UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

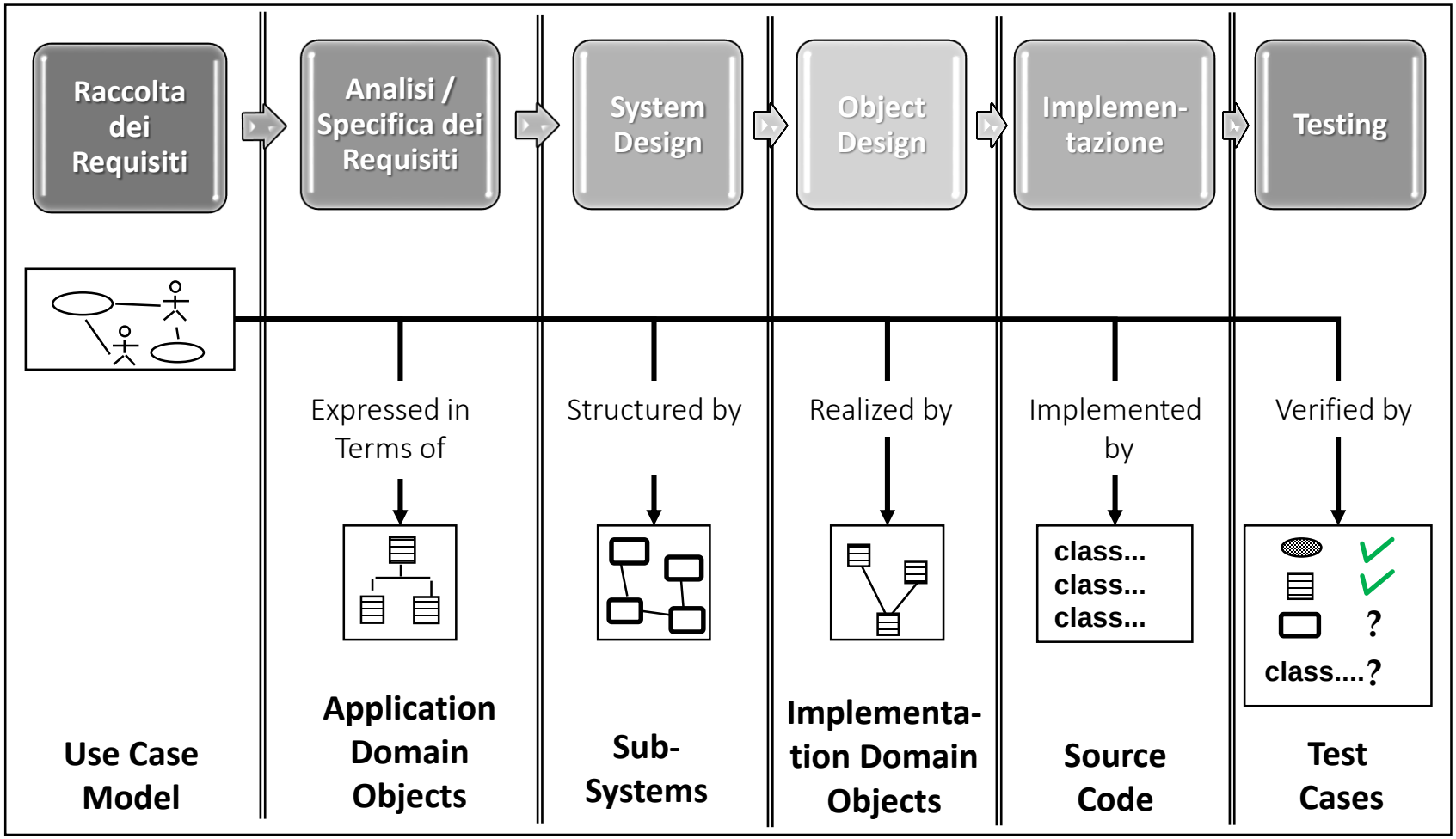
Ingegneria del Software – Introduzione al Corso

Prof. Sergio Di Martino

Contenuti della lezione

- Cos'è l'Ingegneria del Software
- Informazioni pratiche sul corso
- Informazioni sul docente
- Introduzione all'Ingegneria del Software
  - Discutere i concetti di Prodotto Software
  - Introdurre il concetto di Qualità del Software

Ciclo di Vita del Software



L’Ingegneria

- L’ingegneria propone ***metodologie di sviluppo***, che riassumono e formalizzano esperienze e conoscenze pregresse
  - Edilizia
    - Marco Vitruvio Pollione, *De Architectura* (1° Secolo A.C.)
  - Aereonautica, Automobili, etc…
  - Elettronica
  - Nucleare

