Introduction to Software Engineering

Fall 2022

Nguyen Van Vu

Outline

- Teaching Staff
- Course Description
- Learning Objectives
- Course Requirements
- Grading
- Academic Integrity
- Class Schedule
- Introduction to SE

Teaching Staff

- Instructor
  - Nguyen V. Vu
    - FIT, HCMUS
    - Email: nvu@fit.hcmus.edu.vn
    - Phone: 090-817-5957
  - Tran Duy Thao
    - Email: tdthao@fit.hcmus.edu.vn

Course Description

- One of the first courses in Software Engineering
- Introduces basic concepts, principles, practices, methods, techniques, and tools in software development and maintenance
- Applies software engineering principles and practices to developing software in multi-person teams
- Prerequisites
  - Programming skills
  - Data structure

Topics

- Topics covered
  - Software management
  - Software processes
  - Software requirements engineering
  - Software analysis and design
  - Software testing
  - User interface design
  - Software reuse
  - Software configuration management
  - Software maintenance and evolution
  - Component-based SE
  - Service-oriented SE
- Text books
  - Required*: Software Engineering*, 10th Ed, Ian Sommerville, Addison-Wesley
  - Optional
    - *The Mythical Man-Month,* Frederick Brooks, Jr., Addison-Wesley, 1995
    - *Concise Guide to Software Engineering, Gerard O'Regan, Springer, 2017*

Learning Objectives

- By the end of the class, students will
  - Understand basic concepts, principles, methods, and techniques in software engineering
  - Be able to apply requirements engineering concepts to define a system requirements
  - Be able to analyze and design a software system
  - Be able to design and write a test plan and test cases for a software system
  - Be able to apply software testing techniques to test a software system
  - Be able to determine a suitable process for a software project based on its characteristics
  - Apply the best practices in planning, monitoring, and controlling a software project
  - Be able to manage project risks
  - Be able to practice teamwork

Course Requirements

- Students must obtain a **non-zero** grade for each of the grading components, including
  - project assignments (weekly)
  - in-class quizzes
  - In-class activities, participation
  - final exam

Course Requirements (cont'd)

- Project assignments
  - Students will be assigned to 3-5 student project
  - Performs all activities of the software development lifecycle to deliver software
  - Deliver written and oral reports
  - Oral presentation given in class at the end
- In-class quiz
  - Short quizzes are given randomly in class (unannounced in advance)
  - Given before or after lecture
- In-class activities, participation, and discussion

Course Requirements (cont'd)

- Moodle used for material distribution and communication
- Questions beneficial to both the questioner and others should be posted on Moodle's forum
- Students encouraged to ask questions in class, via forum, email, or in-person
- Late submission policy
  - 15% grade reduction for each day late
  - Zero grade for 4 or more days late
  - Exceptions are given for certain cases, e.g., illness

Grading

- Grade Distribution
  - Project assignments 40%
  - In-class quiz 10%
  - In-class activities, participation, and discussion 10%
  - Final exam (cumulative) 40%
- Grade in the 100th scale will be scaled into the 10th scale

Academic Integrity

- Students are prohibited from copying
  - from classmates, friends even if allowed
  - from the Internet without proper citation (see next slide)
- Students are prohibited from allowing others to copy
- Other kinds of cheating and plagiarizing

- If the academic integrity violated, serious measures will be taken
  - 1st violation: zero grade for the assignment violating
  - 2nd violation and more: students will be failed the class and report to the Faculty

Academic Integrity (cont'd)

- How to cite sources properly?
  - If copying verbatim, put copied sentences/phrases in the double quotes
  - If rephrasing a source, put a reference to the source
- Copying whole phrase or sentence:

> *"It is a matter of some urgency that we as a research community define and agree reporting protocols and methods for comparison"* [1]

- Rephrasing:

> Shepperd believes that the research community needs to define a reporting protocols and methods for comparison [1]

- Reference:

> [1] Shepperd M, "Software project economics: a roadmap", Future of Software Engineering (FOSE'07), 2007

Class Schedule

- See the schedule in Syllabus for detail

Question about the class?

