SOFTWARE METHODOLOGY

MI Qing (Lecturer)

Telephone: 15210503242

Email: miqing@bjut.edu.cn

Office: 410, Information Building

QR Code

群聊: Software Methodology - 2024

该二维码7天内(9月8日前)有效，重新进入将更新

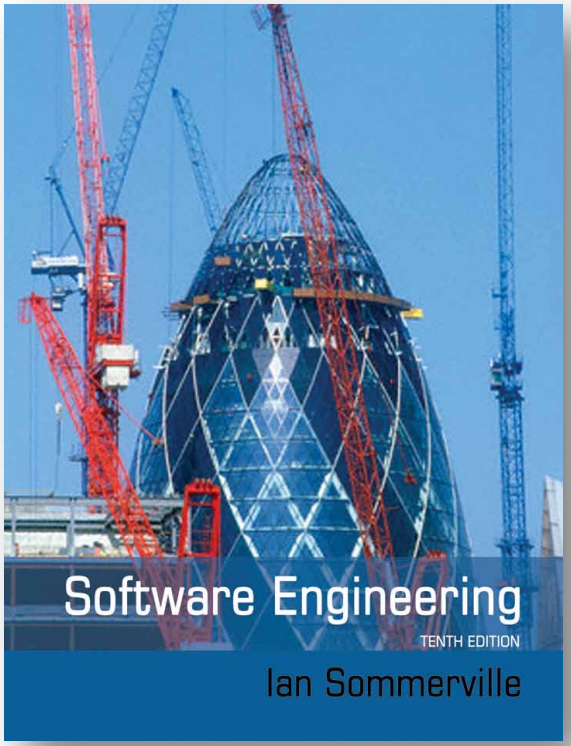
Biography

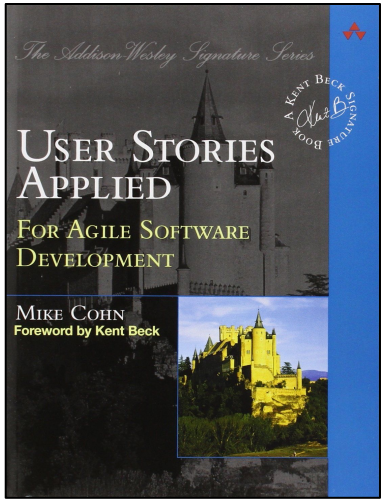
- ✧ Development Experience
  - Client Software Development Engineer in Tencent
  - Projects: SOSO Toolbar, Tencent TBH, WebGuard, IEHint Plugin, CoolGame
- ✧ Research Experience
  - Ph.D. in Software Engineering from the City University of Hong Kong
  - Research interests: code analysis, software quality assurance, deep learning and empirical experiments

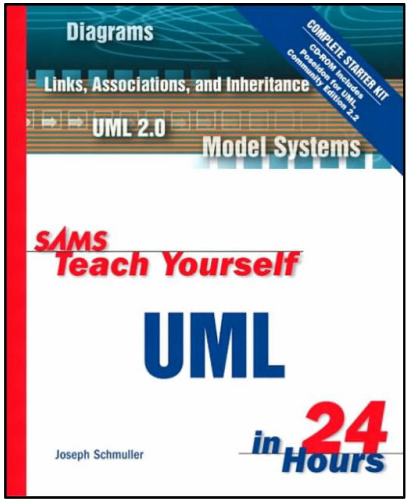
Textbook and Courseware

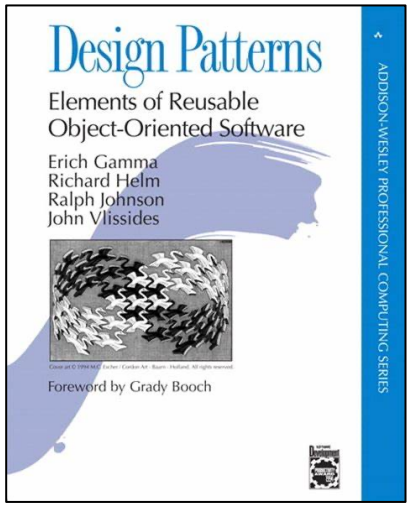
Available at:

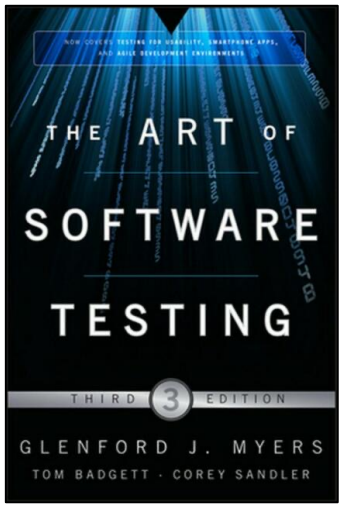
✧ 日新学堂-资料











Course Overview

- This course focuses on the methods, tools and techniques used in the development of software systems.
  - Basic concepts
  - Software processes
  - Agile software development
  - Requirements engineering
  - System modelling
  - Architectural design
  - Design and implementation
  - Software testing
  - Software evolution

Objective
Programmer → Engineer

Evaluation Criterions

Bonus: 3%

✧ Your final score consists of three parts:

- 10%: Attendance and performance in class
- 40%: Assignments (20% project/research and 20% after-class work)
- 50%: Examination

Project

You are required to design and implement an Android Diary APP that allows you to keep a record of your daily thoughts, feelings, and experiences.