Lo sviluppo software

- Molti fattori hanno storicamente limitato l'utilizzo di approcci ingegneristici nella produzione di software
  - L'intangibilità del software
  - Disciplina (relativamente) nuova
  - Approccio "artistico", non strutturato allo sviluppo
  - Mancanza di formalismi e strumenti di supporto

L’Ingegneria del Software

- Definizione IEEE (2007)
  - *“L’ingegneria del software è la disciplina tecnologica e manageriale che riguarda la produzione sistematica e la manutenzione dei prodotti software entro tempi e costi preventivati”*

- Informalmente: sviluppo cost-effective di software di **qualità**

Informazioni sul corso

Perché seguire questo corso

- Produrre software non è (solo) un'arte e neppure (solo) una scienza: è un'*industria*
  - Si lavora sempre in un contesto di gruppo e di azienda
  - vincoli economici e requisiti di qualitá
- Come in ogni industria, per produrre software sono state sviluppate *metodologie* di progetto, di sviluppo e di verifica
- Il laureato in Informatica *deve* conoscerle!
  - Non basta che siate i migliori programmatori possibili
  - Dovete sapere come ANALIZZARE, PROGETTARE e VALIDARE un software nella sua interezza

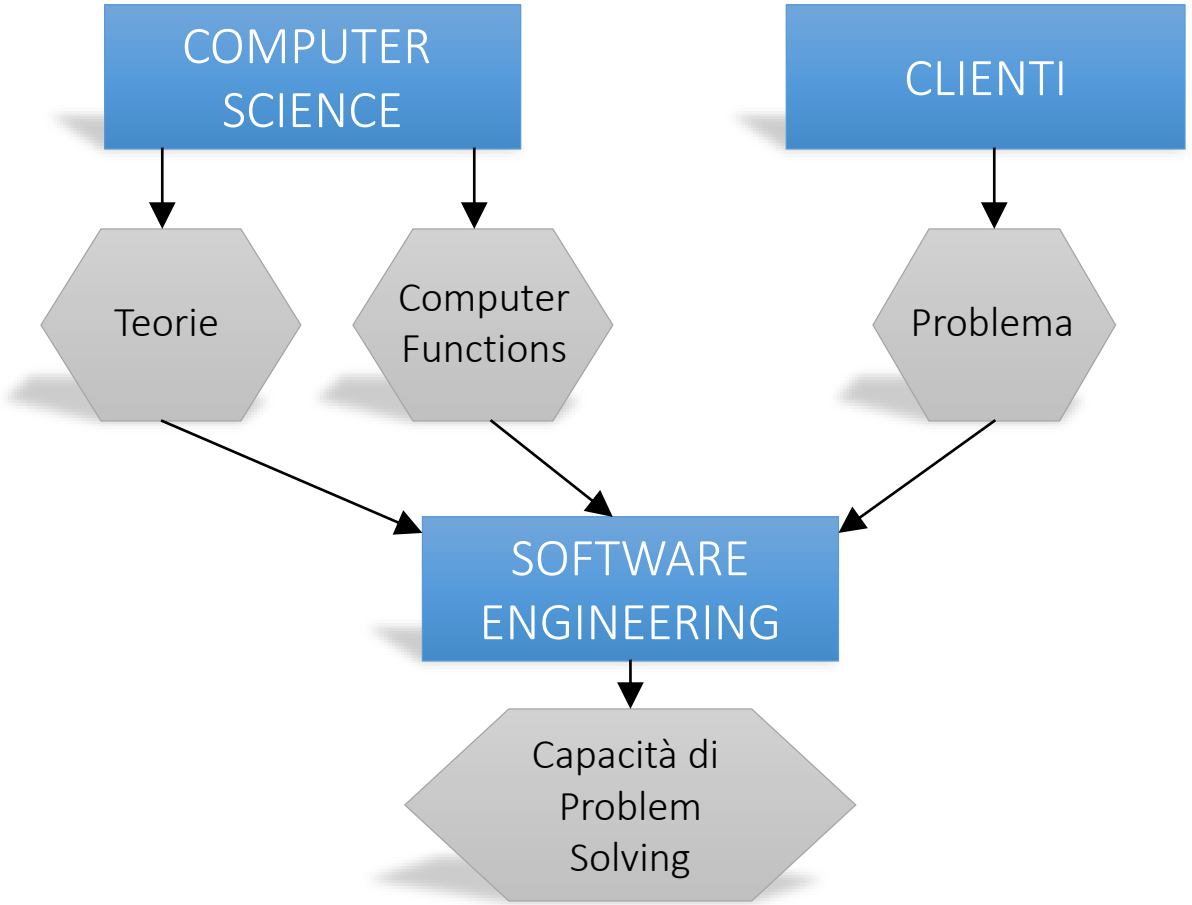
Ingegneria del Software

- Conoscenze che si intendono trasmettere (sapere):
  - Concetti di base dell'ingegneria del software, dei processi di ingegneria del software e delle relative fasi, attività e deliverable;
  - Metodi di analisi e progettazione, e importanza dei linguaggi di modellazione del software per la comunicazione tra diversi attori coinvolti in un processo di ingegneria del software;
  - Concetti e tecniche di testing e validazione del software;
  - Concetti di manutenzione del software, e principali problematiche della gestione dei progetti software

Ingegneria del Software

- Capacità che si intendono sviluppare (saper fare):
  - Saper usare un approccio ingegneristico/strutturato all'analisi, design e testing del software.
  - Saper comprendere e produrre gli appropriati documenti durante tutte le varie fasi del processo di sviluppo software
  - Sapere usare la notazione UML per modellare il software
  - Saper sviluppare software di qualità

Ruolo dell'Ingegneria del Software



Informazioni pratiche

- Corso obbligatorio per la laurea triennale
  - Nuovo ordinamento (Mat. N86): 9 CFU
- Svolgimento
  - I semestre: 6 ore settimanali
  - Presenza non obbligatoria ma fortemente consigliata

Comunicazioni

- Gruppo Google per la comunicazione-discussione:
  UNINA ING SW 2015-16
- Indirizzo per chiedere adesione

  https://groups.google.com/d/forum/unina-ing-sw-2015-16

- Il nick del richiedente DEVE essere nel formato

