Chpater1

An Introduction to Software Engineering

Objectives

- To introduce software engineering and to explain its importance
- To set out the answers to key questions about software engineering
- To introduce ethical and professional issues and to explain why they are of concern to software engineers

Why Software engineering?

- Software is important, is soul of computer-based system.

  Software is indispensable in almost each domain.

  It affects nearly every aspect of our lives and has become pervasive in our commerce, our culture, and our everyday activities.

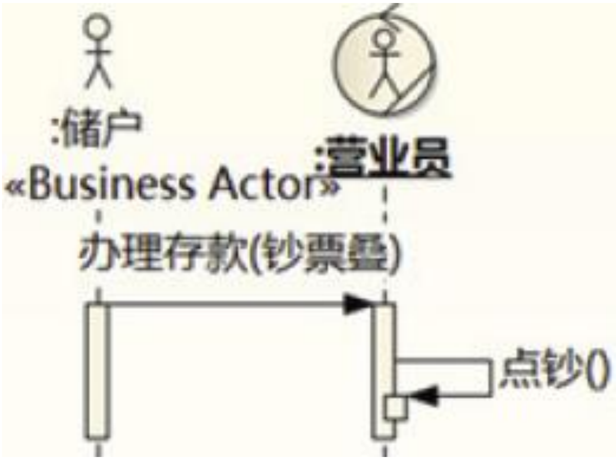
  law of unintended consequences

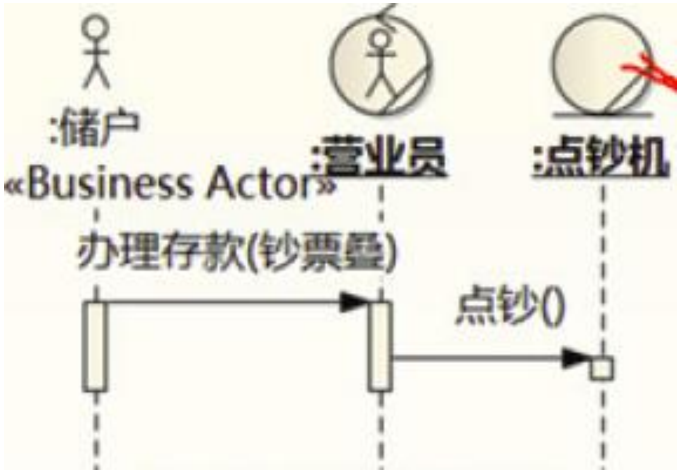
Review

- Software plays a key role in nearly each fields.

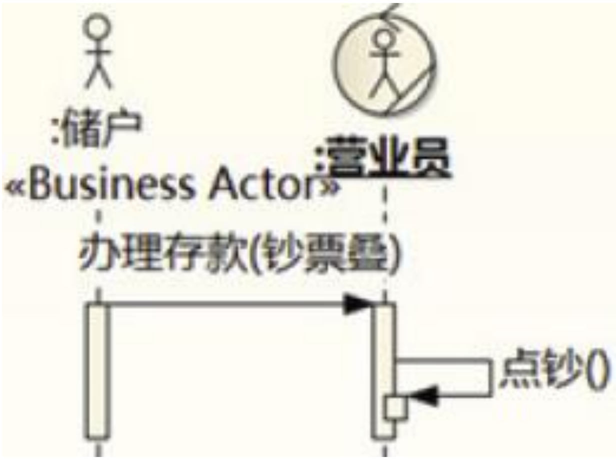
Why Software engineering?