Research

Anyone who can have his/her research paper published on CCF recommended conferences/journals will obtain full marks for the part of project/research. (https://www.ccf.org.cn/Academic_Evaluation/TCSE_SS_PDL/)

Any questions?

Chapter 1 - Introduction

MI Qing (Lecturer)

Telephone: 15210503242

Email: miqing@bjut.edu.cn

Office: 410, Information Building

Topics Covered

- ✧ History and development of software engineering
- ✧ Professional software development
  - What is meant by software engineering
- ✧ Software engineering ethics
  - A brief introduction to ethical issues that affect software engineering

History and development of software engineering

Past of Software

1949 - 1959

独立编程服务

代表：CSC

特点：为单个客户提供定制软件，包括技术咨询、软件编程和维护，软件销售是一次性的。

1959 - 1969

软件产品

代表：Informatics

特点：保护知识产权，以许可证的方式销售软件产品。

1969 - 1981

企业解决方案

代表：Oracle

特点：面向领域，以企业解决方案供应商面目出现。

1981 - 1994

面向大众的成套软件

代表：Microsoft

特点：基于个人计算，服务大众市场。

1994 - 至今

网络增值服务

代表：Google

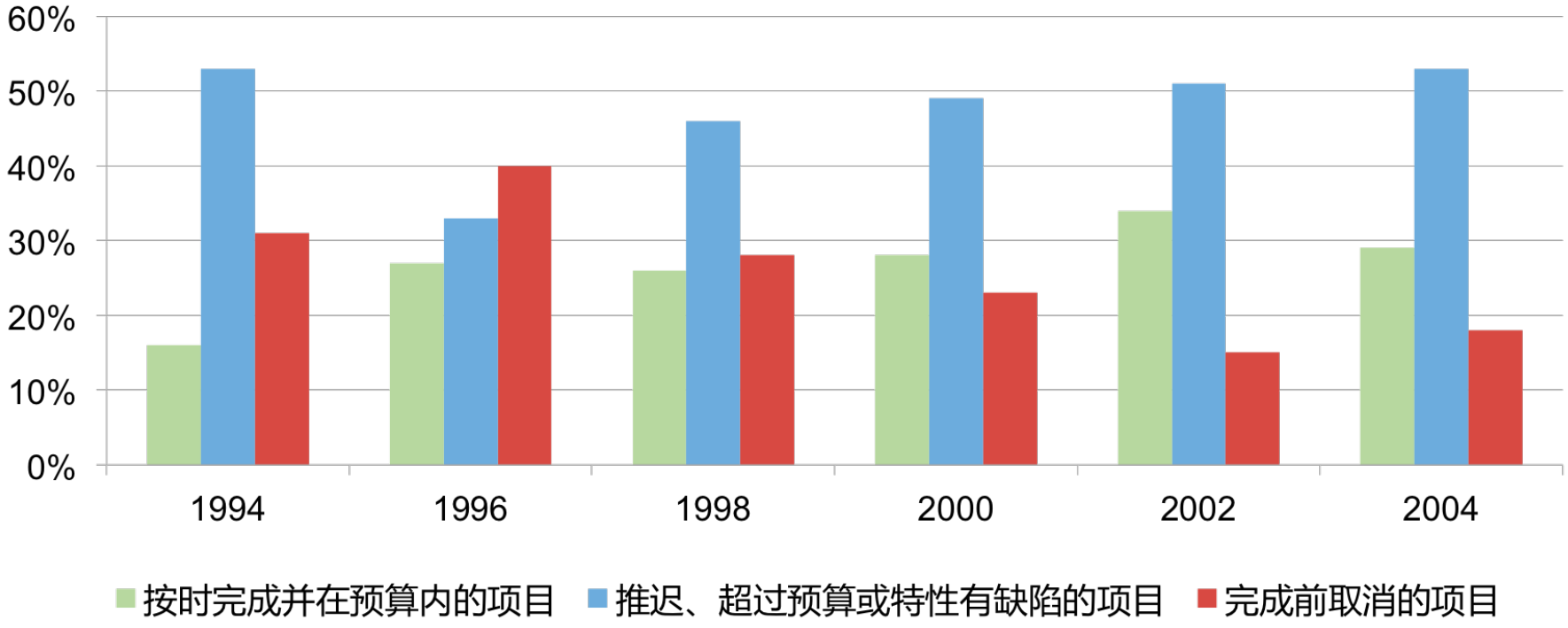
特点：向端用户收小头，向寄生在服务上的广告商收大头。

Present and Future of Software

- Present
  - The economies of ALL developed nations are dependent on software.
  - More and more systems are software controlled.
  - More than 75% of the world's population have a software-controlled mobile phone.
- Future