  Cognome Nome Matricola

- Materiali del corso sul Sito del docente:
  www.docenti.unina.it/sergio.dimartino

Modalità di esame

- Progetto obbligatorio di gruppo
  - Tre documenti da produrre:
    1. Analisi,
    2. Progettazione
    3. Testing
- Scritto: esercizi e domande sull'intero programma.
  - Obbligatorio.
- Orale: intero programma.
  - Facoltativo per chi abbia avuto allo scritto un voto >= 21
  - Obbligatorio per chi abbia avuto allo scritto un voto < 21

Progetto come gioco di ruolo

- Un’unica traccia, declinata differentemente
- Committente: Docente
  - Saranno fornite specifiche incomplete e potenzialmente inconsistenti.
  - Le specifiche vanno raffinare in incontri programmati e contingentati (per numero e durata).
- Azienda produttrice: Gruppo di studenti.
- Si dovranno produrre tre documenti di progettazione
  - Analisi dei requisiti
  - Progettazione di sistema
  - Progettazione dei casi di test

Valutazione del Progetto

- Deadline ben precise e assolutamente non derogabili
  - Normalmente ogni 30, 15 e 7 giorni prima degli appelli
- Valutazione dell'intera interazione committente-contraente.
  - Interazione e uso degli strumenti di comunicazione.
  - Qualità grafica dei documenti prodotti.
  - Qualità della progettazione.
- Una valutazione almeno sufficiente su tutti e tre i documenti da produrre è prerequisito per accedere alle prova scritta.
- I tre documenti contribuiscono in maniera paritetica alla valutazione complessiva del progetto

Formazione dei gruppi

- Consistenza numerica: Min 2 persone Max 4 persone
  - Esperienza di lavoro in team.
  - Saranno ammessi gruppi singoli solo per motivati e documentati impedimenti (ad esempio lavorativi)
- Formazioni dei gruppi
  - Autonome comunicate nel gruppo del corso
  - Operate dal docente in base alle disponibilità per studenti che non riescano a stabilire formazioni autonome.
- Variazione dei gruppi
  - Ogni variazione di un gruppo ufficializzato deve essere concordata con il docente

Cheating Policy

- In caso di due o più progetti ritenuti troppo simili, ad entrambi i gruppi il progetto sarà annullato e sarà data una nuova traccia, più estesa e complessa di quella originaria
- Detection rate: almeno una coppia di progetti l'anno.

Materiali di studio (1)

- Libri di testo generali consigliati
- Parti generali
  - I. Sommerville. Software Engineering, VIII ed., Addison Wesley, 2007.
- Progettazione ad oggetti
  - C. Larman, Applicare UML e i Pattern - Analisi e Progettazione orientata agli Oggetti, III ed. Prentice-Hall, 2005.
  - B. Bruegge, A. Dutoit. Object-Oriented Software Engineering, Pearson, 2008. (Alternativo a C. Larman).

