Software Engineering

U-01 軟體工程概述

Neter Chao Ph.D.

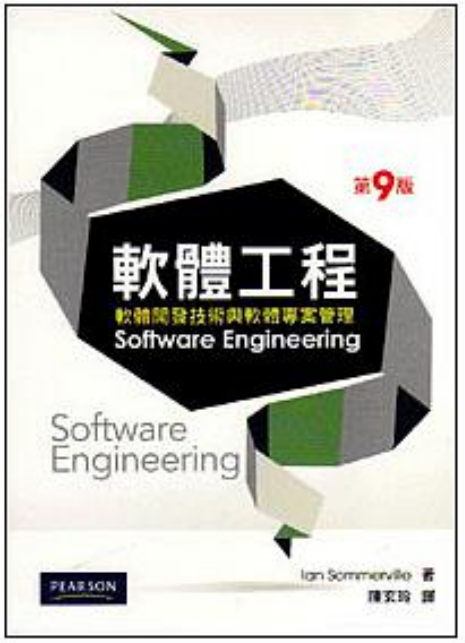
Agenda

- 軟體工程與系統分析設計
- 資訊系統的三種觀點

軟體工程與系統分析設計

參考用書

- **軟體工程：軟體開發技術與軟體專案管理**
- Sommerville：Software Engineering 9/E
- 作者： 陳玄玲/譯
- 出版社：高立圖書
- 出版日期：2011/02/01
- 綱要：
- 第1單元　軟體工程概觀
  第2單元　可信賴度和保全性
  第3單元　軟體工程進階議題
  第4單元　軟體管理



詳細單元之1

- **第1單元　軟體工程概觀**

- 第 1 章　導論
- 第 2 章　軟體程序
- 第 3 章　敏捷式軟體開發
- 第 4 章　需求工程
- 第 5 章　系統塑模
- 第 6 章　架構設計
- 第 7 章　設計與實作
- 第 8 章　軟體測試
- 第 9 章　軟體演進

詳細單元之2

- **第2單元　可信賴度和保全性**

- 第10章　社會化技術系統
- 第11章　可信賴度與保全性
- 第12章　可信賴度與保全性規格
- 第13章　可信賴度工程
- 第14章　保全工程
- 第15章　可信賴度與保全性的保證

詳細單元之3

- **第3單元　軟體工程進階議題**

- 第16章　軟體再利用
- 第17章　元件式軟體工程
- 第18章　分散式軟體工程
- 第19章　服務導向架構
- 第20章　嵌入式軟體
- 第21章　觀念導向軟體工程

詳細單元之4

- **第4單元　軟體管理**

- 第22章　專案管理
- 第23章　專案規劃
- 第24章　品質管理
- 第25章　組態管理
- 第26章　程序改善

軟體工程

- 所有已開發國家的經濟都跟軟體有關
- 愈來愈多系統是受軟體控制
- 軟體工程是一門討論開發專業軟體的理論、方法與工具的學科
- 已開發國家對軟體工程的花費佔GNP非常大的比率
- 維基百科的說明：

「**軟體工程**是研究和應用如何以系統性的、規範化的、可定量的程序化方法去開發和維護軟體，以及如何把經過時間考驗而證明正確的管理技術和當前能夠得到的最好的技術方法結合起來的學科。它涉及到*程式語言*、*資料庫*、*軟體開發工具*、*系統平台*、*標準*、*設計模式*等方面。」(維基百科)

軟體工程

北大西洋公約組織（NATO）在1968年舉辦了首次軟體工程學術會議，並於會中提出「軟體工程」來界定軟體開發所需相關知識，並建議「軟體開發應該是類似工程的活動」。軟體工程自1968年正式提出至今，這段時間累積了大量的研究成果，廣泛地進行大量的技術實踐，藉由學術界和產業界的共同努力，軟體工程正逐漸發展成為一門專業學科。

[Ref]

NATO SCIENCE COMMITTEE的SOFTWARE ENGINEERING Report

➔**http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF**

軟體工程的核心知識(SWEBOK)

- ACM與IEEE Computer Society聯合修定的SWEBOK (Software Engineering Body of Knowledge)提到，軟體工程領域中的核心知識包括：

1. 軟體需求（Software requirements）
2. 軟體設計（Software design）
3. 軟體建構（Software construction）
4. 軟體測試（Software test）
5. 軟體維護與更新（Software maintenance）
6. 軟體構型管理（Software Configuration Management, SCM）
7. 軟體工程管理（Software Engineering Management）
8. 軟體開發過程（Software Development Process）
9. 軟體工程工具與方法（Software Engineering Tools and methods）
10. 軟體品質（Software Quality）

系統分析

- 維基百科的說明：

1. **系統分析**，旨在研究特定系統結構中各部分（各子系統）的相互作用，系統的對外介面與界面，以及該系統整體的行為、功能和局限，從而為系統未來的變遷與有關決策提供參考和依據。系統分析的經常目標之一，在於改善決策過程及系統性能，以期達到系統的整體最優。

2. 系統分析被看作是系統工程的一個重要程序和核心組成部分，以及系統理論的一項應用。 在系統開發生命周期中，系統分析階段先於系統設計，是系統開發前期不可或缺的工作。 系統分析大量借用數學模型、數學分析、計算機模擬等定量分析方法，試圖在具有不確定約束或邊界條件的情況下，對系統要素進行綜合分析、描述，得出較為準確或合理的結論。」(維基百科)