Key Software Development Trends

A Survey from Standish Group

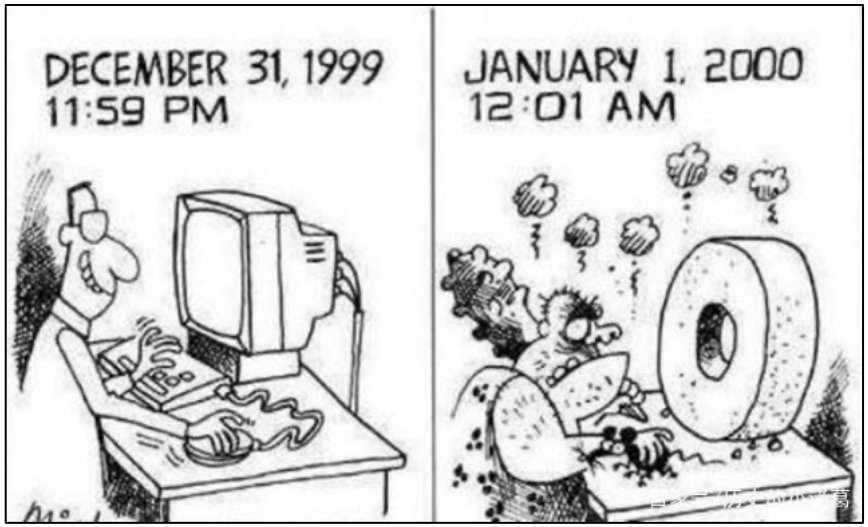


Example 1: The Ariane 5 Disaster

- The worst computer bugs in history
  - On June 4th, 1996, the Ariane 5 rocket ignited its engines. 37 seconds later, the rocket flipped 90 degrees in the wrong direction. This caused the self-destruct mechanism to trigger.
- Reason
  - The software attempted to stuff a 64-bit variable into a 16-bit integer.

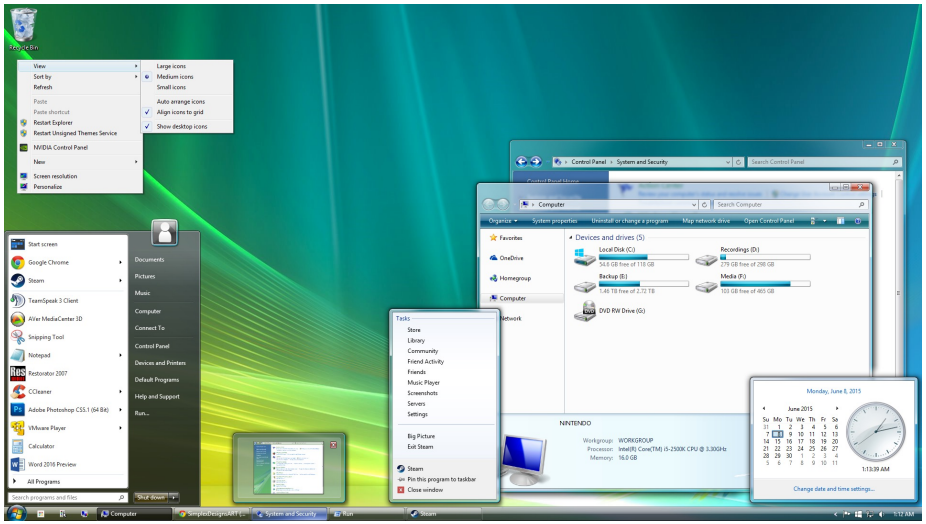
Example 2: The Y2K Bug

- ✧ What is the Y2K bug
  - Y2K bug is a technical flaw related to the computer systems and computer networks that were projected to created havoc as the year changed from 1999 to 2000.
- ✧ Reason
  - When complex computer programs were first written in the 1960s, engineers used a two-digit code for the year, leaving out the "19".



Example 3: Windows Vista

- 该系统从2001年开始研发，整个过程历时5年半，先后有9000位开发人员投入其中，耗资60亿美元，代码规模超过5000万行。
- 按照微软公司最初的计划，该系统面世时间应该在2003年，之后推迟到2004年下半年再到2005年初，最终在取消一些高级功能后于2006年11月正式发布。
- 从Vista Beta 1进入公开测试以来，程序错误总数已经超过2万个，这其中还不包括微软内部未公开的一些错误。



The Nature of Software

软件特性，**填空**

- Brooks has famously observed that four properties of software, taken together, differentiate it from other kinds of engineering artifacts. These four properties are:
  - Conformity (一致性)
  - Invisibility (不可见性)
  - Complexity (复杂性)
  - Changeability (演化性)
- Brooks, F. P., “No silver bullet: essence and accidents of software engineering”, IEEE Computer , Vol. 20, No. 4, pp.10-19, 1987

Conformity

- Software, unlike a physical product, has no underlying natural principles which it must conform to, such as Newton's laws of motion. However, software must conform to exacting specifications in the representation of each of its parts, in the interfaces to other internal parts, and in the connections to the environment in which it operates.
- Lack of conformity can cause problems when an existing software component cannot be reused as planned because it does not conform to the needs of the product under development.

Invisibility

- Software is said to be invisible because it has no physical properties. Work products such as requirements specifications, design documents, source code and object code are representations of software, but they are not the software.
- Software project manager cannot view the progress of the project due to the invisibility of the software until it is completely developed. Thus the invisibility causes a major problem to the complexity of managing a software project.

Complexity

- The complexity of software arises from the large number of unique interacting parts in a software system. Software parts have several different kinds of interactions, including serial and concurrent invocations, state transitions, data couplings, and interfaces to databases and external systems.
- Complexity can hide defects that may not be discovered easily, thus requiring significant additional and unplanned rework.

Changeability

- Software coordinates the operation of physical components and provides most of the functionality in software-intensive systems. Because software is the most malleable (easily changed) element in a software-intensive system, it is the most frequently changed element.
- However, this does not mean that software is easy to change. Complexity and the need for conformity can make changing software an extremely difficult task.

