F1I018: INGEGNERIA DEL SOFTWARE

Introduction to Software Engineering

Lecturer: Prof. Henry Muccini

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Università degli Studi dell'Aquila

Copyright Notice

3

Agenda

Welcome to the Software Engineering Course

Why Software Engineering?

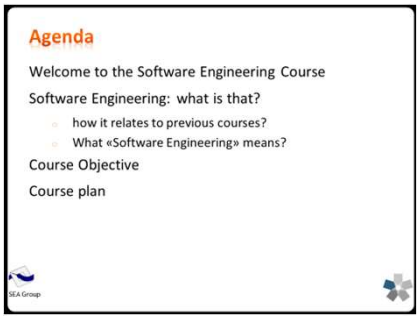
Software Engineering: what is that?

- how it relates to previous courses?
- What «Software Engineering» means?

Course Objective

Course plan

WELCOME

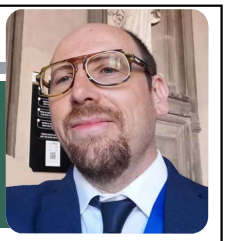

Agenda
Welcome to the Software Engineering Course
Software Engineering: what is that?
how it relates to previous courses?
What «Software Engineering» means?
Course Objective
Course plan
SEA Group

MY PROFILE

- Professor in Computer Science
  - Software Engineering area
- Topics of interest:
  - Software Engineering
  - Software Architetture
  - Model Driven Engineering
  - Software Engineering ML systems
- Teaching:
  - Software Engineering (2006-today)
  - Software Architetture (2006-today)
  - Design of web/mobile Applications @Master in Web Technology (2004-today)
  - Advanced Software Architecture (2010-today)

5

On developing methods and tools for the analysis and design of software architectures



- Machine Learning a supporto delle Architetture Adattive
- Microservizi, IoT, e Modernization
- Green AI
- Architetture dei dati

Course Administration

Codice del Team: 8n5mbah

Course Contract

- Lectures start at 2:45 on Monday and 11:45 on Wednesday
- Take notes!
  - Slides are NOT enough.
- Registration for the exam is "mandatory"
  - No registration, no exam
- Ask questions!

SOFTWARE ENGINEERING: WHY?

Fault management

Complexity management

Composizionalità/riuso

Why Software Engineering?

DISASTRO CROWDSTRIKE

Blocco globale dei PC Windows, cause e conseguenze: attenzione alle truffe

Home > Attacchi Hacker E Malware: Le Ultime News In Tempo Reale E Gli Approfondimenti

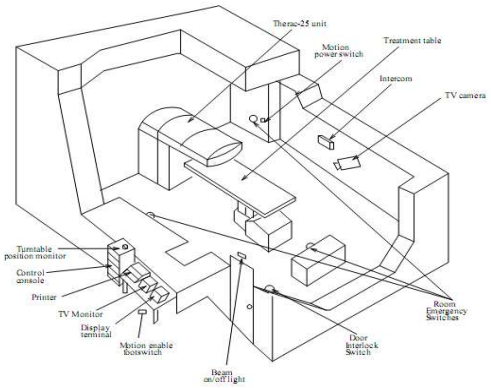
Il grande cyber caos che ha impattato su diversi milioni di computer Windows, ha rimarcato l'importanza di prestare attenzione a eventi di questa portata, proprio per le attività malevole che immediatamente si innescano

Pubblicato il 22 lug 2024

Examples of «Badly» Engineered Software

Therac-25 **safety** failure:

- approximately 100 times the intended dose of radiation
- 3 people died, and 6 got injured

