華南理工大學
South China University of Technology

Intro to Computer Science and Software Engineering

Software Engineering

Dr Yubei Lin
yupilin@scut.edu.cn
School of Software Engineering

Software Engineering

- Why Engineering?
  - Art or/and Science
- Considering a complex system?
  - Lots of programmers
  - Long time
  - Complex requirements
  - … …
- **Software crisis (软件工程危机), 1960s**

Software Engineering

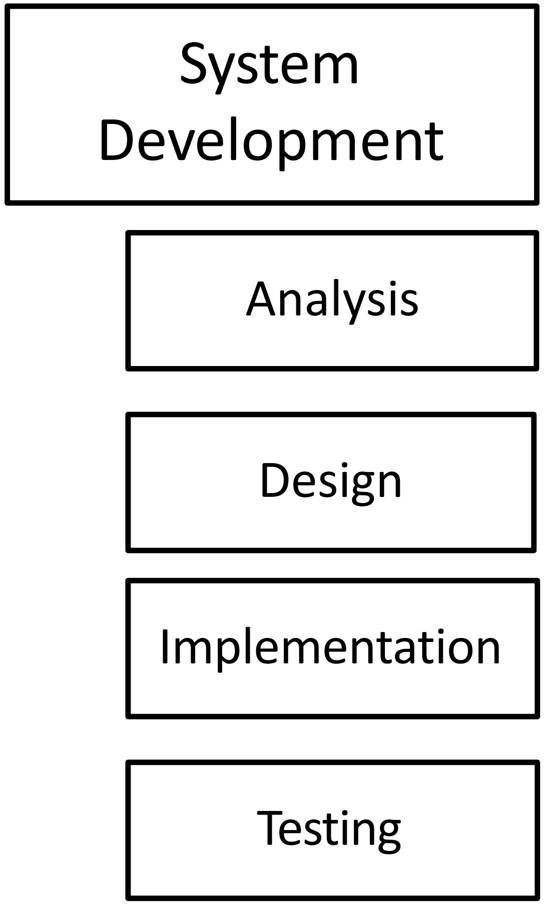
- Software engineering is the application of engineering to the development of software in a systematic method.
- The establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines.

Software Engineering

- Software life cycle, a system goes through a cycle of repeating phases
  - The development phase
  - The use phase
  - The modification phase
  - The obsolete phase

The development process

- Four phases:
  - Analysis
  - Design
  - Implementation
  - Testing



The analysis phase

- Defining the users and their needs
  - Very difficult~
- Defining the requirements of the system and specified the methods
  - Some needs of users might not be satisfied by software systems
- Analysis can be viewed as the process of identified the gap of users’ needs and the requirements of the system

The design phase

- Defining how the system will accomplish what was defined in the analysis phase
  – Persistence: files and/or database
- Modularity is the well-established principle!
  – The system is divided into small modules with each being designed and tested!
- Tools, e.g. structure chart, is commonly used.
  – A module ~ a logical step in a structure chart
  – The interaction between modules

The implementation phase

- Tools might be used to show the logical flow of the programs before the actual writing of code.
  - The flowchart
  - The pseudocode
- Coding: actually writing the code in language specific for the project.

The testing phase

- Can be a tedious and time-consuming part of program development.
  - Really?
- Two type of testing
  - Black box testing
    - usually by the test engineers
    - Against the requirements
  - White box testing
    - usually by the programmers
    - to test every instructions and every possible suitations