Example

- 某公园有一个游船码头，负责人希望开发一款游船管理系统，要求如下：当游客租船时，管理员输入S表示租船周期开始；当游客还船时，管理员输入E表示租船周期结束。一天结束时，要求系统打印出租船次数和平均租船时间。

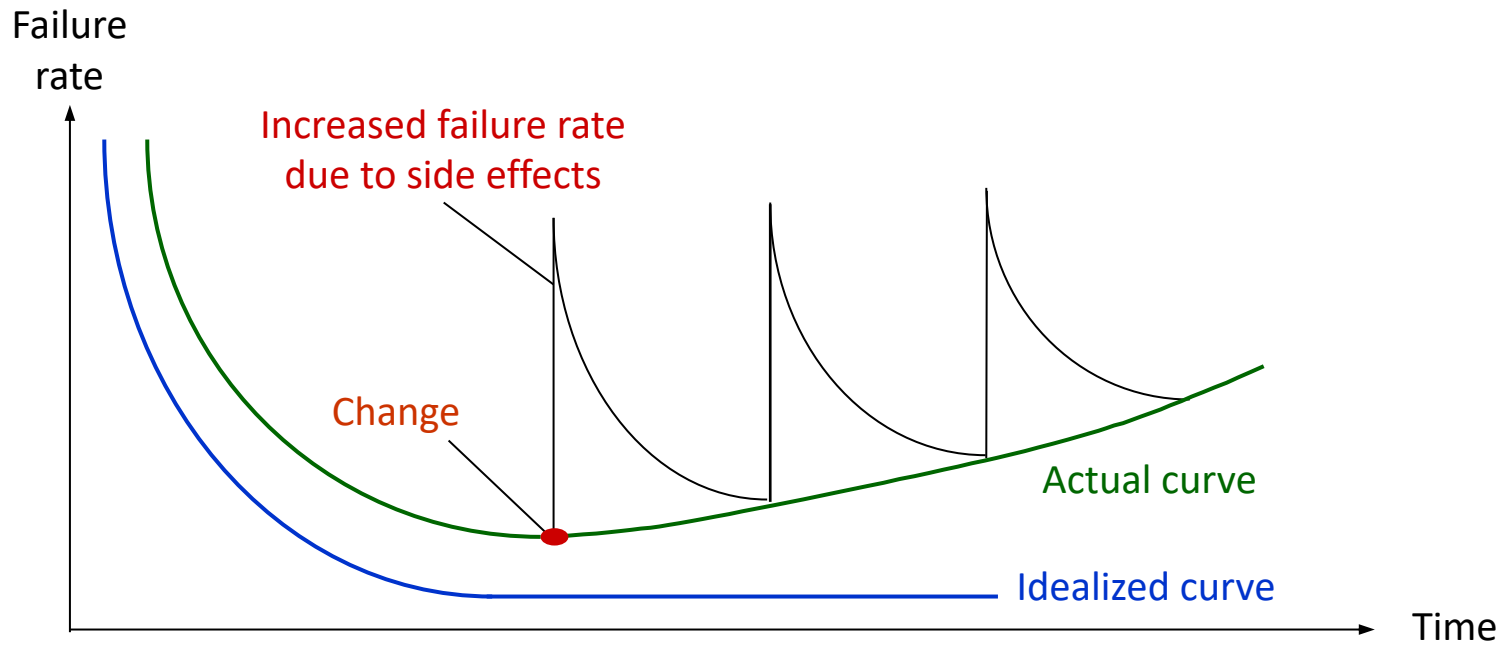
```
Number = Total_time = 0;
Get Message;
While ( !End_of_stream ) {
  if (Code == S) {
    Number ++;
    Total_time  -= Start_time; }
  else  Total_time  +=  End_time;
  Get Message; }
Print Number;
If (Number)  Print (Total_time / Number);
```

输出一天中的最长租用时间

将报告分上午和下午输出

当通信线路出问题时，能从计算中删除一切不完整的租船信息

Changeability



Software Project Failure

✧ Increasing system complexity

- As new software engineering techniques help us to build larger, more complex systems, the demands change. Systems have to be built and delivered more quickly; larger, even more complex systems are required; systems have to have new capabilities that were previously thought to be impossible.

✧ Failure to use software engineering methods

- It is fairly easy to write computer programs without using software engineering methods and techniques. Many companies have drifted into software development as their products and services have evolved. They do not use software engineering methods in their everyday work. Consequently, their software is often more expensive and less reliable than it should be.

Software Engineering

✧ The notion of software engineering was first proposed in 1968 at a conference by Fritz Bauer.

- 软件开发没有方法可循
- 软件设计是在开发人员头脑中完成的隐藏过程
- 60世纪中期的软件危机

史前时代

1956 - 1967

- 1968年提出软件工程
- 结构化开发方法
- 瀑布式软件生命周期模型成为典型

瀑布过程模型

1968 - 1982

- 面向对象开发方法
- 软件过程改进运动
- CMM/ISO9000/SPICE等质量标准体系

质量标准体系

1983 - 1995

- 敏捷开发方法流行
- 更紧密的团队协作
- 有效应对需求变化
- 快速交付高质量软件
- 迭代和增量开发

20世纪90年代至今

Professional software development

Individual Programming VS Professional Software Development

✧ Output

- Individual Programming → Program
- Professional Software Development → Software

✧ User

- The key distinctions are that professional software is intended for use by someone apart from its developer. It is maintained and changed throughout its life.

Software is not just the programs themselves but also all associated documentation, libraries, support websites, and configuration data that are needed to make these programs useful.

Software engineering is intended to support professional software development rather than individual programming.