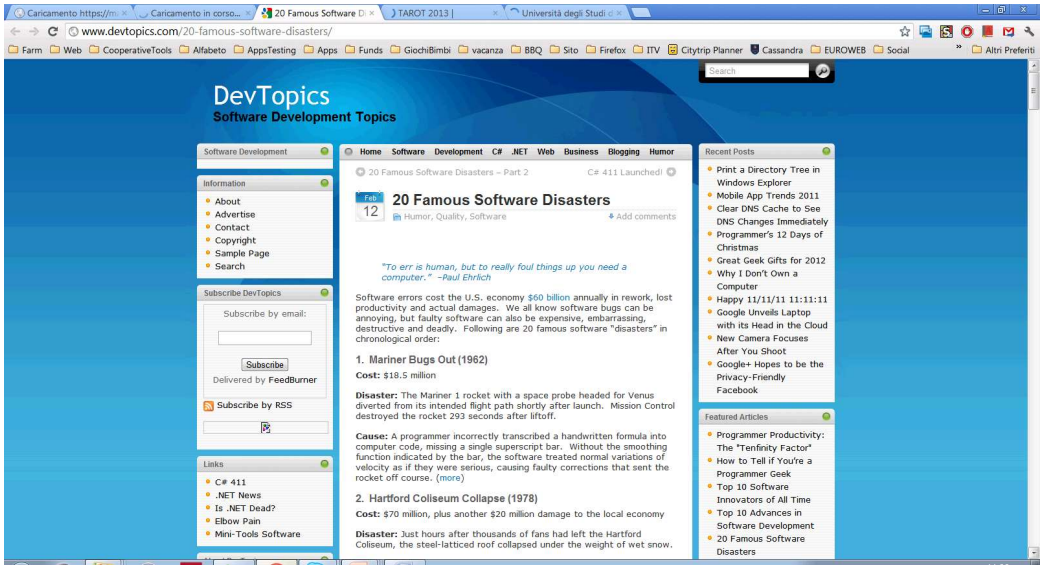


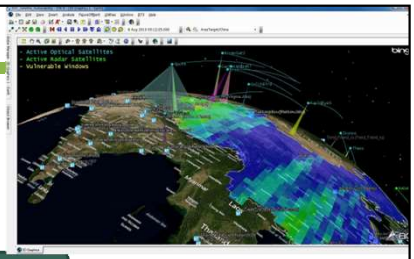
see article at: http://sunnyday.mit.edu/papers/therac.pdf

Factors:

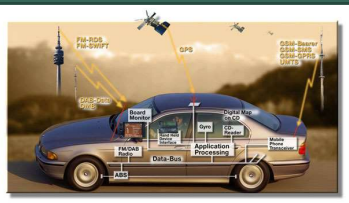
- Overconfidence in Software
- Confusing reliability with safety
- Lack of defensive Design
- Failure to eliminate fault causes
- Inadequate software engineering practices
- …

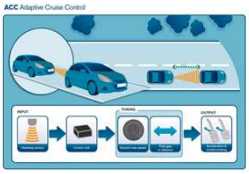
http://www.devtopics.com/20-famous-software-disasters/



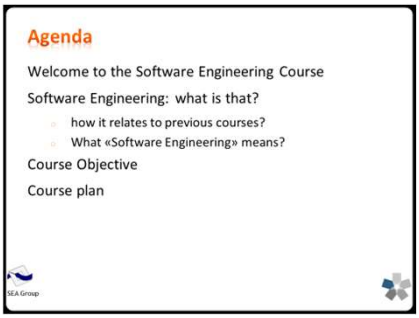


Gestire sistemi «complessi»





Discussione sui tipi di complessità



SOFTWARE ENGINEERING: WHAT IS THAT?

Engineering

Engineering is the use of scientific principles to design and build machines, structures, and other items, including bridges, tunnels, roads, vehicles, and buildings.[1] The discipline of engineering encompasses a broad range of more specialized fields of engineering, each with a more specific emphasis on particular areas of applied mathematics, applied science, and types of application. See glossary of engineering.

Civil Engineering

Civil engineering is a professional engineering discipline that deals with the design, construction, and maintenance of the physical and naturally built environment, including public works such as roads, bridges, canals, dams, airports, sewage systems, pipelines, structural components of buildings, and railways.[1][2]

[https://en.wikipedia.org/wiki/Civil_engineering]

My Software Engineering working Def.