Materiali di studio (2)

- Libri di testo generali consigliati
- UML
  - Stevens Rod Pooley, Usare UML, Addison Wesley, 2008.
  - J. Arlow, Ila Neustadt, UML2 e Unified Process, McGraw-Hill, 2006.
- Altri testi su aspetti specifici
  - P. Amman, J. Offutt. Introduction to software testing, Cambrigde Univerisity Press, 2008.
  - E. Gamma, R. Helm, R.Johnson, J.Vissides. Design patterns, Addison Wesley,

Materiali di studio (3)

- Lucidi delle lezioni
  - I lucidi saranno disponibili on-line.
  - I lucidi delle lezioni **non sono sostitutivi** dei libri di testo.
- Esempio di un progetto svolto.
- Materiali vari indicati di volta in volta al corso.

Informazioni sul docente

Prof. Sergio Di Martino

Contatti

- Ufficio: Stanza T.03, via Claudio 21
- Ricevimento: Venerdì 11:00 – 13:00

- Sito del docente:
  www.docenti.unina.it/sergio.dimartino

Comunicazione via mail

- Mail:
  - sergio.dimartino@unina.it
  - Subject: [INGSW1] *e poi l'oggetto*
  - Firmare SEMPRE le mail
  - Non mandare mail per quesiti su aspetti già descritti nel sito web e/o sul gruppo Google.
  - Le mail che non rispettano queste regole non solo non riceveranno risposta, ma saranno fonte di valutazione negativa!

- La comunicazione cliente/committente è uno degli aspetti chiave nell'Ingegneria del Software!

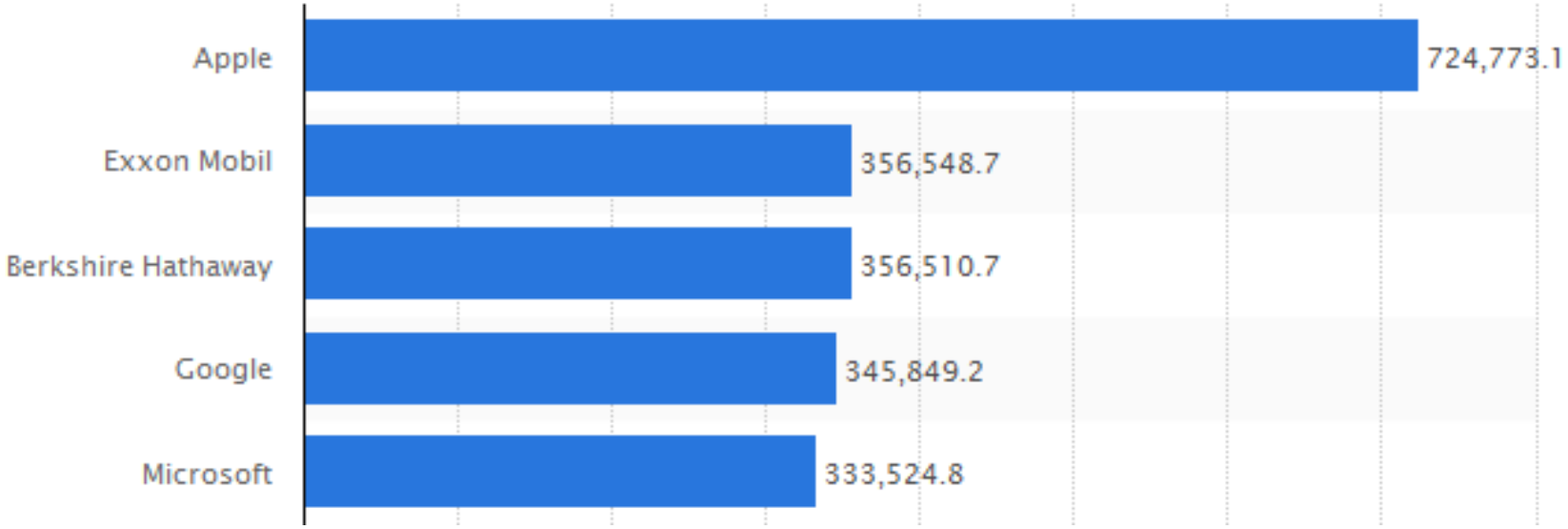
L’Ingegneria del Software: Introduzione

Evoluzione della Produzione di Software

- **Arte:** applicazioni sviluppate da singole persone e utilizzate dagli stessi sviluppatori
- **Artigianato:** applicazioni sviluppate da piccoli gruppi specializzati per un cliente
- **Industria:** diffusione del software in molteplici settori (tutti?); crescita di dimensioni, complessità e criticità delle applicazioni; mercato e concorrenza; necessità di migliorare la produttività e la qualità; gestione dei progetti; evoluzione del software