Software Products

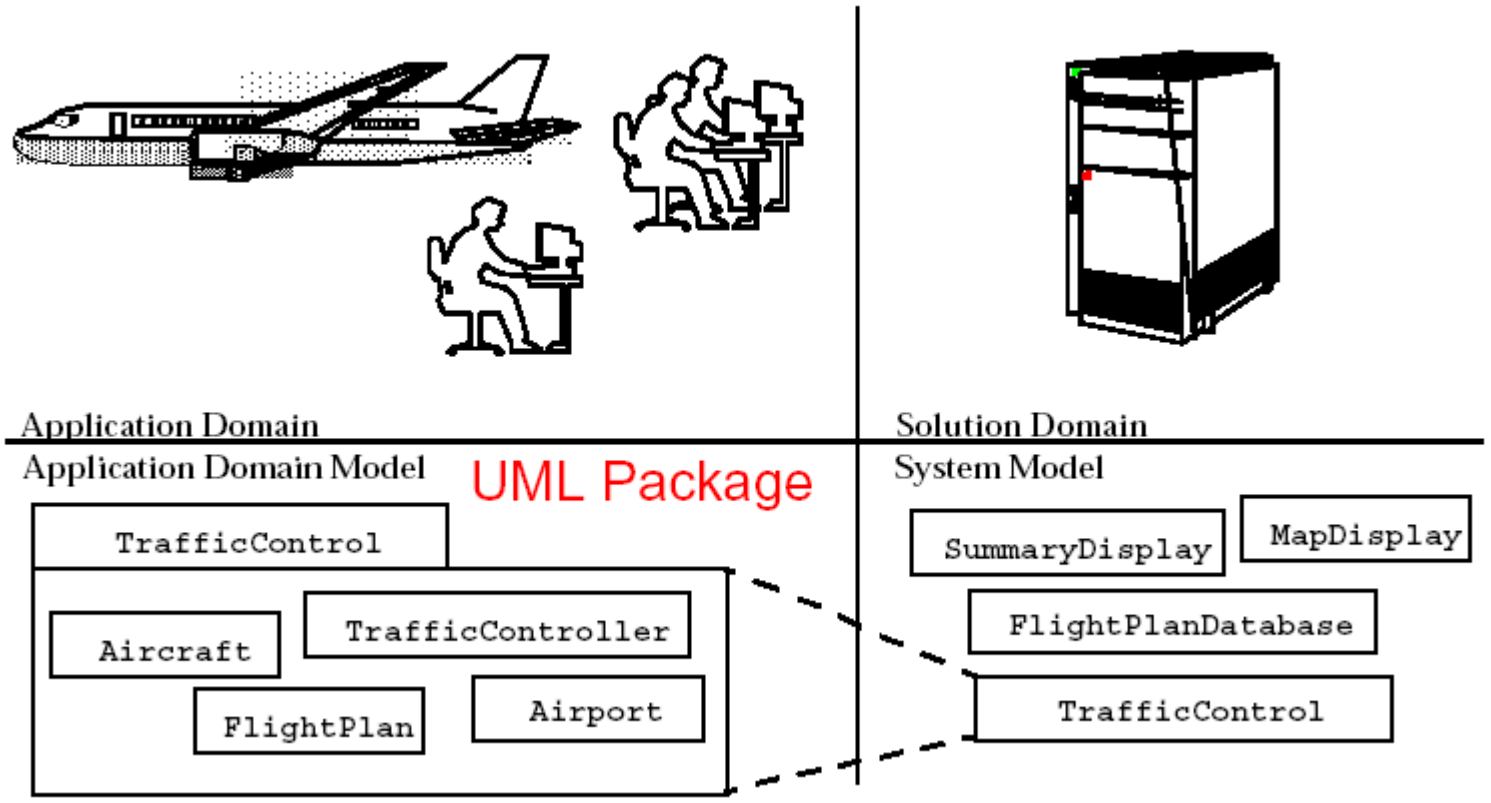
软件产品两种类型的区别

- ✧ Generic products (e.g., PC software such as Photoshop)
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
  - The specification of what the software should do is owned by the software developer and decisions on software change are made by the developer.
- ✧ Customized products (e.g., air traffic control software)
  - Software that is commissioned by a specific customer to meet their own needs.
  - The specification of what the software should do is owned by the customer for the software and they make decisions on software changes that are required.

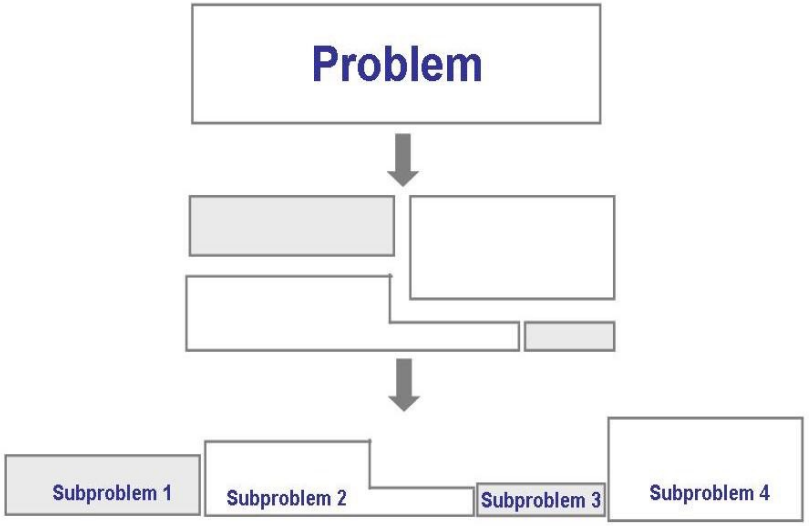
Definition of Software Engineering

- Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use.
- Engineering discipline
  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.
- All aspects of software production
  - Not just technical process of development. Also project management and the development of tools and methods to support software production.