Set of automated methods

to systematically develop quality software

that fulfils customer needs

while satisfying existing constraints

Software Engineering @ NATO conference – 1968 -

https://en.wikipedia.org/wiki/Software_engineering

https://en.wikipedia.org/wiki/Software_engineering

AppsTesting | Apps | Funds | GiochiBimbi | vacanza | Firefox | EUROWEB | Social | Azienda | Easyline | BBQ | Blog dei Genito... | Video2MP3_Yo... | KeepVid: Downl... | Altri Pr

Main article: History of software engineering

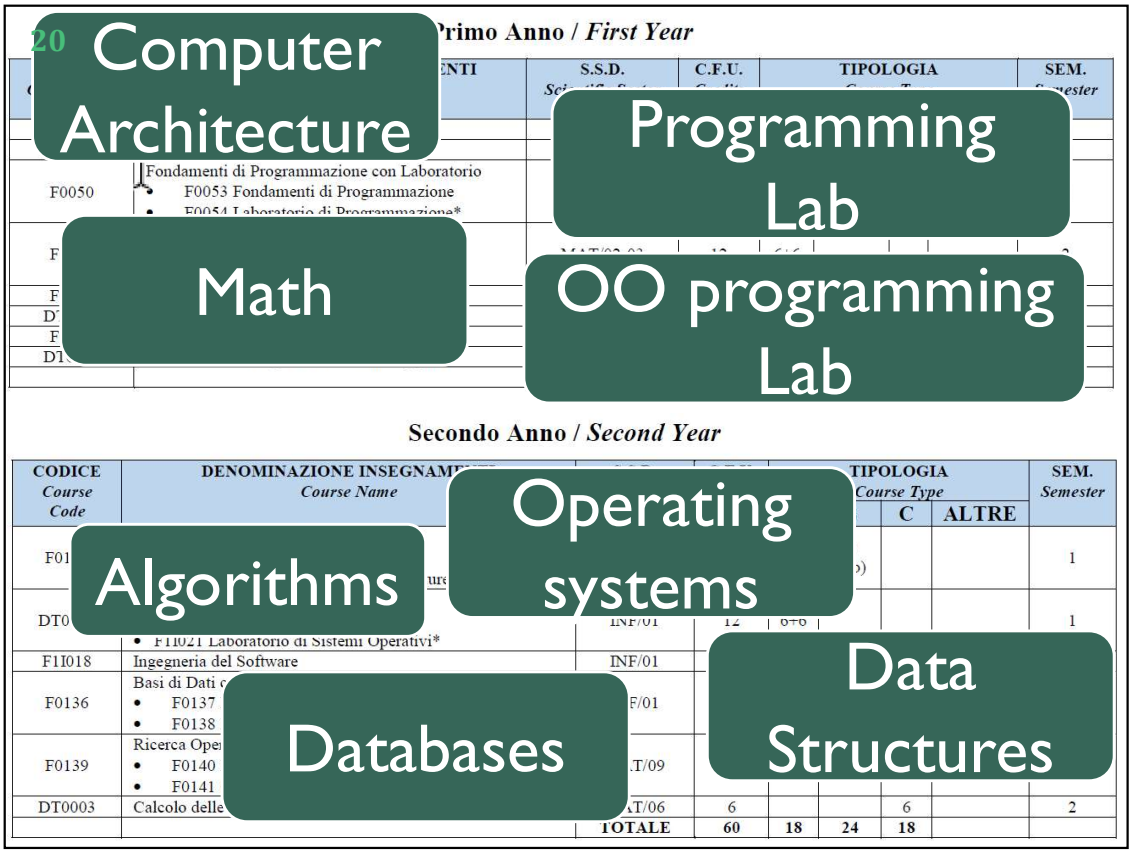
When the first digital computers appeared in the early 1940s,[9] the instructions to make them operate were wired into the machine. Practitioners quickly realized that this design was not flexible and came up with the "stored program architecture" or von Neumann architecture. Thus the division between "hardware" and "software" began with abstraction being used to deal with the complexity of computing.