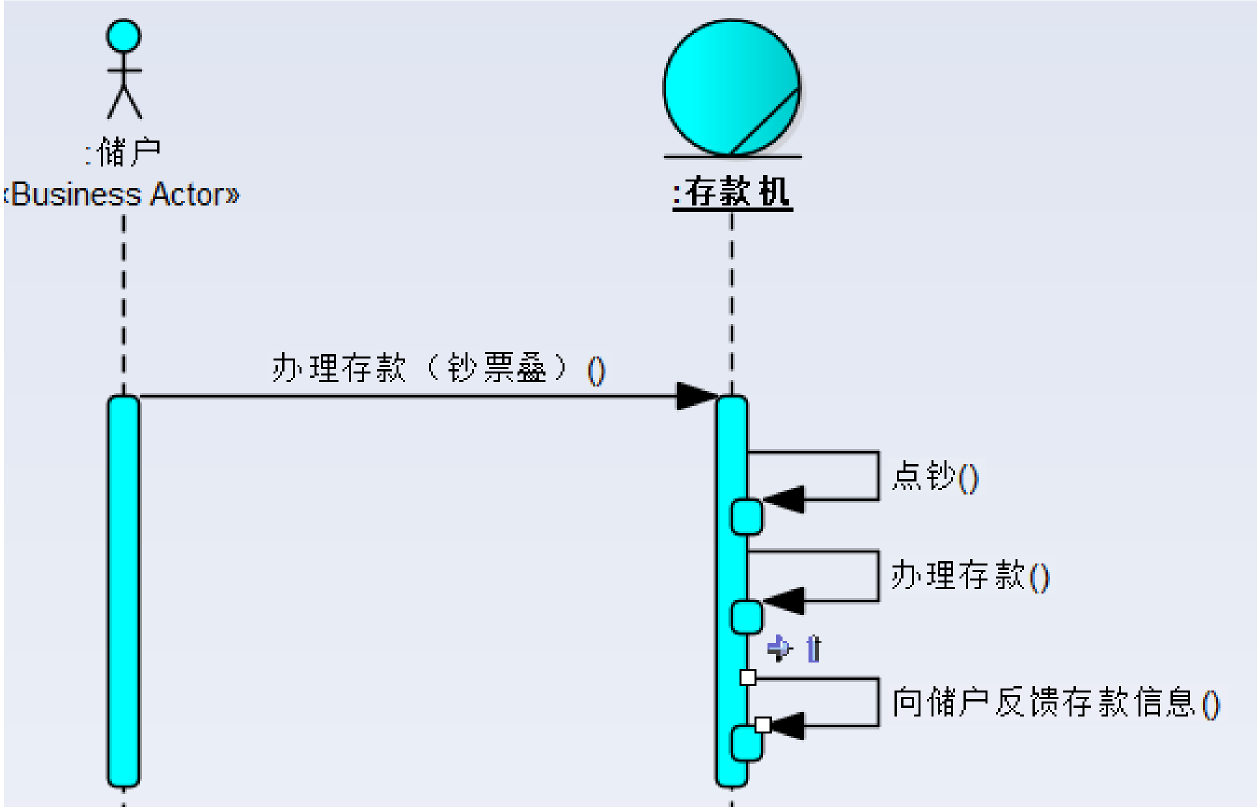
- Three ways that Software improves the current business process:
  - Logical thinking
  - Material flow to information flow
  - Information transfer





Why Software engineering?