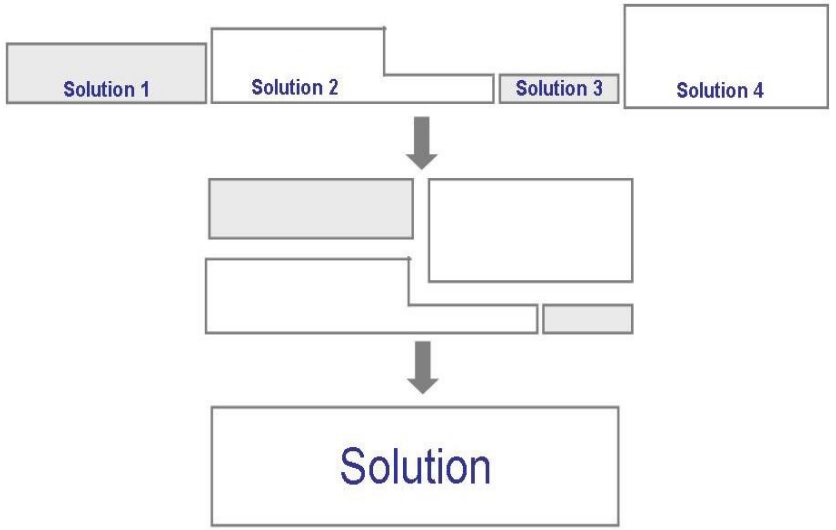
Software Engineering



Software Engineering



分解：将复杂的问题分解成可理解并能够处理的若干小问题进行研究和分析。



合成：在分析的基础上，针对各个不同部分给出解决方案，并将这些小构造块组合成一个大系统。

Importance of Software Engineering

- More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly.
- It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project. For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

Software Process Activities

软件过程四个活动

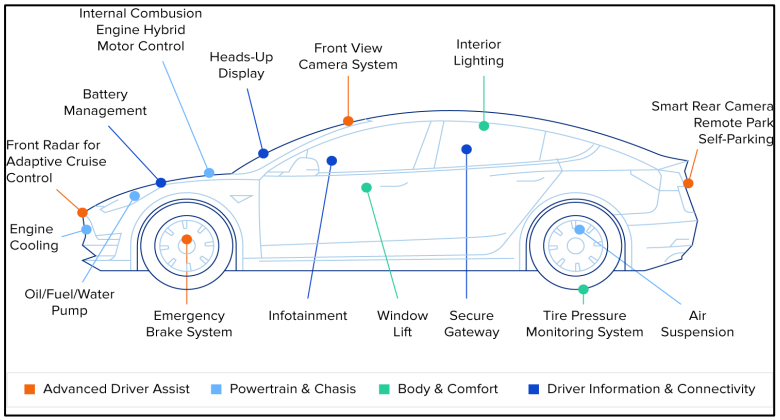
- Software specification, where customers and engineers define the software that is to be produced and the constraints on its operation.
- Software development, where the software is designed and programmed.
- Software validation, where the software is checked to ensure that it is what the customer requires.
- Software evolution, where the software is modified to reflect changing customer and market requirements.

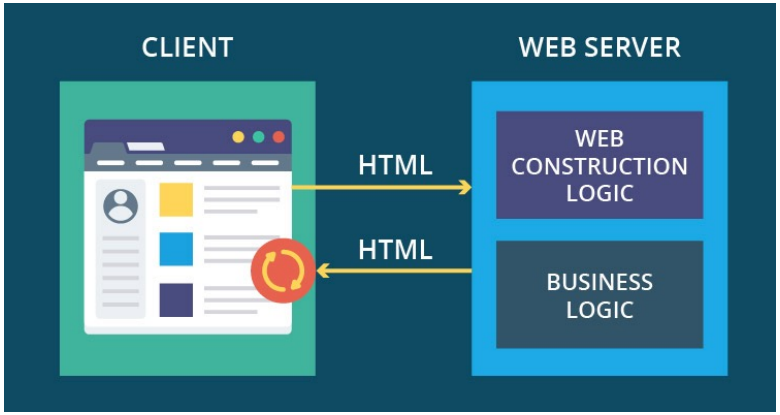
Software Engineering Diversity

- There are many different types of software system and there is no universal set of software techniques that is applicable to all of these.
- The software engineering methods and tools used depend on the type of application being developed, the requirements of the customer and the background of the development team.

Example

- An embedded control system in an automobile is safety-critical and is burned into ROM (read-only memory) when installed in the vehicle. It needs extensive verification and validation before delivery.
- For an interactive web-based system or app, iterative development and delivery is the best approach.