Software Engineering
Introduction

Adapted from the Slides of Software Engineering, 10th Ed. by Ian Sommerville

Topics covered

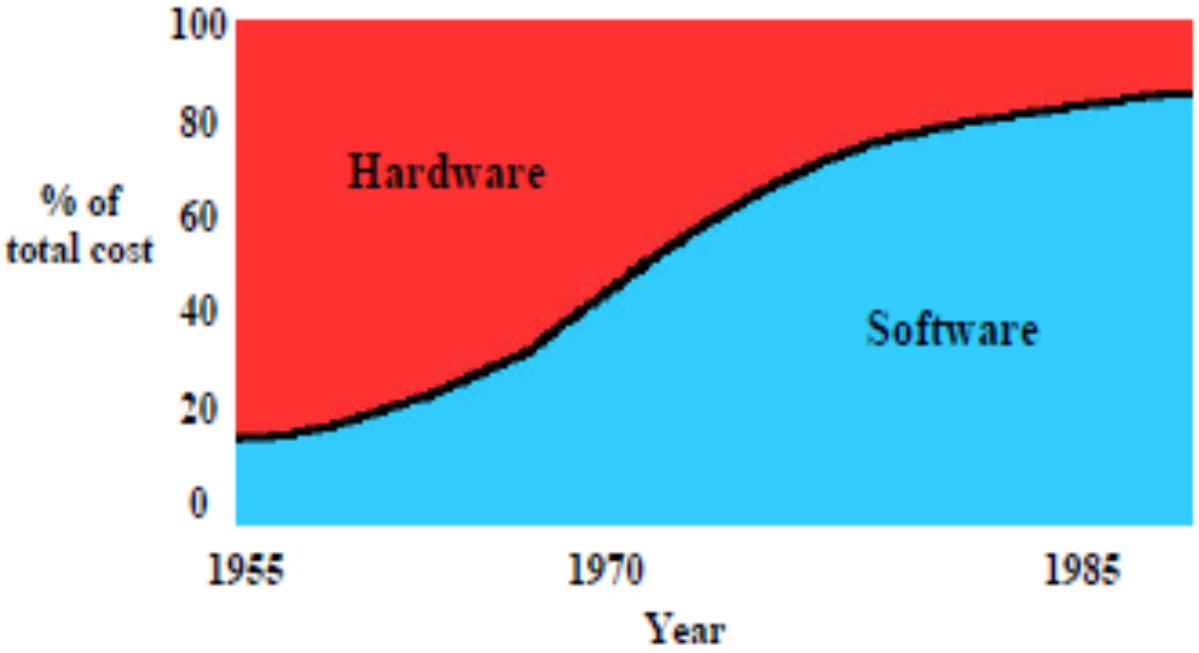
- FAQs about software engineering
- Professional and ethical responsibility

Software engineering

- Economies of ALL developed nations are dependent on software
- More and more systems are software controlled

- Is there anything that connects to the Internet without being software?

Software costs (Boehm, ‘81)