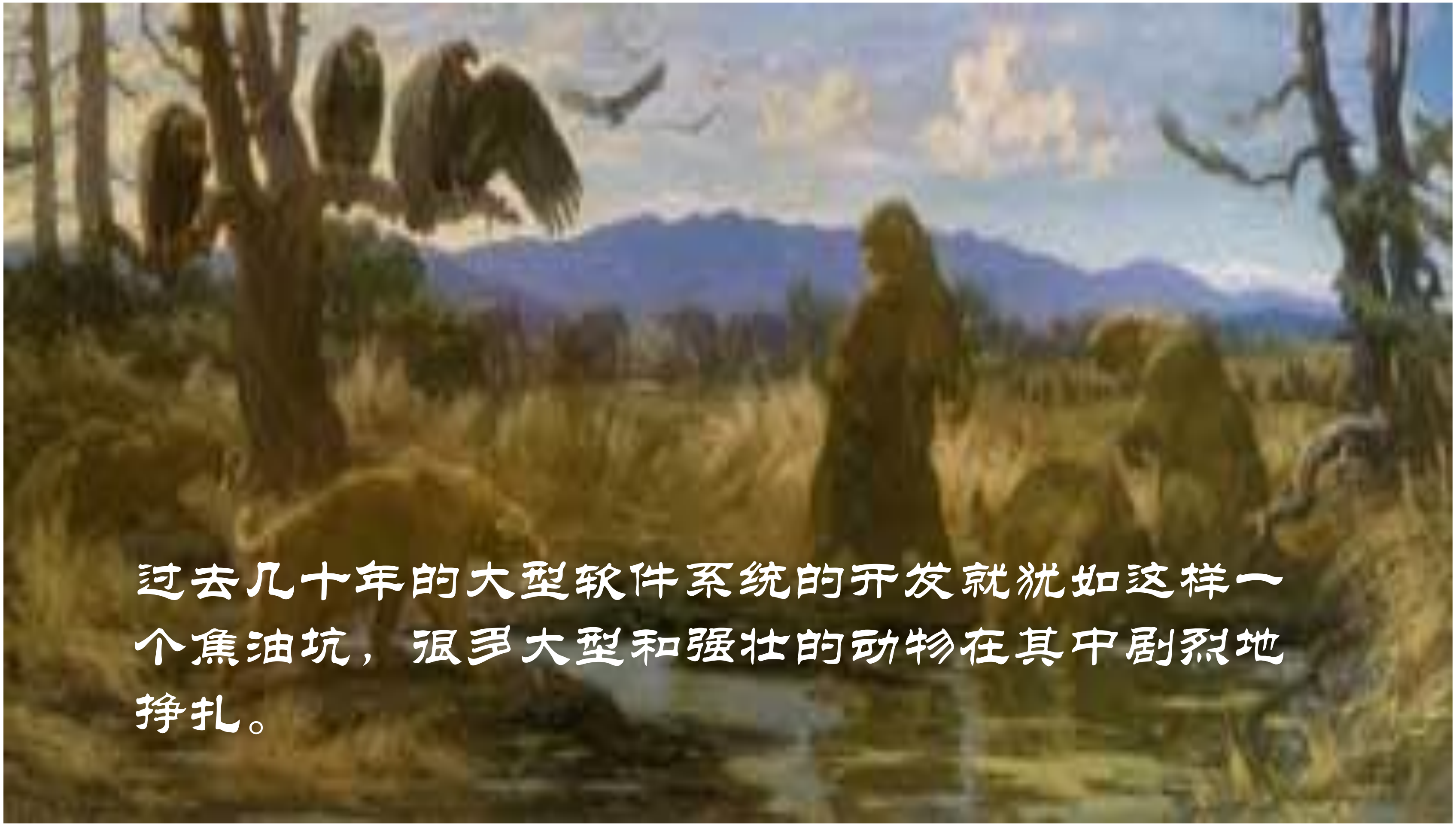




Why Software engineering? (continued)

- Software crisis: a set of problems present in developing software
  - Over budget (bad design)
  - Over schedule: delivered late
  - Deficient in the stated requirements
- Reasons for crisis
  - complexity is inherent (essential) property[1]
  - Intangible (creative effort)
  - Change
  - No effective technology and management approach

Software crisis



过去几十年的大型软件系统的开发就犹如这样一个焦油坑，很多大型和强壮的动物在其中剧烈地挣扎。

拉布雷阿的焦油坑（Mural of La Brea Tar Pits）

A classic example

美国IBM公司在1963年至1966年开发的IBM360机的操作系统。这一项目花了5000人一年的工作量，最多时有1000人投入开发工作，写出了近100万行源程序。......据统计，这个操作系统每次发行的新版本都是从前一版本中找出1000个程序错误而修正的结果。......这个项目的负责人F. D. Brooks事后总结了他在组织开发过程中的沉痛教训时说：“......正像一只逃亡的野兽落到泥潭中做垂死的挣扎，越是挣扎，陷得越深，最后无法逃脱灭顶的灾难。......程序设计工作正像这样一个泥潭，......一批批程序员被迫在泥潭中拼命挣扎，......谁也没有料到问题竟会陷入这样的困境......”。IBM360操作系统的历史教训成为软件开发项目的典型事例为人们所记取。

Why Software engineering? (continued)

- Attack software crisis: software engineering is concerned with theories, methods and tools for professional software development.

