/standard
  - Soluzione a un problema noto e ricorrente
  - Riuso di soluzioni note
  - Innovazione limitata
    - …tipico di discipline mature
- Progetto **innovativo**
  - Soluzioni radicalmente nuove a problemi non noti
- Occorre saper distinguere tra i due

Differenze

- Ingegnerie tradizionali
  - Prevale il progetto di routine
  - Progetto di estremo dettaglio che produce le specifiche per la realizzazione
  - Processo di produzione separato
  - Progetti alternativi convalidati attraverso modelli
  - Dopo il progetto, pochi margini di cambiamento
  - Processi standard per progetto e produzione

Confronto con ingegneria tradizionale

- (Troppo) spesso il software viene trattato come “progetto innovativo”
- (Troppo) spesso l’ingegneria del software viene praticata in modo poco sistematico (ingegneristico/industriale)

Ingegneria del software (1)

- L’ingegneria civile ha alle spalle 3000 anni
  - Un patrimonio di conoscenze
- Ciò è vero per quasi tutte le ingegnerie
- L’ingegneria del software ha solo 50 anni

Ingegneria del software (2)

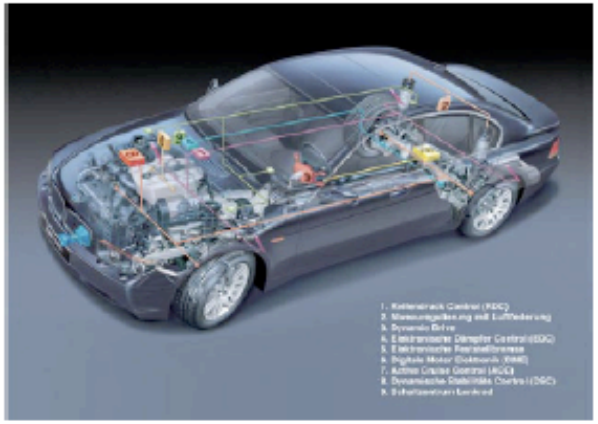
- Congelare le caratteristiche e **specifiche** di prodotto e di progetto è spesso non realistico
- Cambiamenti ed evoluzione spesso **inevitabili**
  - Il software è il cuore di processi sociali e di business
  - Questi continuano ad evolvere

Il software oggi

- Il software è parte essenziale di molti prodotti di largo consumo
  - Dal telefonino alla lavatrice, dall'automobile al forno
- Spesso il software non è il prodotto, ma è una parte del prodotto
  - Deve essere ingegnerizzato con il resto del sistema
- Il software diventa commodity, le revenues arrivano dall'hardware

Automotive Software: An Emerging Domain
A Software Perspective

- Up to 40% of the vehicles' costs are determined by electronics and software
- 90% of all innovations are driven by electronics and software
- 50 – 70% of the development costs for an ECU are related to software
- Premium cars have up to 70 ECUs, connected by 5 system busses



- **Growing system complexity**
- **More dependencies**
- **Costs play significant role**


SEARCH
12:10
Events
28
DEC 2012
27
DEC 2012
Select
Import
Camera
12:10
Wednesday, 2 January
14 tweets received
SEARCH
Music
Recently played
Lana Del Rey
Born To Die - The Paradise Edition
Kylie Minogue
Impossible Princess
Ciara
Gimmie Dat
Marina and the Diamonds
Obsessions
+ View all (135)


Knowledge
Information

Complessità, criticità, e dimensione

- Fanno la vera differenza
  - Richiedono un approccio sistematico (ingegneristico) per poter ottenere la necessaria qualità controllando costi e tempi
- Secondo F. Brooks (The Mythical Man Month)
  - "programmare per se stesso" rispetto a "programmare per altri" → costo al quadrato


-00:00:17

```
float a;
int b;
// …
b = (int) a;
```

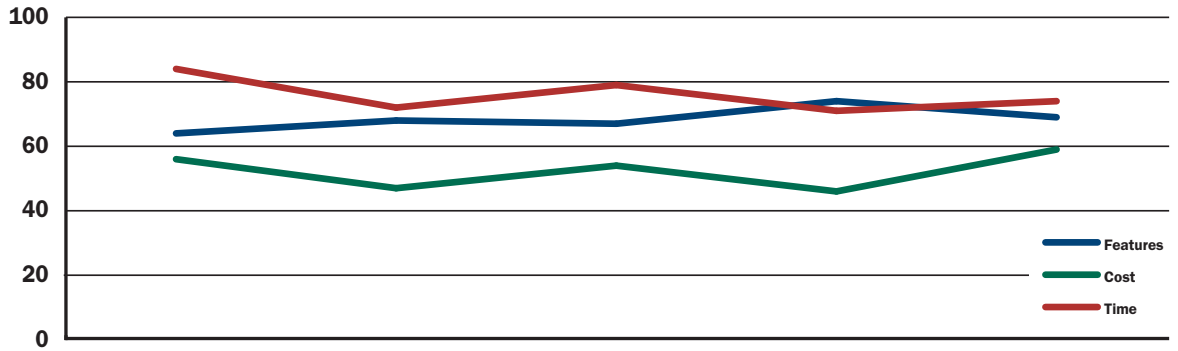
RESOLUTION

	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

Project resolution results from CHAOS research for years 2004 to 2012.

OVERRUNS AND FEATURES

Time and cost overruns, plus percentage of features delivered from CHAOS research for the years 2004 to 2012.



	2004	2006	2008	2010	2012
TIME	84%	72%	79%	71%	74%
COST	56%	47%	54%	46%	59%
FEATURES	64%	68%	67%	74%	69%

Progettazione vs. Programmazione

- Saper programmare **non basta** per essere un ingegnere del software
  - Programmatore
    - Sviluppa un programma completo
    - Partendo da specifiche fornite da altri
    - Lavora individualmente
  - Ingegnere del software
    - Analizza problemi e domini applicativi
    - Coglie i requisiti e sviluppa specifiche
    - Progetta componenti, potenzialmente riusabili
    - Lavora in (ed alle volte coordina) un gruppo

Progettazione

- Scomposizione di un sistema in moduli
  - **Scomporre** un problema in sotto-problemi che possano essere risolti indipendentemente
- Quali obiettivi della scomposizione?
  - Governare la complessità
  - Rendere efficiente il processo
    - Sviluppo indipendente delle parti
    - Riduzione di conflitti/incomprensioni fra gli sviluppatori