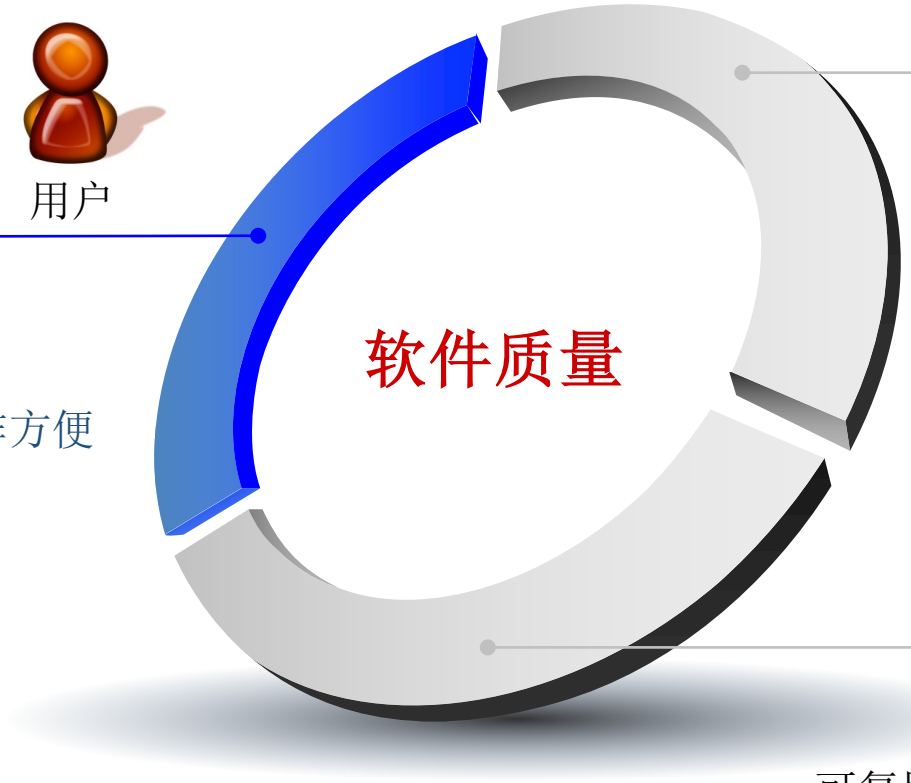




Software Engineering Fundamentals

- Some fundamental principles apply to all types of software system, irrespective of the development techniques used:
  - Systems should be developed using a managed and understood development process. Of course, different processes are used for different types of software.
  - Dependability and performance are important for all types of system.
  - Understanding and managing the software specification and requirements (what the software should do) are important.
  - Where appropriate, you should reuse software that has already been developed rather than write new software.

What makes a good software?