No SE, like flash marriage

Why Software engineering? (continued)

- **study and practice following skills(基本功）**
  - Business Modeling
    - 检查疾病
  - Requirements Modeling
    - 诊断报告
  - Analysis Modeling
    - 拟定治疗方案
  - Design and Coding
    - 实施治疗方案，进行治疗

six year old example:看病

Why Software engineering? (continued)

- **study and practice[2] following skills[1]**
  - **business modelling**
    - **SQ, EQ, IQ, AQ**
  - **requirements modelling**
    - **SQ, EQ, IQ, AQ**
  - **analysis modelling**
    - **SQ, EQ, IQ, AQ**
  - **design modelling**
    - **IQ**
  - **……**

Why Software engineering? (continued)

- 练好各项技能，成就梦想！
  - 软件开发之路是充满荆棘与挑战之路,也是充满希望之路，没有捷径可走。梦想像《天龙八部》中虚竹一样被无崖子醍醐灌顶(tí hú guàn dǐng)而轻松获得一甲子功力，是很不现实的。每天仰天大叫"天神啊，请赐给我一本**葵花宝典**吧"，殊不知即使你获得了葵花宝典，除了受自宫其身之苦外，你也不一定成得了"东方不败"，倒是成"西方失败"的几率高一点。
  - "不走弯路，就是捷径"，佛经说的不无道理。[1]
  - **开始慢是为了后来的快**[2]
    - 杰伦用水果弹钢琴
    - 白百何
    - 白(bái)百(bǎi)何(hé)

Why Software engineering? (continued)

But Brooks told us in book, The Mythical Man-Month:

No panacea. No silver bullet

As Brooks points out, “Einstein argued that there must be simplified explanations of nature, because God is not capricious or arbitrary. No such faith comforts the software engineer. Much of the complexity that he must master is arbitrary complexity”

What is Software engineering?

- What is software?
- What is software engineering?
- What is the difference between software engineering and computer science?
- What are three elements of SE
- What is a software process?
- What is a model?

What is Software engineering? (continued)

- What is a software process model?
- 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?

- software = programs + document[1]
  - document = business + requirement + analysis + design + ………
  - document
    - model
      - developer oriented
        - » UML model
    - view
      - stakeholder oriented [2]
- Computer programs and associated document such as business, requirements, analysis, design models and user manuals.

Review

- Key skills for SE

What is software engineering?

Many definitions:

Software Engineering is a collection of techniques, methodologies and tools that help with the production of

- a high quality software system
- within a given budget
- before a given deadline

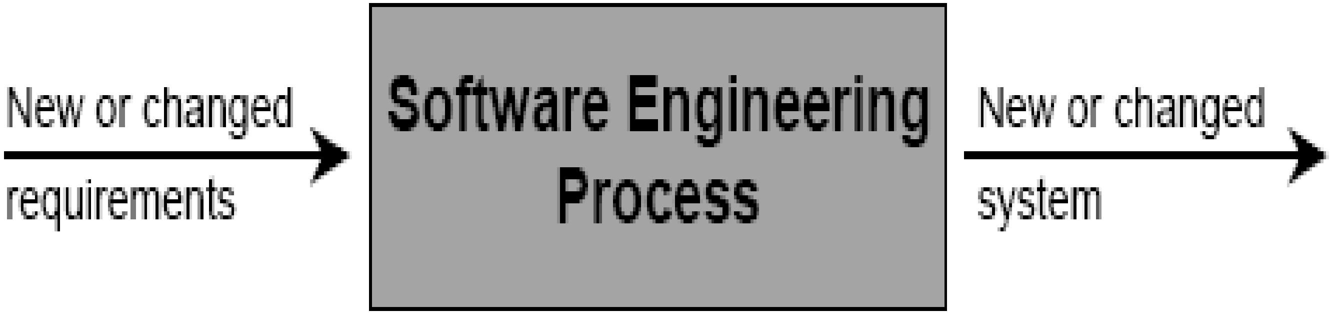
while change occurs.

What are three elements of SE

- Software Process
- Methodology
- Tool: CASE

What is a software process?

A process defines **Who** is doing **What**, **When**, and **How**, in order to reach a certain goal.



