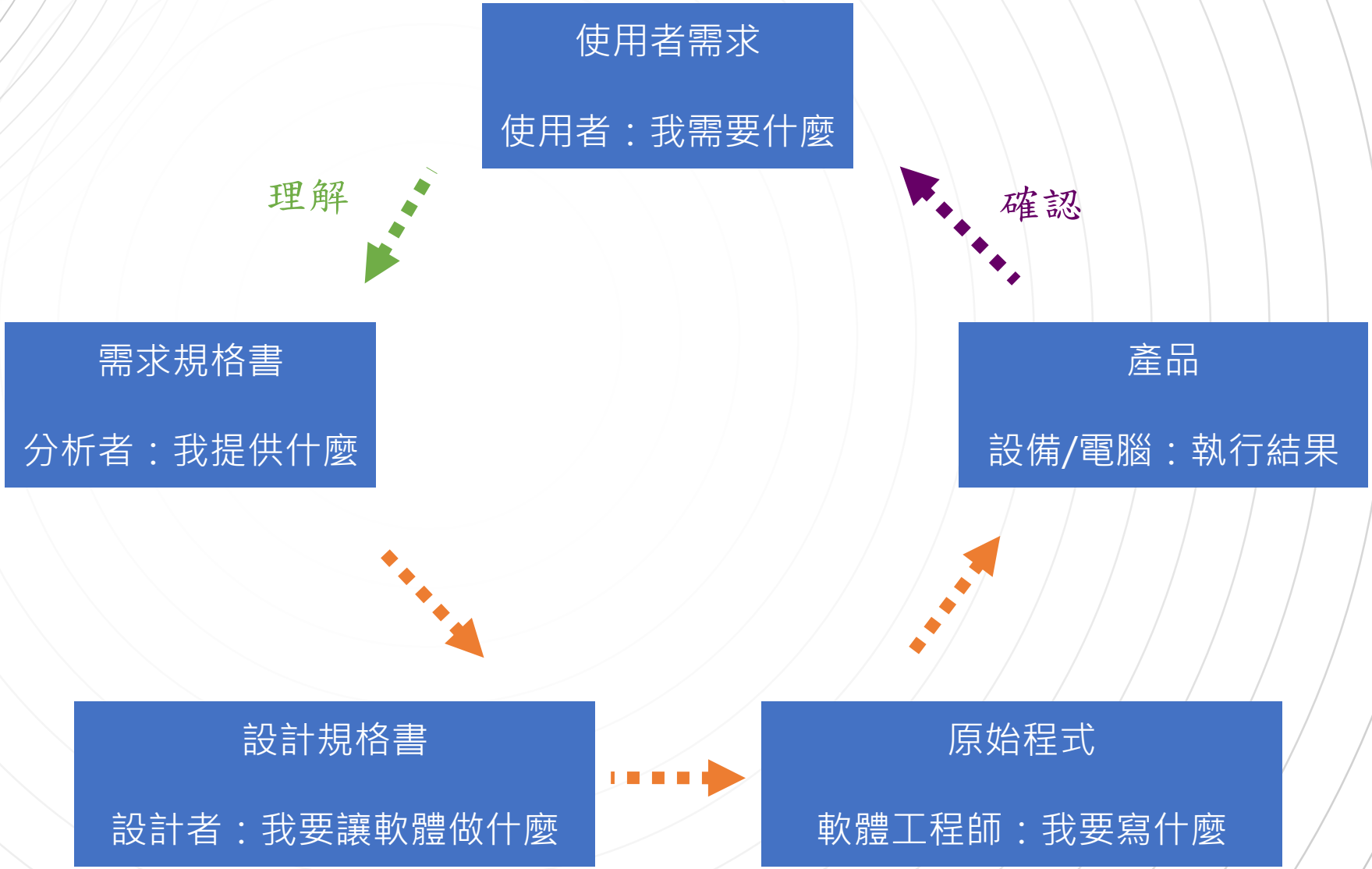
# Chapter 1
# 軟體特性及軟體工程的範圍

# 課程說明

- 課堂上課：線上+教室
- 助教：吳貴仁 (M0929004@cgu.edu.tw)
- Office hour : 星期四下午14:00 – 17:00 (cclin@mail.cgu.edu.tw)
- 專題：
  ① 成員：3或4人一組
  ② 題目：麥當勞點餐
- 評分
  ① 期中考+期末考：**20%**
  ② 報告：**40%**
    - 需求規格書、設計規格書
  ③ 期末專題成果：**40%**
    - 系統展示：web based AP (php, C#)+ database (ex. MySQL)
    - 測試報告+操作手冊