Development process models

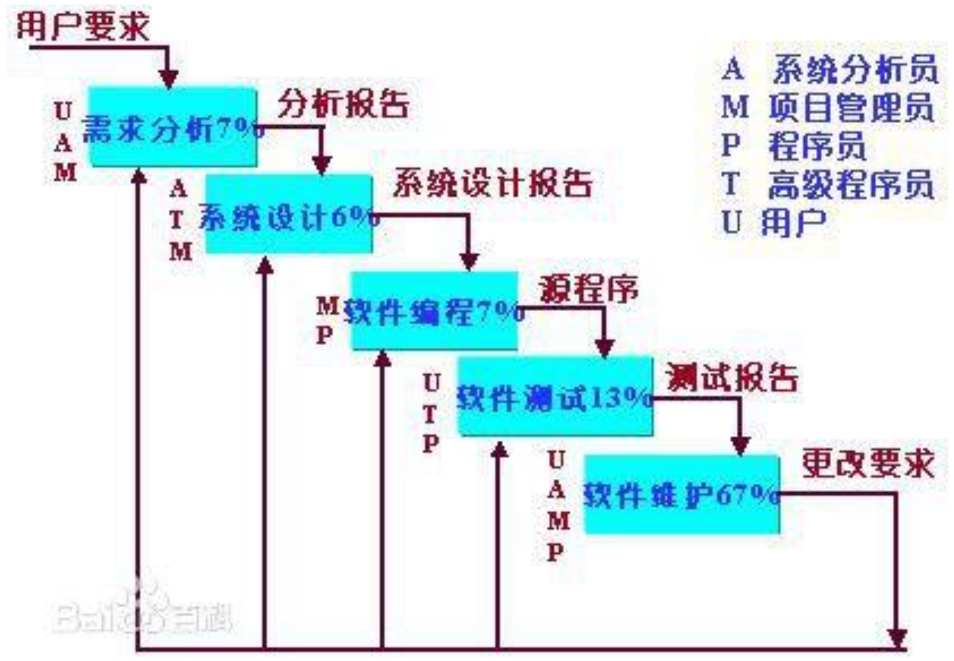
- A splitting of development work into distinct phases (or stages) containing activities with the intent of better planning and management.
  - to find repeatable, predictable processes that improve productivity and quality.
- A variety of such frameworks have evolved over the years, each with its own recognized strengths and weaknesses.
- One software development methodology framework is not necessarily suitable for use by all projects.

Development process models

- Most common
  - Prototyping
  - Waterfall model
  - Incremental model
  - Rapid application development
  - Extreme programming
  - … …

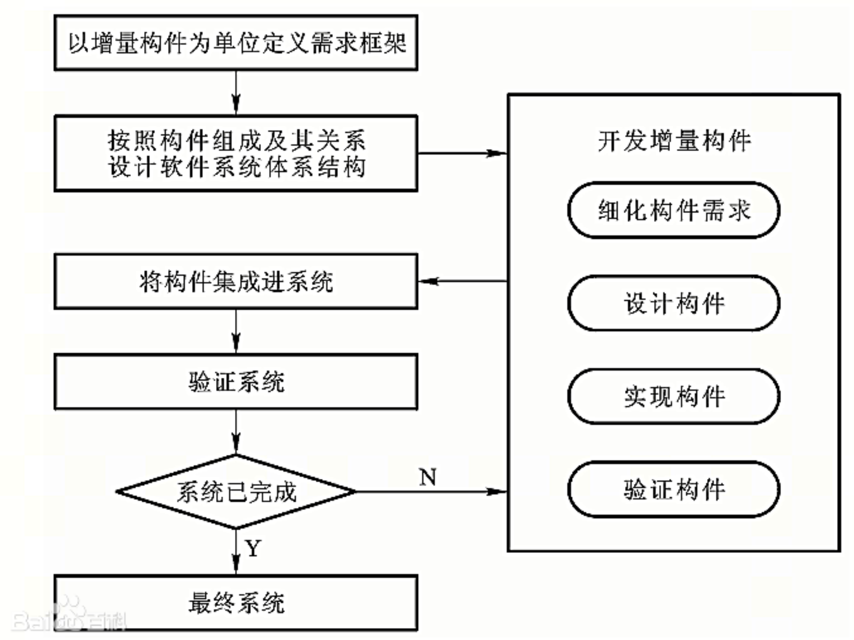
The waterfall model

- The development process flows in only one direction, that is a phase cannot be started until the previous phase is completed.
  - Advantages: each phase must be well accomplished!!
  - Disadvantages: the difficulty in locating a problem



The incremental model

- The process is developed in a series of steps
  - Each step completes parts of the system, also called an increment
  - Commonly the first increment completes the main part (or module)
  - The remain increment continuously add all submodules
- Locating a problem in a module is considered easier than in the whole system



Agile Development Model

- The agile development model emphasizing the results of the software engineering effort over the process of getting to the results.
- The traditional approach of investigation and analysis, design, development, implementation, and management is deemphasized (淡化).
- In contrast, the agile approach emphasizes construction and delivery (交付).

Agile Development Model

- Agile development calls for small, highly talented, highly responsive teams that construct software in small increments, focusing on the essential requirements.
- The chief architects of the methodology have articulated 12 principles that govern the agile development.

Agile Development Model

1. Place the highest priority on customer satisfaction through early and continuous delivery of valuable software systems.
2. Welcome changing requirements that enhance the customer's competitive, irrespective of the stage in the development.
3. Deliver working software frequently, and within a short timeframe.