New or changed requirements

Software Engineering Process

New or changed system

What is a software process? (continued)

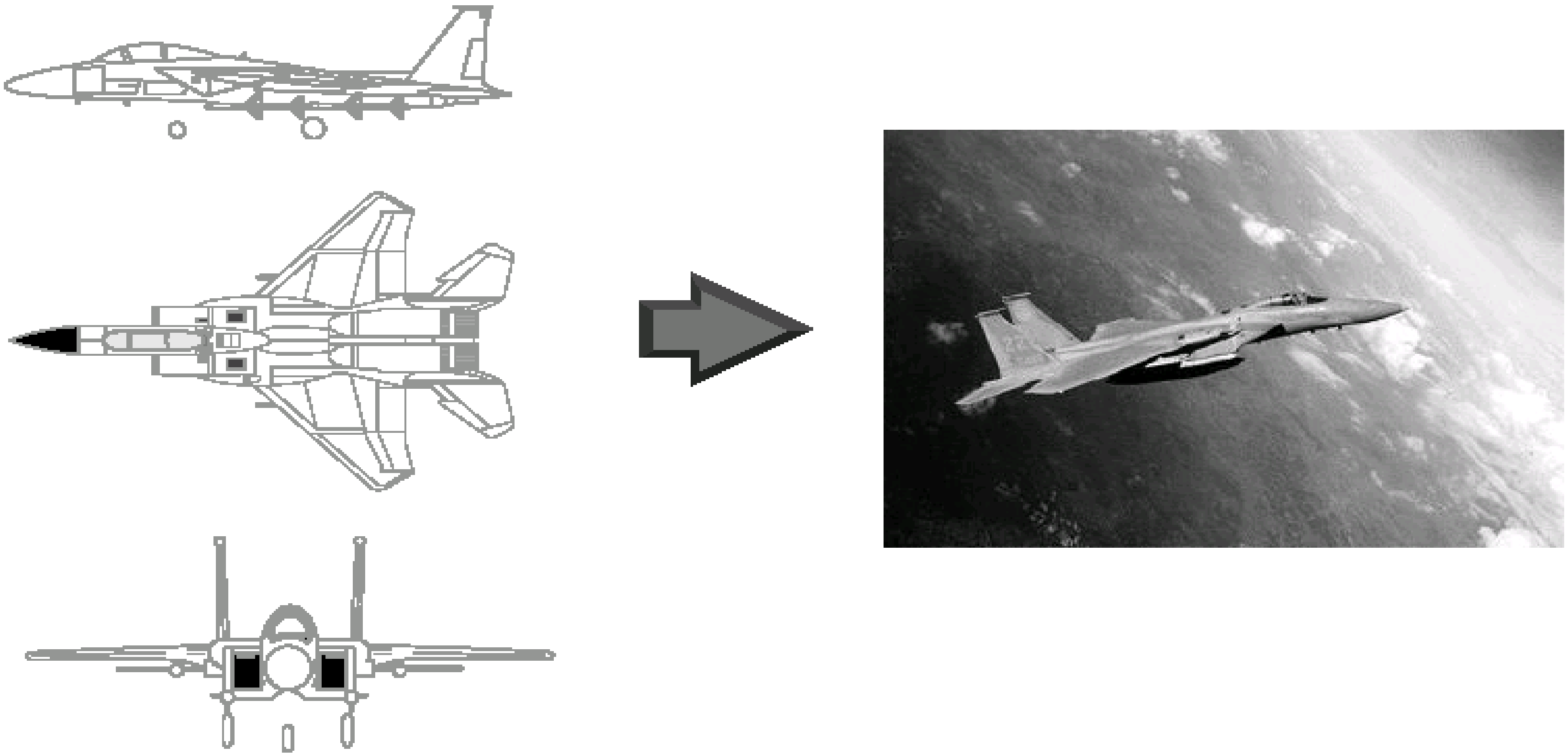
- A set of activities
- Generic activities in all software processes are:
  - business modelling
  - requirement modelling
  - analysis modelling
  - design and implementation
  - test
  - deployment
  - maintenance
  - ………

Software Process

- Software process is complicated and includes many activities.
- We should first focus on some milestone activities---Process model.

Software Process model

- What is a Model?
  - **A model is a simplification or abstraction of reality.**



Software Process model(continued)

- **Why model?**[1]
  - **You build models of complex systems because you cannot comprehend such a system in its entirety**

2+- 7 law[2]

- 建模能力的重要性[3]
  - 错误的模型导致错误的结果

What is a software process model?