# 軟體開發生命週期


使用者需求
使用者：我需要什麼
理解
確認
需求規格書
分析者：我提供什麼
產品
設備/電腦：執行結果
設計規格書
設計者：我要讓軟體做什麼
原始程式
軟體工程師：我要寫什麼

# Topics covered

- Important Characteristics of  software
- Purpose and  scope of software  Engineering
- Professional and ethical responsibility

# 1.1軟體的重要特性

# What is software?

- Computer programs and associated documentation such as requirements, design models and user manuals.
- Software products may be developed for a particular customer or may be developed for a general market.
- Software products may be
  - Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - Bespoke (custom) - developed for a single customer according to their specification.
- New software can be created by developing new programs, configuring generic software systems or reusing existing software.

# 1 軟體是工具或娛樂性玩具

- **從軟體的用途/目的作**軟體的分類
  - WORD. Window OS, 選課系統,手機內軟體等等,使用動機:提升工作效率/方便性　(工具)
  - 但COMPUTER GAME?　(娛樂)
- TOOL or ENTERTAINMENT
- 工具:→產品成功的條件!

① 為何要用(目的)?
② 什麼人用?
③ 何時用?
④ 如何用?
⑤ 得到什麼效益?

# 2 軟體是一個系統

- 系統:由一群相互連結的組件( components)組成,共同運作以完成特定任務
  - 任務的複雜性影響系統的複雜度及組件的數量
  - 每一組件均有被使用的時機
  - 一任務的每一工作由一或多個組件的正常運作來達成,多個組件間有連動關係
- 組件:程式模組、資料模組(檔案、資料庫、資料結構)
- 大組件由一群小組件組成