Agile Development Model

4. Get the business people and the software developers to work together on a consistent basis throughout the project.
5. Build projects around motivated individuals. Give them the required resources and trust them to get the job done.
6. The most efficient and effective method of communication
7. within a software engineering team is face- to-face conversation.

Agile Development Model

7. The primary measure of progress and success is a working software system or component.
8. The agile development process promotes sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. There should be continuous attention to technical excellence and good design.

Agile Development Model

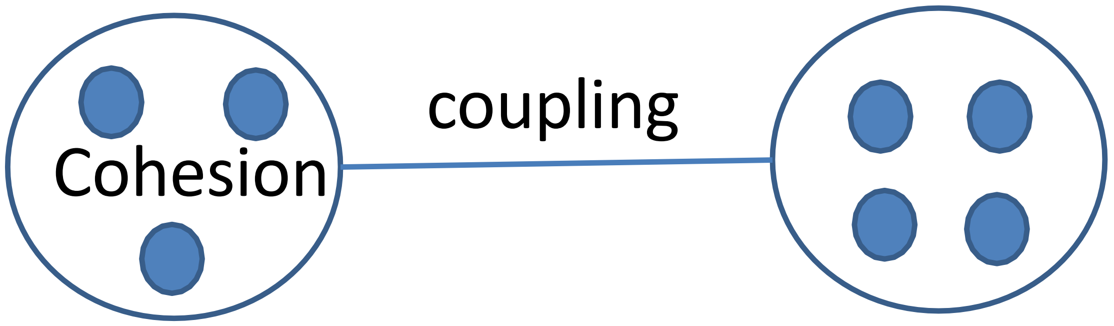
10. There should be great emphasis on simplicity as an essential means of maximizing the amount of work not done.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. The software engineering team should periodically reflect on how to become more effective, and then refine its behavior accordingly.

Modularity

- Modularity means to divide a larger program into small parts that can communication with each other.
- Tools used in achieving modularity
  - Structure chart in procedural programming
  - Class diagram in object-oriented programming, e.g. Unified Modeling Language (UML)

Coupling and Cohesion

- Coupling (耦合) is a measure of how tightly two modules are bound to each other.
  - 模块之间
- Cohesion (内聚) is a measure of how closely the processes in a module are related.
  - 模块之内



Coupling

Data coupling 数据耦合

Stamp coupling 标记耦合

Control coupling 控制耦合

Global coupling 公共耦合

Content coupling 内容耦合

Loose coupling is desired

- Independent ~ reusable
- Low possibility of error propagation
- Easy maintenance

loose 弱

tight 强

Cohesion

Functional Cohesion 功能内聚

Only one thing in on place

Sequential Cohesion 顺序内聚

Output of the former process is the input of the following process

Communicational Cohesion 通信内聚

Processes work on the same data

Procedural Cohesion 过程内聚

Combine unrelated processes by a control flow

Temporal Cohesion 时间内聚

Combine unrelated processes that always occur together

Logical Cohesion and Coincidental unrelated processes 逻辑/偶然内聚

High Cohesion is desired

high 高

low 低

Documentation

- For a software to be used properly and maintained efficiently, documentation is needed, and is very important!
- Two separate sets:
  - User documentation
  - System documentation
- There should be system documentation for all four phase of systems development
  - Analysis, design, implementation and testing

Quality

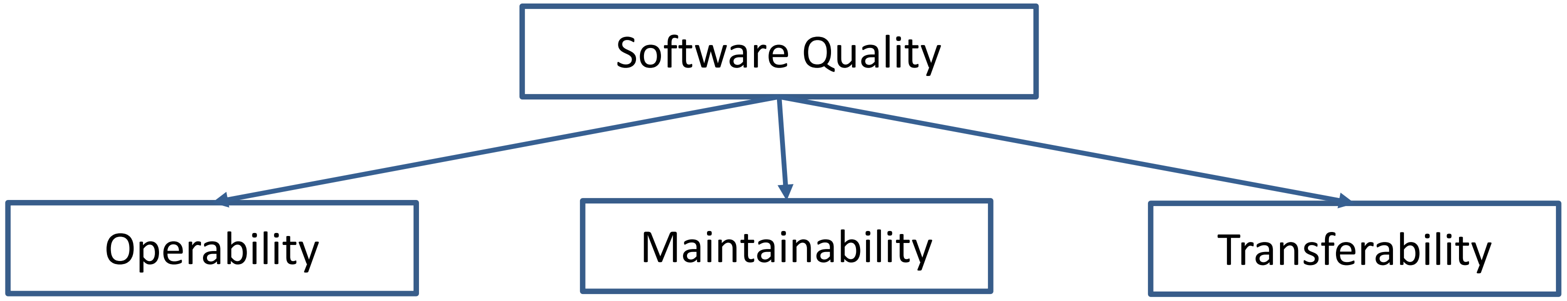
- Quality software is defined as

Software that satisfies the user's explicit and implicit requirements , is well documented, meets the operating standards of the organization, and runs efficiently on the hardware for which it was developed.