Rivelevanza della produzione dei Sistemi Software nell'economia moderna

- L'ICT rappresenta uno dei principali settori dell'economia mondiale.
- Le cinque più grandi aziende del mondo (e loro valore in $):



Dimensioni di un tipico software

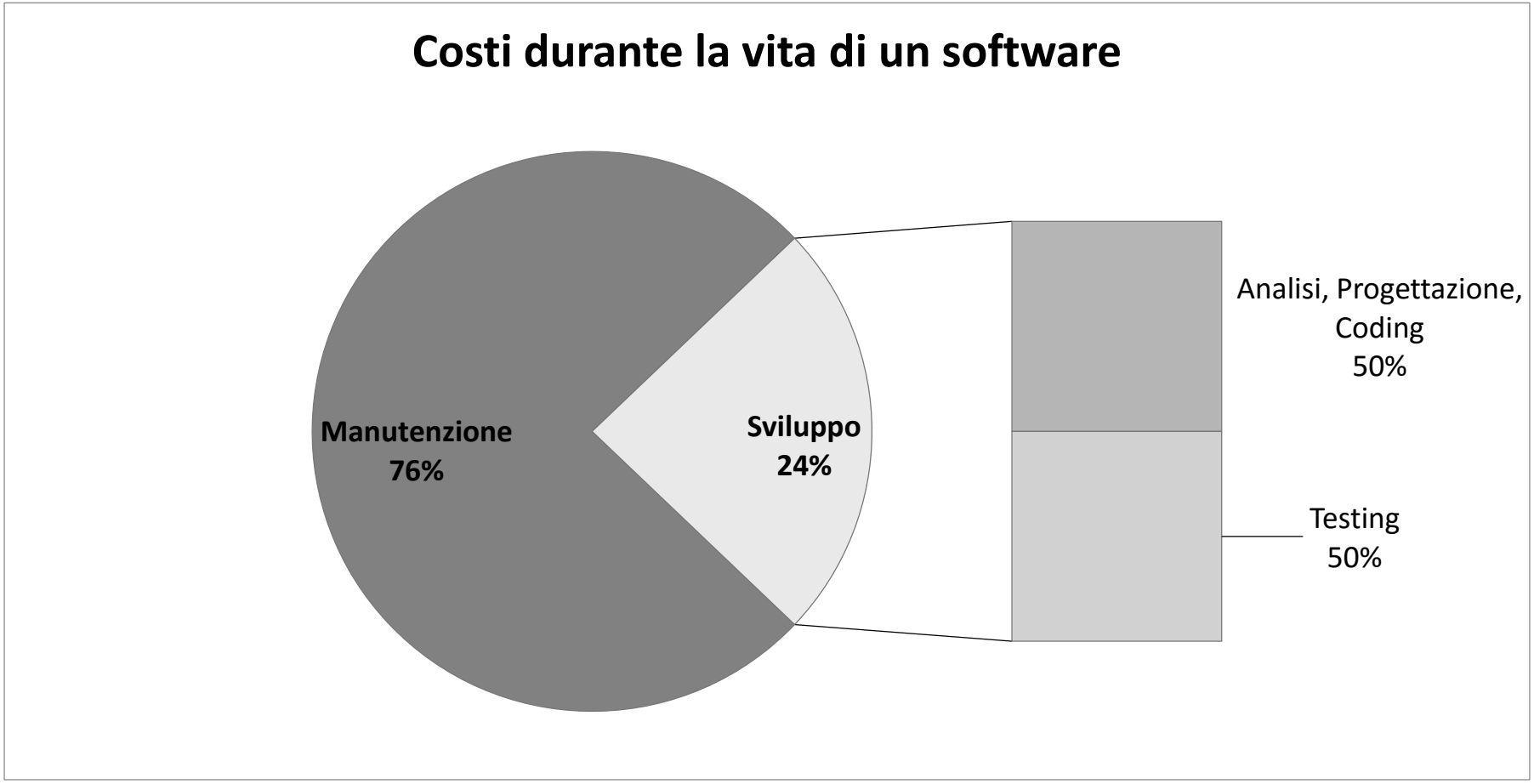
- Dimensioni in termini di Linee di Codice per alcuni sw di uso comune:

Sistema	Linee di codice
App Media per iOs	400.000
Firefox	18.000.000
Windows 7	40.000.000
Vettura alto di gamma	100.000.000