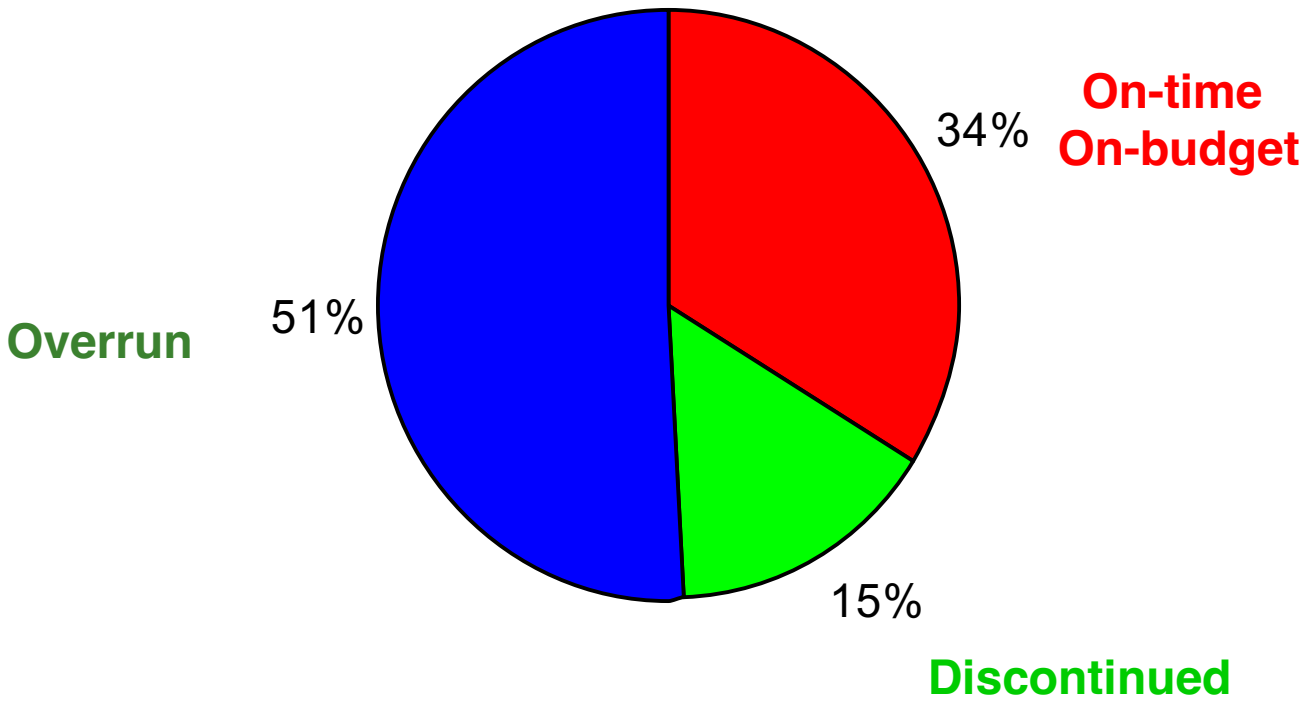
Software costs (cont'd)

- Software costs often dominate computer system costs
- Costs of software on a PC are often greater than the hardware cost
- Software costs more to maintain than it does to develop

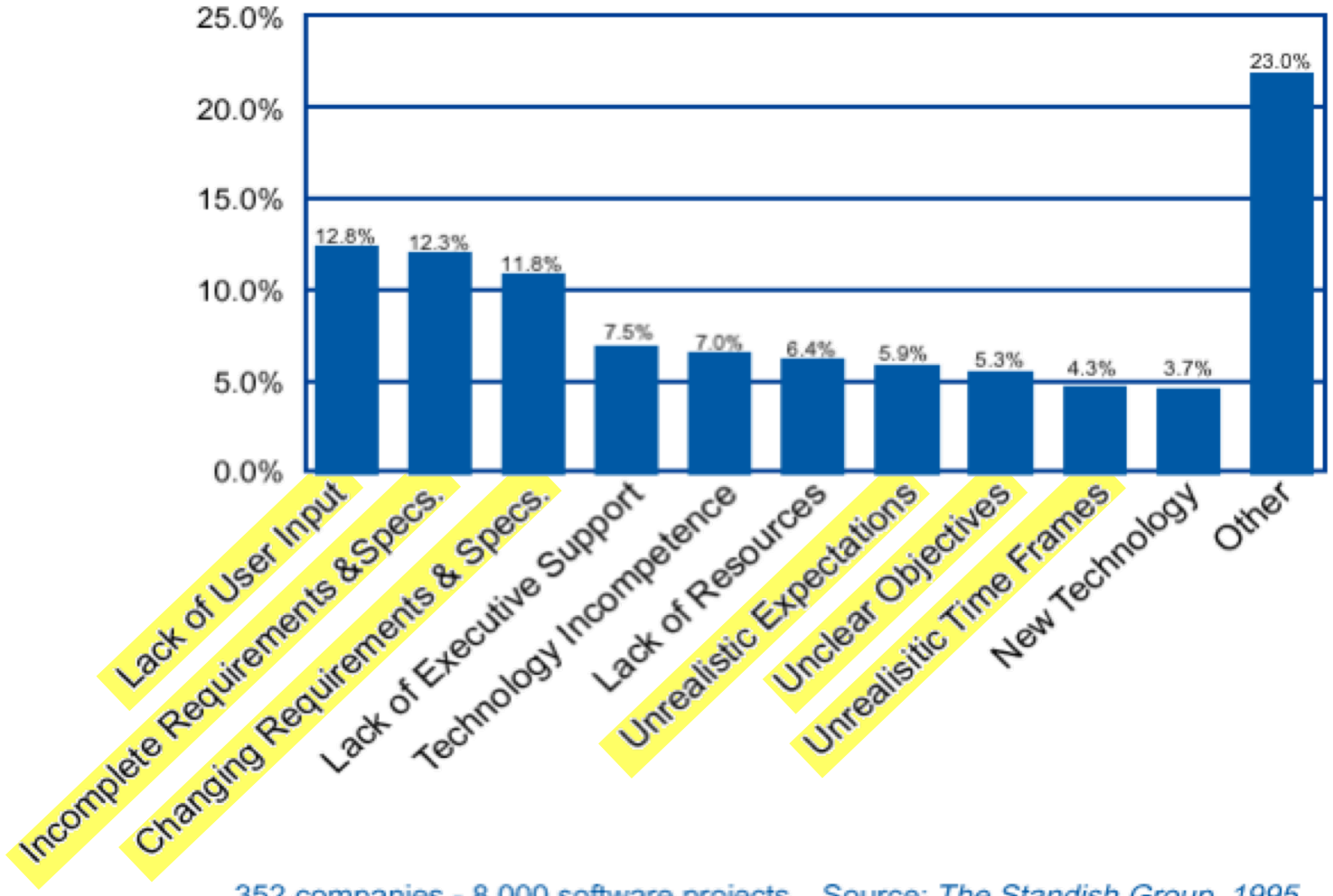
- Key objective of software engineering: **cost-effective software development**