Quality software is what we want, but no one of them is without a flaw or two!!

Quality factors

Two types: quantitative and qualitative



可操作性

Accuracy 正确性
Efficiency 性能
Reliability 可靠性
Security 安全性
Timeliness 及时性
Usability 可用性

可维护性

Changeability 易变性
Correctability 修复性
Flexibility 弹性
Testability 可测试性

可转移性

Code reusability 重用性
Interoperability 交互性
Portability 可移植性

Desirable Features of Computer Software

- Maintainability 可维护性: How easily maintained is the software?
  - This will depend on the quality of the design as well as the documentation.
- Documentation文档: How well documented is the system?
- Efficiency 效率: How efficiently are certain core operations carried out? Of importance are the response time and the system throughput.

Desirable Features of Computer Software

- User Friendliness用户友好性: How well designed is the user interface? How easy is it to learn and use the system?
- Compatibility兼容性: with existing software products.
- Security安全性: Are there mechanisms to protect and promote confidentiality and proper access control?
- Integrity完整一致性: What is the quality of data validation methods?

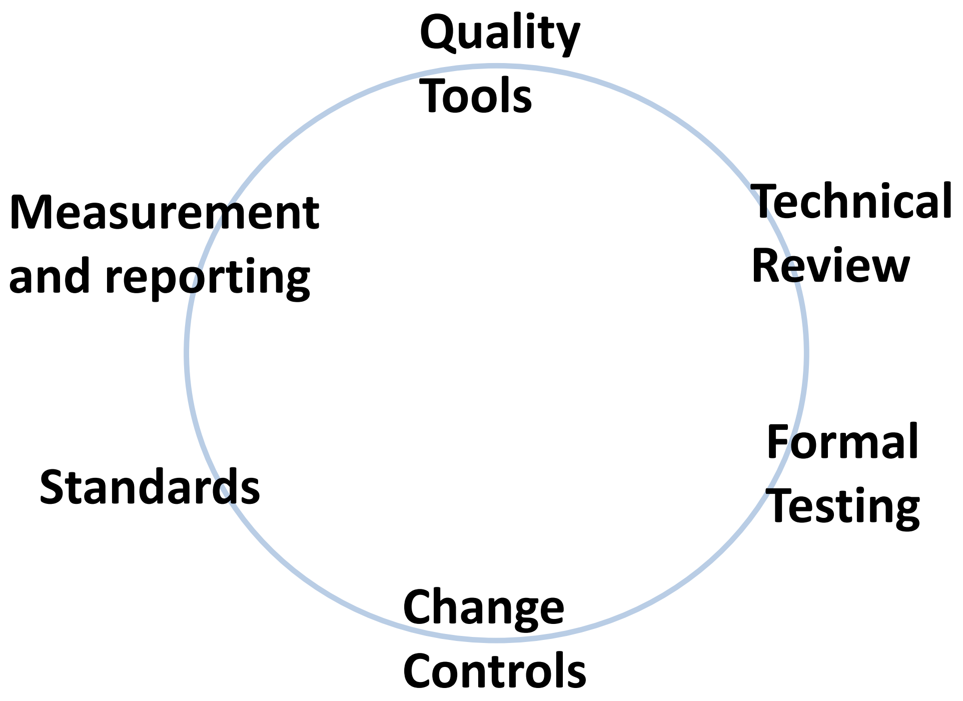
Desirable Features of Computer Software

- Reliability可靠性: Will the software perform according to requirements at all times?
- Growth potential(增长潜力,可扩展性): What is the storage capacity? What is the capacity for growth in data storage?
- Functionality and Flexibility(功能性和灵活性): Does the software provide the essential functional features required for the problem domain? Are there alternate ways of doing things?

Desirable Features of Computer Software

- Differentiation(差异化): What is unique about the software?
- Adaptability(适应性): How well are unanticipated situations handled?
- Productivity(生产力): How will productivity be affected by the software system?
- Comprehensive Coverage (综合保障): Is the problem comprehensively and appropriately addressed?

The quality circle


Quality Tools
Technical Review
Formal Testing
Change Controls
Standards
Measurement and reporting