Problemi della produzione software: Costi

- Il software ha costi elevati
  - Sono i costi delle risorse usate: ore lavoro (manpower), hardware, software e risorse di supporto. Il manpower è dominante !
    - Il costo è espresso in mesi/uomo
    - Tipico costo di un programmatore per un'azienda: 10.000€/mese
  - La manutenzione costa più dello sviluppo
    - Per sistemi che rimangono a lungo in esercizio i costi di manutenzione possono essere svariate volte il costo di produzione

Software e costi



Ingegneria del Software

- Software engineering. (IEEE SE Glossary 612.20-1990)
  1. *The application of a systematic, disciplined, quantifiable ap- proach to the development, operation, and maintenance of software; that is, the application of engineering to software.*
  2. *The study of approaches as in (1).*
- Produzione di prodotti di alta qualità
- Possibile uso ed integrazione di prodotti/soluzioni preesistenti
- Rispetto di vincoli di budget e tempo di produzione.
- Aspetti metodologici, tecnologici, qualitativi, manageriali.

Tendenza sulle professioni più retribuite (2011)

- Registered Nurses $64.690
- Accountants and Auditors $61.690
- Management Analysts $78.160
- **Computer Application Software Engineers $91.180**
- Physicians and Surgeons $94.000-153.000
- **Computer Systems Analysts $77,740**
- Market Research Analysts $60.570
- Civil Engineers $77,650
- Dental Hygienist $68.250
- Personal Financial Advisor $64.750

Computer Application Software Engineers

- Total new jobs (2008-2018): 175,100
- Pct. increase: 34%
- Median income: 94,180
- States with most jobs per capita: Washington, Colorado, Virginia
- *In the age of the mobile app and cloud computing, few positions are becoming as essential as software application developers. In 2008, 514,000 people worked in this position. By 2018, the BLS National Employment Matrix projects there will be 175,000 additional positions. According to the BLS's description of the job prospects for software developers, "As a result of rapid employment growth over the 2008 to 2018 decade, job prospects for computer software engineers should be excellent. Those with practical experience and at least a bachelor's degree in a computer-related field should have the best opportunities." Those willing to put in the time to get certified will be well compensated. The median annual income is nearly $100,000.*

Computer System Analists

- Total new jobs (2008-2018): 108.010
- Pct. increase: 20%
- Median income: 77.740
- States with most jobs per capita: Virginia, Delaware, new Jersey
- *Computer systems analysts build and manage computer networks for companies for use in file sharing and inter-office communication. They also maintain web security within the network. This position, only 30-years-old, is expected to grow by 20%, adding nearly 110,000 jobs before the end of this decade. While an increasing number of businesses are requiring bachelor's degrees, it is still very possible to get a job as a systems analyst with just a high school diploma and some certification. The median annual income, $77,740, is also one of the best salaries one can get without a college degree.*

Il concetto di Prodotto Software

Programmi vs Prodotti (Sommerville)

- **Programma**: l'autore è anche l'utente (e.g., non è documentato, quasi mai è testato, non c'è progetto)
  - non serve approccio formale, molto difficilmente ha un mercato
- **Prodotto software**: usato da persone diverse da chi lo ha sviluppato
  - è software industriale il cui costo è circa 10 volte il costo del corrispondente programma
  - Necessità di un approccio formale allo sviluppo

Prodotto software

- Più che un programma eseguibile:
  - Programmi eseguibili
  - I dati di configurazione, che permettono di installarlo
  - Il manuale utente
- E' un prodotto industriale
  - Sviluppato con standard produttivi industriali
  - Prevede tutta la documentazione che descrive la progettazione e realizzazione del sistema.

Tipologie di Prodotti Software

- **Prodotti generici** (general purpose) OTS: off the shelf
  - sistemi stand-alone prodotti da una organizzazione e venduti a un mercato di massa
- **Prodotti specifici** (specific purpose)
  - sistemi commissionati da uno specifico utente e sviluppati specificatamente per questo da un qualche contraente
- La fetta maggiore della spesa è nei prodotti generici ma il maggior sforzo di sviluppo è nei prodotti specifici
- La differenza principale
  - Chi dà le specifiche del prodotto (il produttore o il consumatore).

Il Concetto di Qualità del Software

Qualità del Software

- Quality (IEEE SE Glossary).
  1. *The degree to which a system, component, or process meets specified requirements.*
  2. *The degree to which a system, component, or process meets customer or user needs or expectations.*
- Gran parte della ricerca nel campo dell'ingegneria del software è dedicata, direttamente o indirettamente, al tema della **qualità**.