Software Engineering Is Not Well-Practiced Today

-Standish Group CHAOS Report 2003



Why Software Projects Fail



352 companies - 8,000 software projects. Source: *The Standish Group, 1995*

Discussion

- Form groups of 3-5 each to discuss
  - What is software engineering?
  - Why is software engineering important?
  - What is software process?
  - What is the difference between software engineering and computer science?

  - **Note: do not search the Internet**

- Each group will present its answers
- Each has 5 minutes to discuss and 2 minutes to present

FAQs about software engineering

- What is software?
- What is software engineering?
- What is the difference between software engineering and computer science?
- What is the difference between software engineering and system engineering?
- What is a software process?
- What is a software process model?

FAQs about software engineering

- What are the costs of software engineering?
- What are software engineering methods?
- What is CASE (Computer-Aided Software Engineering)
- What are the attributes of good software?
- What are the key challenges facing software engineering?

What is software?

- Computer programs and associated documentation such as requirements, design models and user manuals.

- Software products may be
  - Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - Custom (bespoke) - developed for a single customer according to their specification.
- Software can be created by
  - by developing new programs
  - configuring generic software systems
  - reusing existing software.

What is software engineering?

- Software engineering is an engineering discipline that is concerned with theories, methods, tools for professional software development

- Goals
  - Cost effective (within budget)
  - On time
  - High quality
  - Satisfying customer's needs

Software engineering vs. Computer science?

- Computer science
  - concerned with theory and fundamentals
- Software engineering
  - concerned with the practicalities of developing and delivering useful software

- Computer science theories are still insufficient to produce successful software

Software engineering vs. System engineering?

- System engineering
  - concerned with all aspects of computer-based systems development including hardware, software and process engineering

- Software engineering is part of this process concerned with developing software

What is a software process?

- A set of activities whose goal is the development or evolution of software

- Generic activities in software processes
  - Specification - what the system should do and its development constraints
  - Development - production of the software system
  - Validation - checking that the software is what the customer wants
  - Evolution - changing the software in response to changing demands.

What is a software process model?