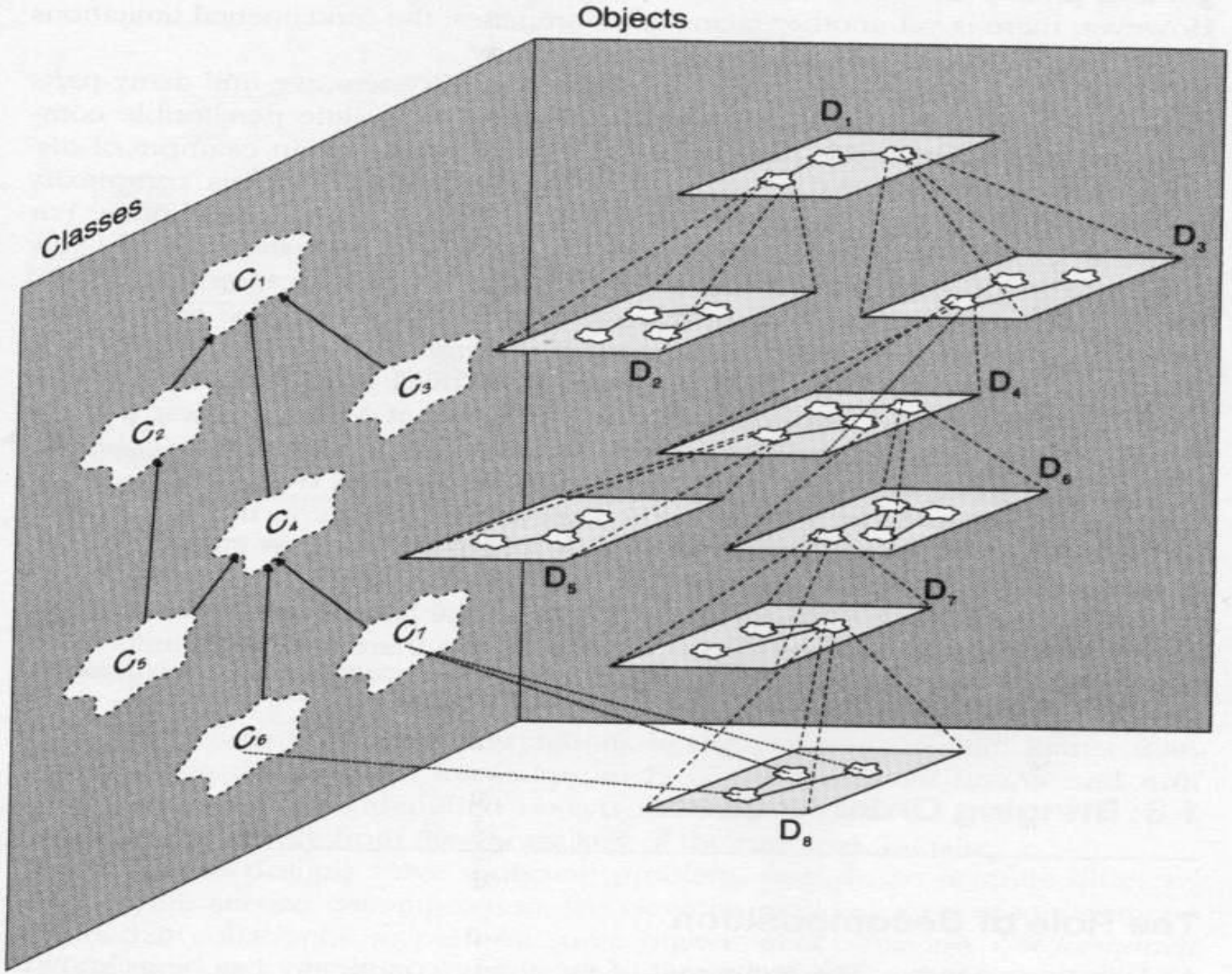



**Figure 1-1**
The Canonical Form of a Complex System

# 3 軟體不會磨損,只是不斷演進

- 軟體會讓你繼續用下去的理由: meet customer needs, high quality(助益大、好操作、可靠、等等)
- 軟體該停用的理由:  沒有價值!

➢Maintenance of bridge vs operating system

  - Bridge: painting, repair minor cracks, resurfacing road <--由磨損造成
  - OS: from batch to time sharing → change of architecture, lots of code should be  rewritten. <--功能改變(更複雜!)

# 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.

# 3 軟體不會磨損,只是不斷演進

- 軟體為何要修改?
  - 操作不便、修改報表格式、錯誤 →修改程式
  - 平台改變 →修改平台介面程式或調整架構
  - 增加新功能→調整架構及增修改程式
- Software maintenance:修改軟體以維持軟体正常運作且滿足客戶使用
- More complicated for software than other engineering product：軟體比一般工程產品更複雜

# 4 軟體內容包括程式碼與設計文件

- 修改軟體時,只有程式碼有何不便?
  - 不知當時如何設計!
  - 不易找出該修改處!
  - 不易發現修改所造成的影響
- 軟體內容:程式碼與設計文件
  - 使人了解設計的動機、功能、軟體架構、各模組設計方法
  - 程式碼及所需 Data
  - 測試環境、程式、資料、及步驟
  - 建置、使用、維護文件

# 5 軟體是創意作品

- 軟体是為明天的使用而開發
  - 使用者?用來作什麼?得到什麼效益?如何使用?
  - product requirements( i.e. what the users need) of a software at a time.
- 你會要求A公司設計B公司已有的產品嗎?
  - 針對A公司的特性與需求，重新開發軟體提升競爭力
- 軟体高設計成本,低生產成本

# 5 軟體是創意作品

- Requirements: what services the software should provide, constraints of services and development
- Development of a software product is usually started from a vague concept. (模糊概念或想法)
- Usually, user(s) is difficult to list all what he wants the system to do unless he know computer very well; developer(s) does not have enough domain knowledge to help user(s) to find out what services the software should provide to solve user’s problems.

# 6 軟體是高複雜性系統

- 影響軟體複雜度的因素:
  - 供多不同類型的使用者使用
  - 多使用者同時使用
  - 多使用者同時使用,且相互影響
  - 多使用者分散各地同時使用
  - 在多平台可使用
  - 在限定時間內完成工作
  - 不容許錯誤
  - 不容許不當使用
  - ...

# 1.2 軟體工程的目的與範圍

# Importance of software

- More and more systems are software controlled
- The economies of all developed nations are dependent on software.
- Expenditure on software represents a significant fraction of GNP in all developed countries.
- Software engineering is concerned with theories, methods and tools for professional software development.

# What is software engineering?

- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Software engineers should adopt a systematic and organised approach to their work and use appropriate tools and techniques depending on the problem to be solved, the development constraints and the resources available.