Classificazione di qualità

- Esistono varie classificazioni e metriche per la Qualità del Software:
  - Standard ISO/ IEC 9126:2001
  - Qualità Esterna ed Interna
  - Modello di Qualità di McCall
  - Modello di Qualità di Bohem
  - …

Qualità Esterna e Qualità Interna

- I fattori rispetto a cui si può misurare la qualità del software vengono classificati in:
  - **Fattori Esterni (**o di **Prodotto)** - la qualità del software percepita dagli utenti
  - **Fattori Interni (**o di **Processo)** - la qualità del software percepita dagli sviluppatori

Qualità esterne del SW (1)

- **Correttezza**
  - Un software si dice corretto se si comporta in accordo a quanto previsto dalla sua specifica dei requisiti.
- **Affidabilità**
  - Un sistema è tanto più affidabile quanto più raramente, durante l'uso del sistema, si manifestano malfunzionamenti.
- **Usabilità**
  - Un sistema è facile da usare se un essere umano lo reputa tale.

Qualità esterne del SW (2)

- **Scalabilità**
  - Un sistema è scalabile se può essere adattato a diversi contesti con forti differenze di complessità senza che questo richieda la riprogettazione dello stesso sistema.
- **Efficienza**
  - Un sistema è efficiente se usa le risorse HW/SW in modo proporzionato ai servizi che svolge.
- **Robustezza**
  - La robustezza di un sistema è la misura in cui il sistema si comporta in modo ragionevole in situazioni impreviste, non contemplate dalle specifiche.

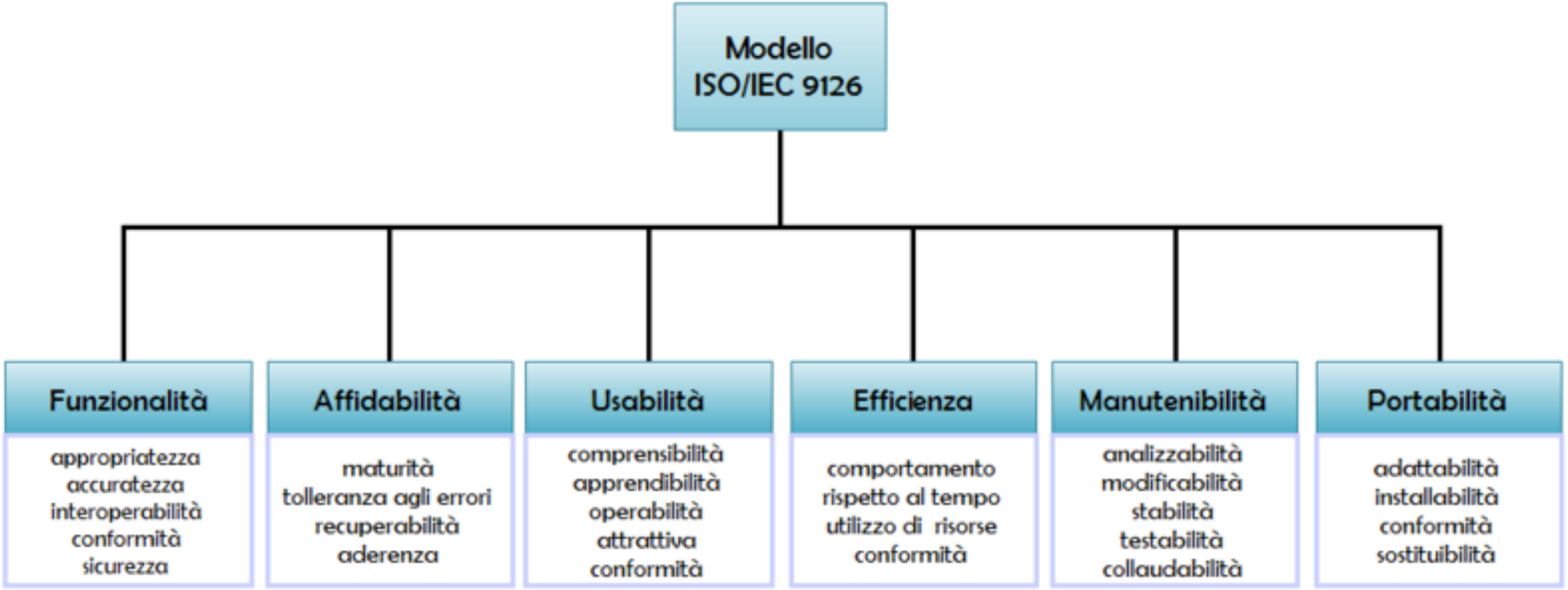
Qualità interne del SW (1)

- **Riparabilità**
  - Un sistema è riparabile se la correzione degli errori è agevole. La riparabilità si persegue attraverso la modularizzazione e opportune metodologie di progettazione.
- **Manutenibilità**
  - Facilità di apportare modifiche a sistema realizzato.
- **Riusabilità**
  - Facilità con cui è possibile riusare parti di sistema per realizzare un prodotto diverso.