- A simplified representation of a software process, presented from a specific perspective

- Examples of process perspectives are
  - Workflow perspective - sequence of activities
  - Data-flow perspective - information flow
  - Role/action perspective - who does what
- Generic process models
  - Waterfall
  - Iterative development
  - Component-based software engineering

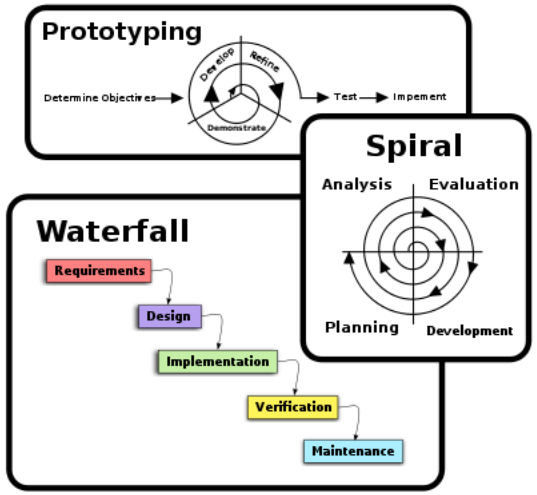
What are the costs of software engineering?

- Roughly 60% of costs are development costs, 40% are testing costs
- For custom software, evolution costs often exceed development costs
- Costs vary depending on many factors
  - Requirements, complexity, personnel, etc.

What are software engineering methods?

- Structured approaches to software development, including
  - system models, notations, rules, design advice and process guidance.

- Model descriptions
  - Descriptions of graphical models which should be produced;
- Rules
  - Constraints applied to system models;
- Recommendations
  - Advice on good design practice;
- Process guidance
  - What activities to follow.



What is CASE?

- CASE = Computer-Aided Software Engineering
  - Software systems that are intended to provide automated support for software process activities.

- CASE systems are often used for method support.
- Upper-CASE
  - Tools to support the early process activities of requirements and design;
- Lower-CASE
  - Tools to support later activities such as programming, debugging and testing.

What are the attributes of good software?

- Software should deliver the required functionality to the user
- It should be maintainable, dependable and acceptable

- Maintainability
  - Software must evolve to meet changing needs;
- Dependability
  - Software must be trustworthy;
- Efficiency
  - Software should not make wasteful use of system resources;
- Acceptability
  - Software must accepted by the users for which it was designed.
  - It must be understandable, usable and compatible with other systems.

What are the key challenges facing software engineering?

- Many, here are some:
  - Heterogeneity, delivery and trust.
  - Heterogeneity
    - Developing techniques for building software that can cope with heterogeneous platforms and execution environments;
  - Delivery
    - Developing techniques that lead to faster delivery of software;
  - Trust
    - Developing techniques that demonstrate that software can be trusted by its users.

Professional and ethical responsibility

- Software engineering involves wider responsibilities than simply the application of technical skills
- Software engineers must behave in an honest and ethically responsible way
- Ethical behavior is more than simply upholding the law

Issues of professional responsibility

- Confidentiality
  - Engineers should normally respect the confidentiality of their employers or clients
- Competence
  - Engineers should not misrepresent their level of competence
  - They should not knowingly accept work which is beyond their competence

Issues of professional responsibility

- Intellectual property rights
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc.
  - They should be careful to ensure that the intellectual property of employers and clients is protected
- Computer misuse
  - Software engineers should not use their technical skills to misuse other people's computers.

ACM/IEEE Code of Ethics

- ACM/IEEE provides Code of Ethics for software engineering processional
- Code of Ethics is used as a guidelines for SE professionals when making their decisions related to their actions

Code of ethics - principles

- PUBLIC
  - Software engineers shall act consistently with the public interest
- CLIENT AND EMPLOYER
  - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest
- PRODUCT
  - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible

Code of ethics - principles

- JUDGMENT
  - Software engineers shall maintain integrity and independence in their professional judgment
- MANAGEMENT
  - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance
- PROFESSION
  - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest

Code of ethics - principles

- COLLEAGUES
  - Software engineers shall be fair to and supportive of their colleagues
- SELF
  - Software engineers shall **participate in lifelong learning** regarding the practice of their profession and shall promote an ethical approach to the practice of the profession

Ethical dilemmas

- Disagreement in principle with the policies of senior management
- Your employer asks you to release a safety-critical system without thorough testing of the system

Key points

- Software engineering is an engineering discipline that is concerned with all aspects of software production
- Software products consist of developed programs and associated documentation
- Software process consists of activities that are involved in developing software products
- Software engineers have responsibilities to the engineering profession and society
  - They should not simply be concerned with technical issues