Programming languages started to appear in the 1950s and this was also another major step in abstraction. Major languages such as Fortran, ALGOL, and COBOL were released in the late 1950s to deal with scientific, algorithmic, and business problems respectively. E.W. Dijkstra wrote his seminal paper, "Go To Statement Considered Harmful",[10] in 1968 and David Parnas introduced the key concept of modularity and information hiding in 1972[11] to help programmers deal with the ever increasing complexity of software systems.

The term "software engineering", coined by Margaret Hamilton,[12][13] was first used in 1968 as a title for the world's first conference on software engineering, sponsored and facilitated by NATO. The conference was attended by international experts on software who agreed on defining best practices for software grounded in the application of engineering. The result of the conference is a report that defines how software should be developed [i.e., software engineering foundations]. The original report is publicly available.[14]

The discipline of software engineering was created to address poor quality of software, get projects exceeding time and budget under control, and ensure that software is built systematically, rigorously, measurably, on time, on budget, and within specification. Engineering already addresses all these issues, hence the same principles used in engineering can be applied to software. The widespread lack of best practices for software at the time was perceived as a "software crisis".[15][16][17]

Margaret Hamilton, credited with coining the term software engineering, standing with the data results from the simulation of the code[*clarification needed*] her team designed for Apollo 11.

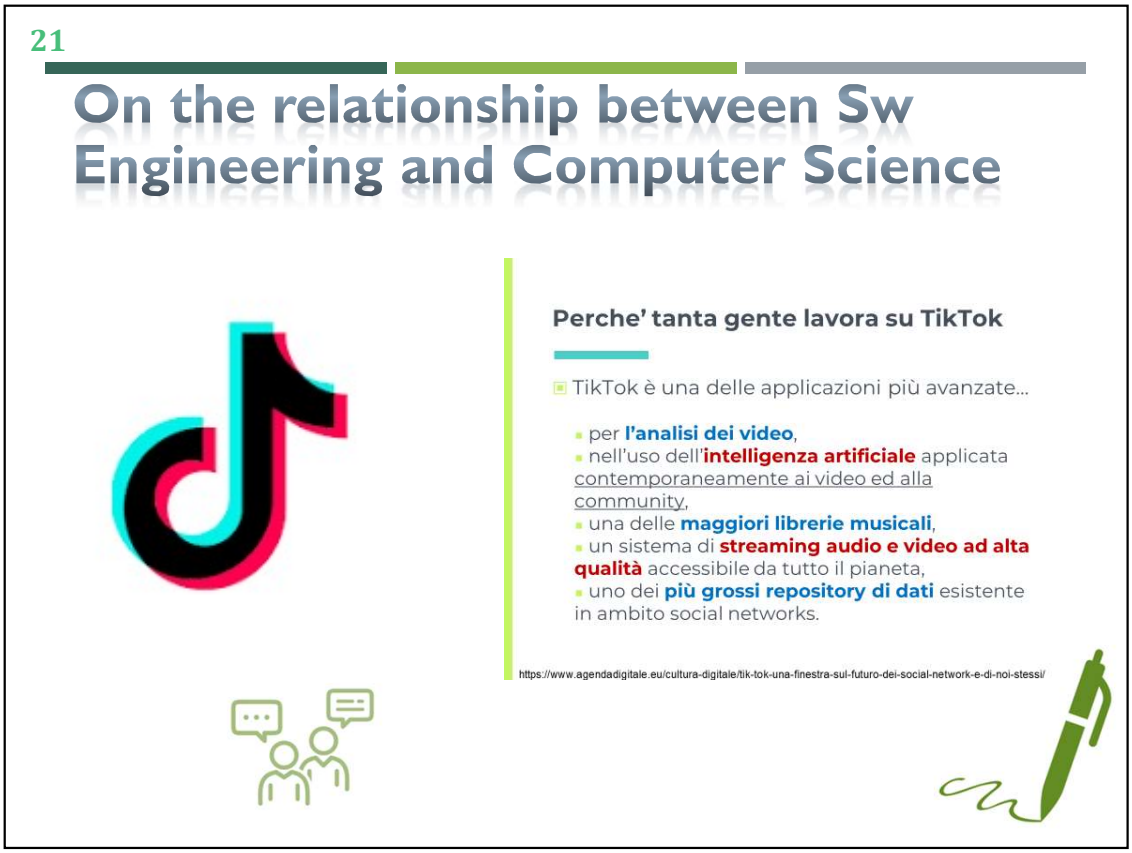


Primo Anno / *First Year*

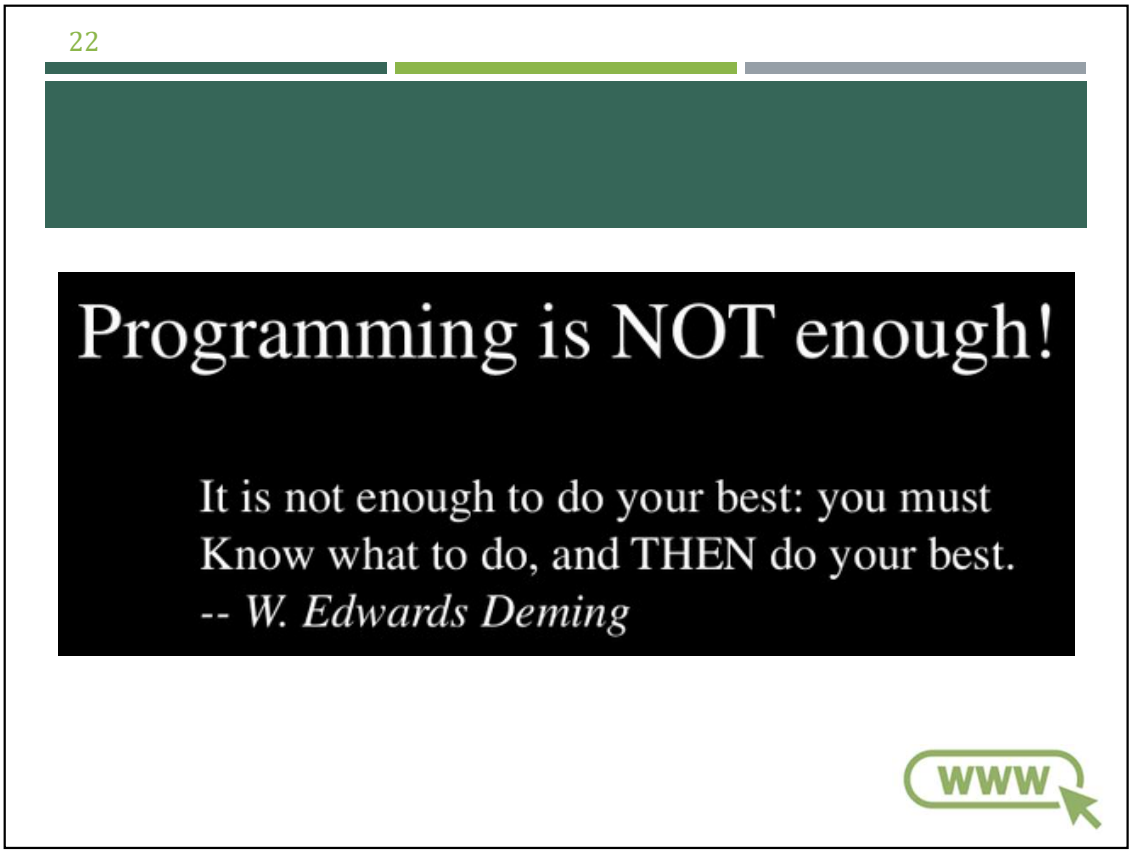
		S.S.D.	C.F.U.	TIPOLOGIA	SEM.
F0050	Fondamenti di Programmazione con Laboratorio • F0053 Fondamenti di Programmazione • F0054 Laboratorio di Programmazione*				