Qualità interne del SW (2)

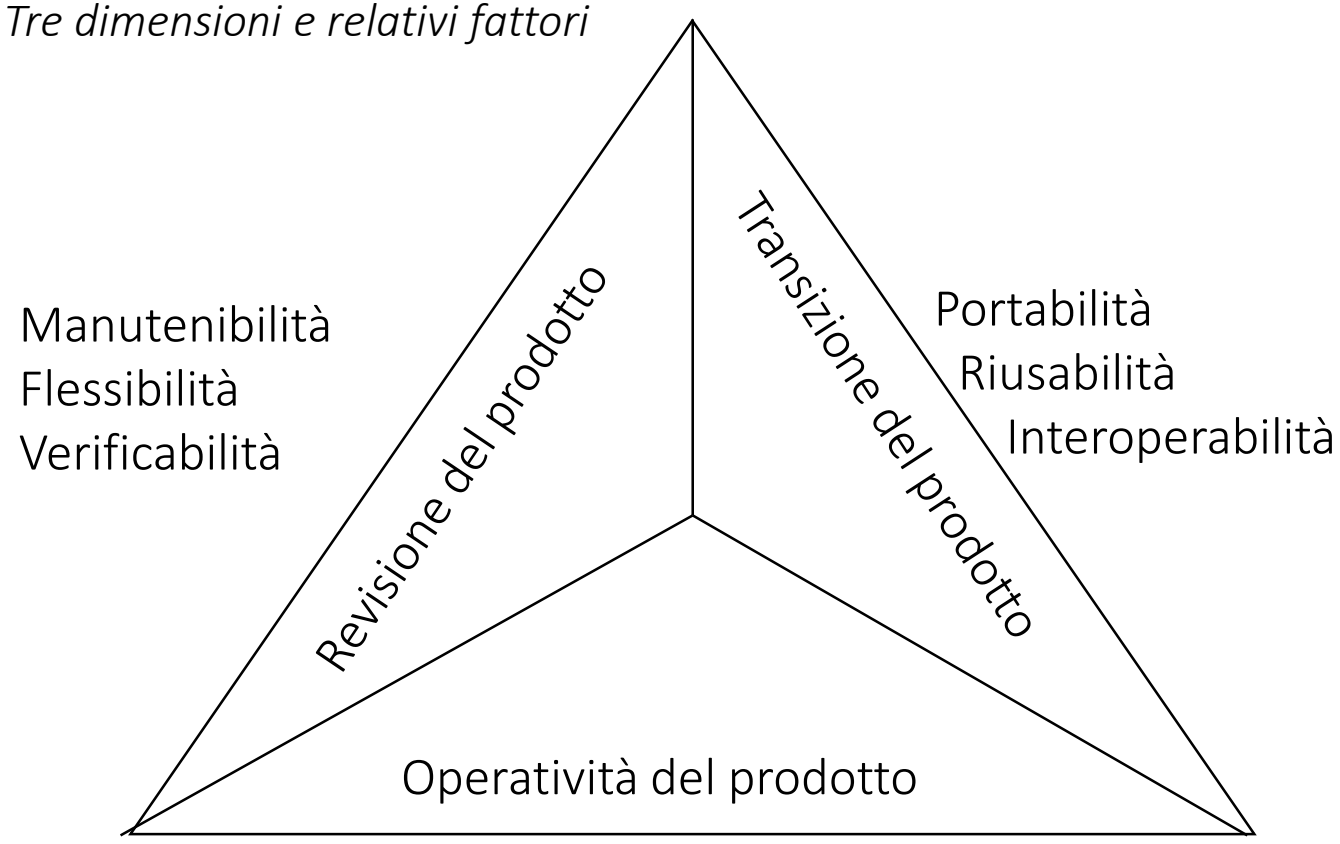
- **Portabilità**
  - Un sistema è portabile se è in grado di funzionare in ambienti diversi.
- **Verificabilità**
  - Un sistema è verificabile se le sue proprietà di correttezza e di affidabilità sono facilmente validabili.

Il modello di qualità ISO/IEC 9126



Il modello di qualità di Mc Call

Tre dimensioni e relativi fattori



Correttezza; Affidabilità; Efficienza; Usabilità; Integrità