开发人员

结构质量

代码可测试性
代码可维护性
代码可读性
代码效率：高效管理资源
代码安全：可预防常见威胁

用户

功能质量

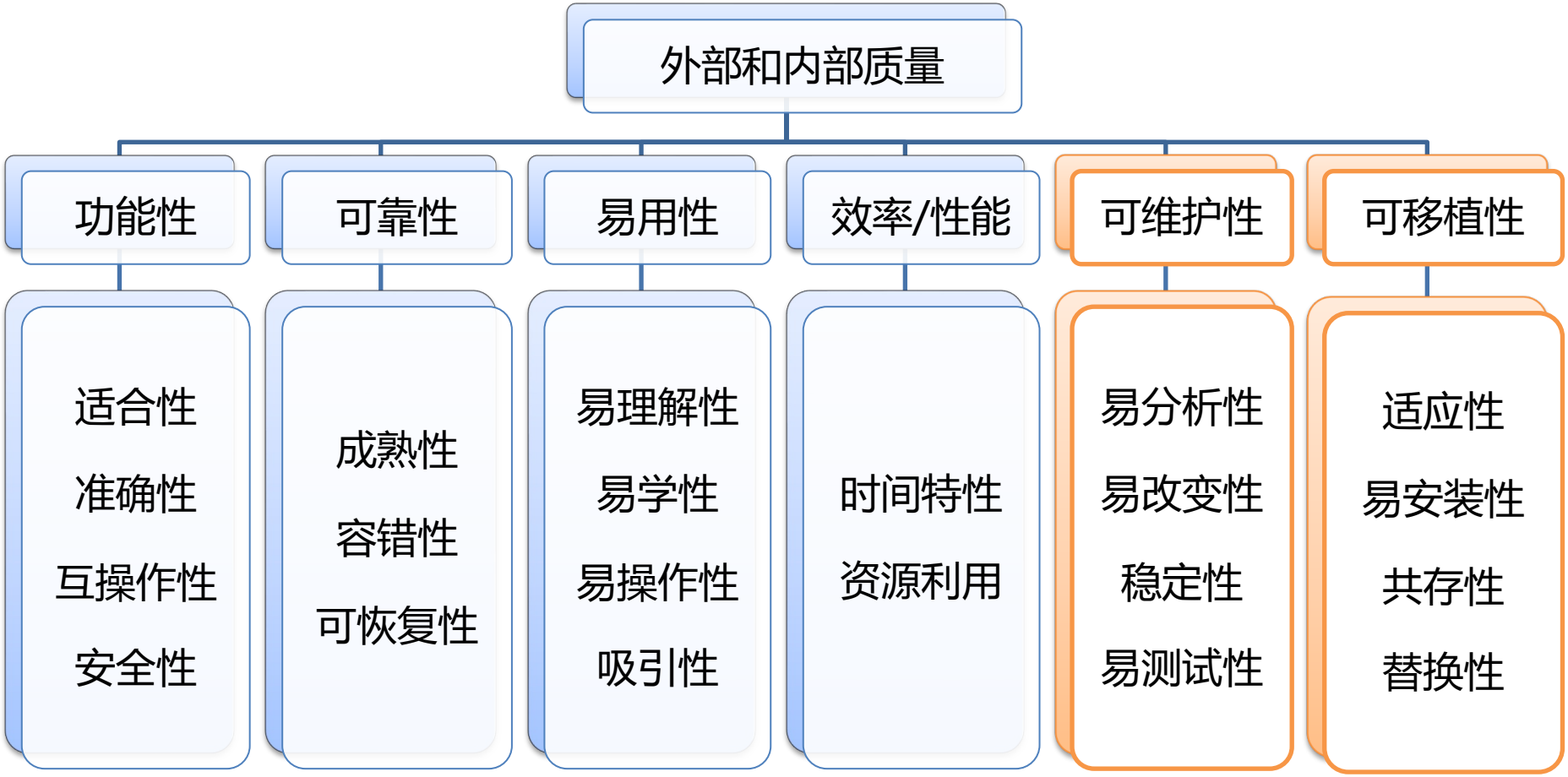
软件符合指定需求
软件几乎没有缺陷
软件性能正常
软件容易上手，操作方便

投资者

过程质量

软件按时交付
软件满足预算
可复用的开发过程，确保交付质量

ISO 9126 Software Quality Characteristics



质量特性

ISO 9126 Software Quality Characteristics

- Functionality: Are the required functions available in the software?
- Reliability: How reliable is the software?
- Usability: Is the software easy to use?
- Efficiency: How efficient is the software?
- Maintainability: How easy is to modify the software?
- Portability: How easy is to transfer the software to another environment?

Software engineering ethics

Software Engineering Ethics

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of Professional Responsibility

✧ Confidentiality

- Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.

✧ Competence

- Engineers should not misrepresent their level of competence. They should not knowingly accept work which is outwith their competence.

Issues of Professional Responsibility

✧ Intellectual property rights

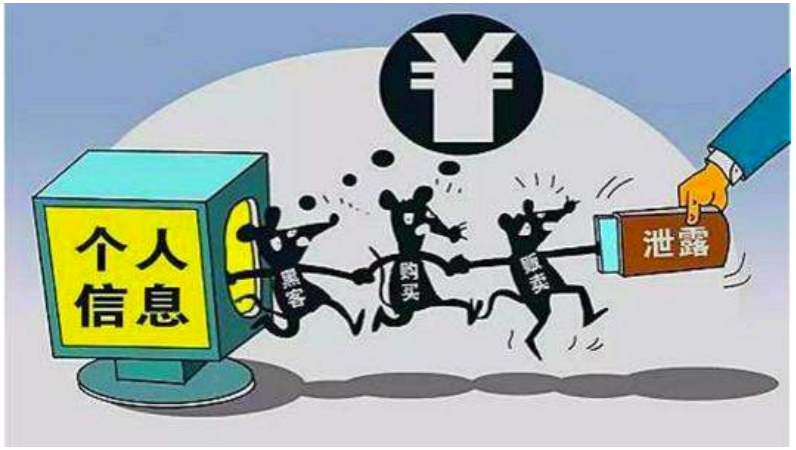
- Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.

✧ Computer misuse

- Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses).

Examples

微盟程序员删库跑路



某些APP侵害用户权益

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Ethical Principles

1. PUBLIC - Software engineers shall act consistently with the public interest.
2. 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.
3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.
8. 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.

Summary

Frequently asked questions about SE

Question 1:

✧ What is software?

Answer:

✧ Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

Frequently asked questions about SE

Question 2:

✧ What are the attributes of good software?

Answer:

✧ Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.

Frequently asked questions about SE

Question 3:

✧ What is software engineering?

Answer:

✧ Software engineering is an engineering discipline that is concerned with all aspects of software production.

Frequently asked questions about SE

Question 4:

✧ What are the fundamental software engineering activities?

Answer:

✧ Software specification, software development, software validation and software evolution.

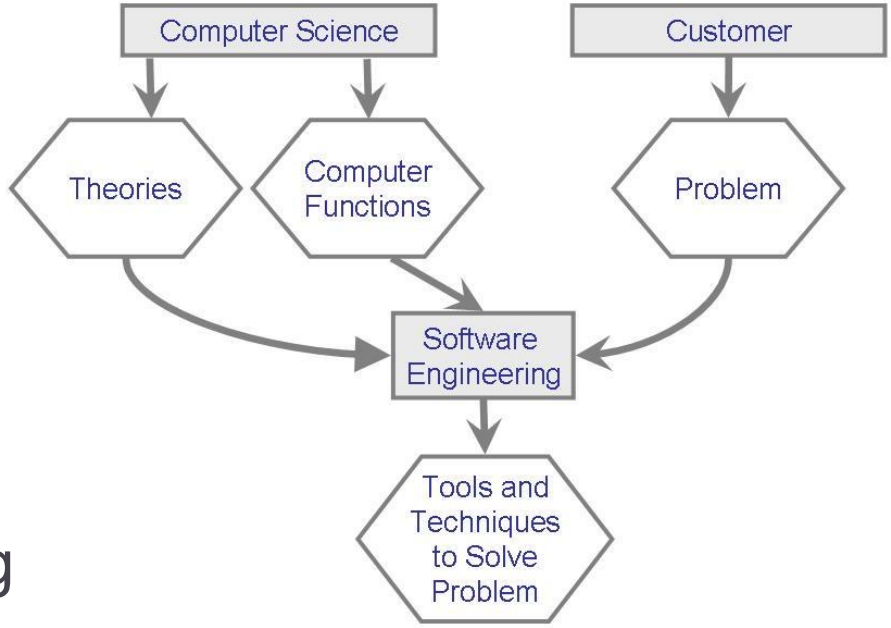
Frequently asked questions about SE

Question 5:

- What is the difference between software engineering and computer science?

Answer:

- Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.



Frequently asked questions about SE

Question 6:

✧ What are the key challenges facing software engineering?

Answer:

✧ Coping with increasing diversity, demands for reduced delivery times and developing trustworthy software.

Frequently asked questions about SE

Question 7:

✧ What are the costs of software engineering?

Answer:

✧ Roughly 60% of software costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development costs.

Frequently asked questions about SE

Question 8:

- What are the best software engineering techniques and methods?

Answer:

- While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.