# Why Software Engineering is needed?

- Software should be developed within limit of **budget**, and on **time**.
- Characteristics of software development
  - Knowledge-intensive job
  - Team work
  - Coordination of people task and progress is very important
- Software engineering is concerned with **cost-effective** software development.

# Learning experience

- Different approaches affect the capability to build complex house.
  - Different modeling technique
  - Different verification technique
  - Different method→ affect process and management
  - Different tools

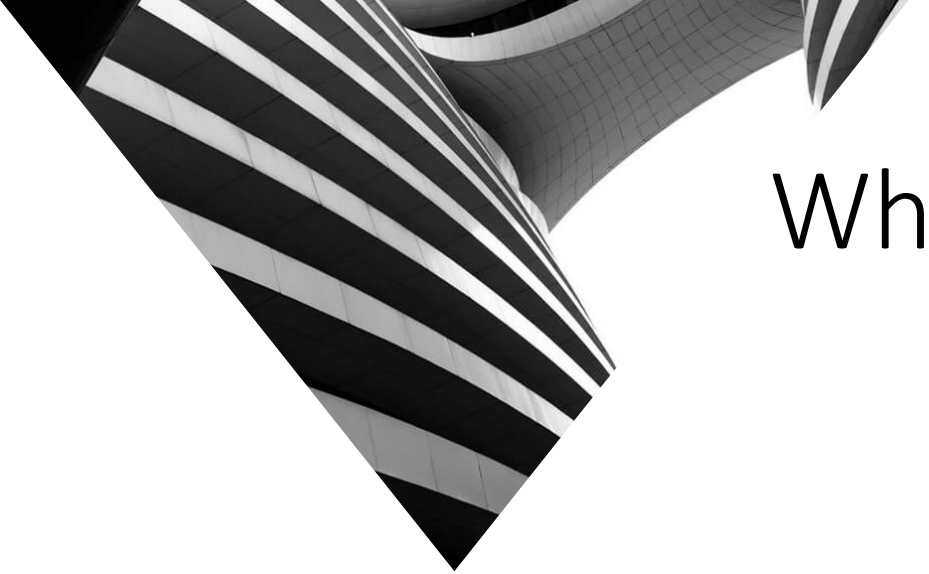
  →Key issues of software engineering research
- 了解問題的起因、性質、限制條件、及預期結果，進而思考可能的解決策略(類比法是一種參考作法)，進行不同解決策略的評估，再探討解決方法,最後分析方法的實用性

# What is a software process?

- A set of activities whose goal is the development or evolution of software.
- Generic activities in all software processes are:
  - 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.

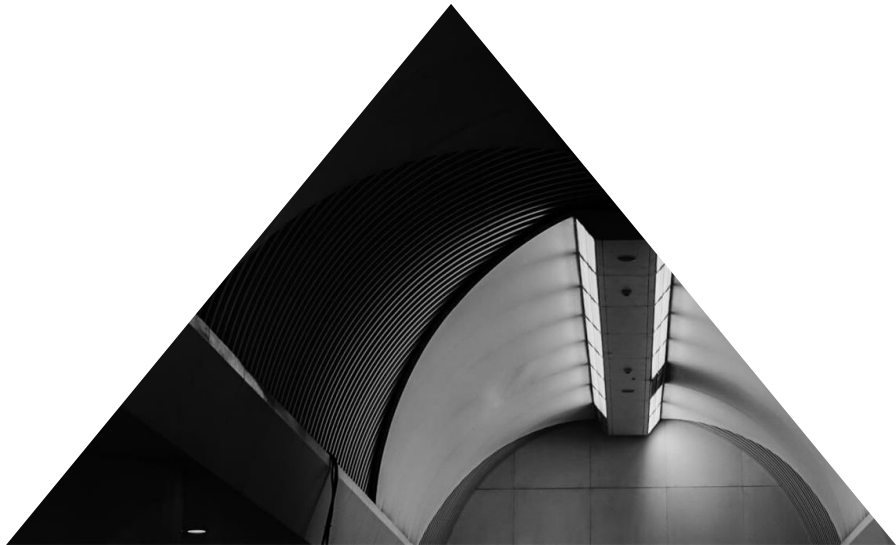
# What are software engineering methods?

- Structured approaches to software development which include 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 we should learn?

- Software engineering theory & methods
- Software development process
- Software project management



# 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.

# 1.3 Professional and ethical responsibility

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

- Competence (權限)
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is out with 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).

# THANKS.