Secondo Anno / *Second Year*

CODICE *Course Code*	DENOMINAZIONE INSEGNAM... *Course Name*					TIPOLOGIA *Course Type* C	TIPOLOGIA ALTRE	SEM. *Semester*
F01								1
DT0	• F11021 Laboratorio di Sistemi Operativi*	INF/01	12	6+6				1
F1I018	Ingegneria del Software	INF/01						
F0136	Basi di Dati c • F0137 • F0138	F/01						
F0139	Ricerca Ope • F0140 • F0141	.T/09						
DT0003	Calcolo delle	AT/06	6			6		2
		TOTALE	60	18	24	18		



Algoritmi per la scelta di video da mostrare, data analytics, mobile/web, data storage, …



https://it.wikipedia.org/wiki/William_Edwards_Deming

23

A Software System is… not only software… not only programming

≠

Programming is NOT enough!

It is not enough to do your best: you must
Know what to do, and THEN do your best.
-- *W. Edwards Deming*

Sw Process: come metto insieme i mattoncini?

Vincoli

Terreno

etc

Quand'e' che un sistema software è ben ingegnerizzato?

Quali sono i tipici problemi da evitare?

Quand'e' fondamentale ingegnerizzare un sistema software

COURSE ORGANIZATION

Course Organization

- Lecture
- Lab
- Project
- Quizz
- Oral Exam

Il Programma del Corso

Requirements Engineering

Software Design

Software Development Process

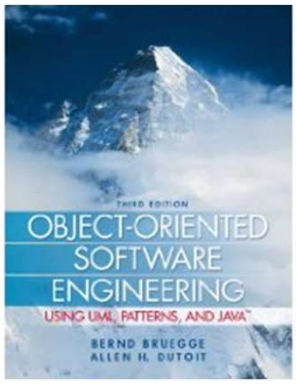
Project Management

Sw Testing

Course Administration

Codice del Team: 8n5mbah

TextBook



Bernd Bruegge, Allen H. Dutoit

Object-Oriented Software Engineering: Using UML, Patterns and Java, 3rd Edition

Publisher: Prentice Hall, Upper Saddle River, NJ, 2009;

ISBN-10: 0136061257

ISBN-13: 978-0136061250

Additional readings will be added during the lectures.

Verifica dell'apprendimento

- L'esame prevede un progetto, una prova orale, e dei quiz.

  - progetto: volto all'applicazione pratica della teoria presentata durante il corso. Il progetto viene sviluppato in maniera incrementale, seguendo un processo di sviluppo RUP. Il progetto viene discusso dal team che lo ha realizzato, e conta per il 60-80% del voto finale (a seconda della complessità dello stesso);

  - orale: volto a comprendere la preparazione teorica dello studente, la sua capacità di applicare la teoria in contesti più ampi del progetto, e la sua capacità espositiva verbale. L'esame orale conta per il 20/40% del voto finale;

  - quiz tematici: volti ad attivare subito l'interesse dello studente alla materia, coprono un tema specifico trattato a lezione. Tipicamente riconoscono 0,5 punti extra. Tali quiz non vengono assegnati tutti gli anni.

Course cataloque (link)

Verifica dell'apprendimento

- L'attribuzione del voto rifletterà i seguenti livelli di preparazione:

  - sufficiente: progetto minimale e conoscenza base degli argomenti trattati durante il corso;
  - adeguata: progetto adeguato ed acquisizione di una basilare abilità di progettazione dei sistemi software;
  - buona: buon progetto e capacità di ragionamento rigoroso;
  - approfondita: ottimo progetto e capacità di applicare la teoria su progetti complessi di diversa natura.

TODO IN CLASS (by you)

Max 10 PUNTI IMPORTANTI PRESENTATI IN CLASSE

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.