- A simplified representation of a software process, presented from a specific perspective.
- Generic process models
  - – Waterfall
  - – Iterative development[1]

What are software engineering methodologies?

SE method is a structured approach to development whose aim is to facilitate the production of high-quality software in a cost-effective way:

- No method: individual hero, craft…
- Programming: individual workshop小作坊
- SE Method: teamwork---finding good players is easy. Getting them to play as a team is another story.

What are software engineering methodologies? (continued)

- PO[1]: procedural oriented : SP, SD, SA
  - DFD: Data Flow Diagram ,E-R diagram
- OO[2] –our focus
  - RUP
  - UML: use case ,activity diagram, sequence diagram…
- There is no ideal method. OO: for interactive not for real-time system
- Combination PO and OO, etc.

What is 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.
- CASE is a Tool:
  - – Instrument or automated systems to accomplish a technique

What are the attributes of good software?

- The software should deliver the required functionality and performance to the user and 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. This means it must be understandable, usable and compatible with other systems.

Factors affecting the quality of a software system

- **Complexity:**
  - **The system is so complex that no single programmer can understand it anymore**
  - **The introduction of one bug fix causes another bug**

Factors affecting the quality of a software system (continued)

- **Change:**
  - **As time goes on, the cost to implement a change will be too high, and the system will then be unable to support its intended task. This is true of all systems .**

Review

- Cope with complexity
  - Abstraction (Modeling)
  - Decomposition
  - Hierarchy
    - Tier, layer

Dealing with Complexity

1. Abstraction (Modeling)
2. Decomposition
3. Hierarchy

   Tier, layer

1. Abstraction: modeling

- Abstraction is one of fundamental ways that we as humans cope with complexity.
- Inherent human limitation to deal with complexity
  - The 7 +- 2 phenomena. Miller’s experiments: an individual can comprehend only about seven, plus or minus two, chunk information at one time.
- Ignore unessential details: => Models

2. Decomposition

- A technique used to master complexity (“divide and conquer”)
- Functional decomposition---SA ,SD
  - The system is decomposed into modules
  - Each module is a major processing step (function) in the application domain
  - Modules can be decomposed into smaller modules

Data-centered, process and data are separated.

2. Decomposition (continued)

- Object-oriented decomposition
  - The system is decomposed into classes ("objects")
  - Each class is a major abstraction in the application domain
  - Classes can be decomposed into smaller classes

Human-centered, data and its process are integrated.

Which decomposition is the right one?

3. Hierarchy

- TCP/IP protocol
- MVC

4. Reusability

Do not reinvent the wheel

Do not repeat yourself

Be lazy

- A good software design solves a specific problem but is general enough to address future problems (for example, changing requirements)
- Experts do not solve every problem from first principles
  - They reuse solutions that have worked for them in the past
- Goal for the software engineer:
  - Design the software to be reusable across application domains and designs
- How?
  - Use design patterns and frameworks whenever possible

5. Design Patterns and Frameworks

- Design Pattern: (蛋炒饭）
  - A small set of classes that provide a template solution to a recurring design problem
  - Reusable design knowledge on a higher level than data structures (link lists, binary trees, etc)
- Framework [1]
  - A moderately large set of classes that collaborate to carry out a set of responsibilities in an application domain.
    - Examples: User Interface Builder, MFC…
  - a kind of reuse

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 if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law.

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 outside 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).

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. Essential product attributes are maintainability, dependability, efficiency and usability.
- The software process consists of activities that are involved in developing software products. Basic activities are software specification, development, validation and evolution.
- Methods are organised ways of producing software. They include suggestions for the process to be followed, the notations to be used, rules governing the system descriptions which are produced and design guidelines.

Key points (continued)

- CASE tools are software systems which are designed to support routine activities in the software process such as editing design diagrams, checking diagram consistency and keeping track of program tests which have been run.
- Software engineers have responsibilities to the engineering profession and society. They should not simply be concerned with technical issues.
- Professional societies publish codes of conduct which set out the standards of behaviour expected of their members.