軟體成本

- 軟體的成本通常高於系統成本；而PC上的軟體成本通常高於硬體的成本
- 軟體成本中維護成本高於開發成本。對使用壽命很長的系統而言，維護成本將會是開發成本的數倍之多
- 軟體工程是討論如何達成最合乎經濟效益的軟體開發或演化

Q&A

- 何謂「軟體」(Software)？
- 何謂「軟體工程」(Software Engineering)？
- 軟體工程與電腦科學(Computer Science)有何不同？
- 軟體工程與系統工程(System Engineering)有何不同？
- 何謂「軟體程序」(Software Process)？
- 何謂「軟體程序模型」(Software Process Model)？

軟體工程常見問答集

- 何謂軟體工程的成本？
- 何謂軟體工程方法？
- 何謂CASE (Computer-Aided Software Engineering)？
- 好的軟體有哪些特性？
- 軟體工程面臨的主要挑戰？

何謂軟體

- 電腦程式與相關文件
- 軟體產品專為特定客戶開發，或是針對一般市場進行開發
- 軟體產品分為兩類
  - 通用產品 (Generic) – 開發的產品可已販賣給各種不同客戶使用
  - 客製化產品 (custom) – 專為某一位客戶所定的規格而開發的產品

軟體工程與電腦科學之不同

- 電腦科學是與電腦和軟體系統基本理論和方法有關的學科；軟體工程則是與生產和交付有用軟體所面臨的實際問題有關
- 電腦科學的理論目前為止不足以扮演軟體工程完整的支撐

軟體工程與系統工程之不同

- 系統工程是跟電腦化系統開發各種層面有關的工程學科，包括硬體、軟體和流程；軟體工程則只是這個流程中的一部分。
- 系統工程師的工作包括系統規格制定、架構設計、整合與佈署等

軟體程序

- 進行軟體開發或演化的一組活動
- 所有軟體程序均有的通用活動：
  - 規格制定 – 定義系統應該做什麼事以及開發的限制
  - 開發 – 軟體系統的製作
  - 確認 – 檢查軟體是否為客戶所要的
  - 演化 – 更改軟體以回應變更的要求

何謂軟體程序模型

- 以某個特定觀點呈現的軟體程序簡化表示
- 程序觀點的範例有
  - 工作流程觀點 – 依序的活動
  - 資料流觀點 – 資訊流
  - 角色/動作觀點 – 誰應該做什麼事
- 通用的程序模型
  - 瀑布式(Waterfall)
  - 演化式開發(Evolutionary development)
  - 正規轉換(Formal transformation)
  - 以再利用元件整合(Integration from reusable components)

軟體工程的成本

- 大約 60% 的成本為開發成本，40%為測試成本。對客製化的軟體而言，演化成本通常會超過開發成本
- 成本會根據正在開發的系統類型以及各種系統屬性的需求而定，例如執行效能和系統可靠度
- 成本的分佈依據使用的開發模型而定

何謂軟體工程方法

- 軟體開發的結構化方法包括有系統模型、代表符號、規則、設計建議以及程序指引等
- 模型描述
  - 應該產生圖形化的模型描述
- 規則
  - 套用至系統模型的限制
- 建議
  - 良好的設計實務的建議
- 程序指引
  - 依循哪些活動

CASE

(Computer-Aided Software Engineering)

- 目的為提供軟體程序活動自動化支援的軟體系統。CASE系統通常用來做為方法的支援
- Upper-CASE
  - 支援如需求與設計等早期程序活動的工具
- Lower-CASE
  - 支援如程式設計、除錯與測試等後期活動的工具

好的軟體有哪些特性

- 軟體應該完成使用者所需的功能和執行效能，也應該能夠可維護、可信賴以及可使用。
- 可維護性(Maintainability)
  - 軟體必須能夠進行演化以符合變更的需求
- 可信賴度(Dependability)
  - 軟體必須能夠信任
- 效率(Efficiency)
  - 軟體不應該浪費系統資源的使用
- 可使用性(Usability)
  - 軟體必須可以讓使用者針對其設計來使用

軟體工程的主要挑戰

- 處理既有舊系統，處理逐漸增加的變化性以及處理減少交付時間的要求
- 既有舊系統(Legacy systems)
  - 舊的但有其存在價值的系統，因此必須加以維護與更新
- 異質性(Heterogeneity)
  - 系統為分散的且包含各種軟硬體的混合
- 完成與交付(Delivery)
  - 軟體快速完成與交付的壓力逐漸增加

道德規範 – 原則

- 1. PUBLIC
  - 軟體工程師應該固守公眾的利益。
- 2. CLIENT AND EMPLOYER
  - 軟體工程師應該以讓他的顧客和雇主得到最佳的利益，並且固守公眾的利益。
- 3. PRODUCT
  - 軟體工程師應該確保他的產品和相關的修改能夠儘可能符合最高的專業標準。

道德規範 – 原則

- 4.JUDGMENT
  - 軟體工程師在專業判斷上應該維護其正直與獨立。
- 5.　MANAGEMENT
  - 軟體工程師的經理和上司們應該在軟體開發與維護上支持與提倡合乎道德的管理方法。
- 6. PROFESSION
  - 軟體工程師應該提高專業的正直與聲譽，並與公眾的利益一致。

Code of ethics - principles

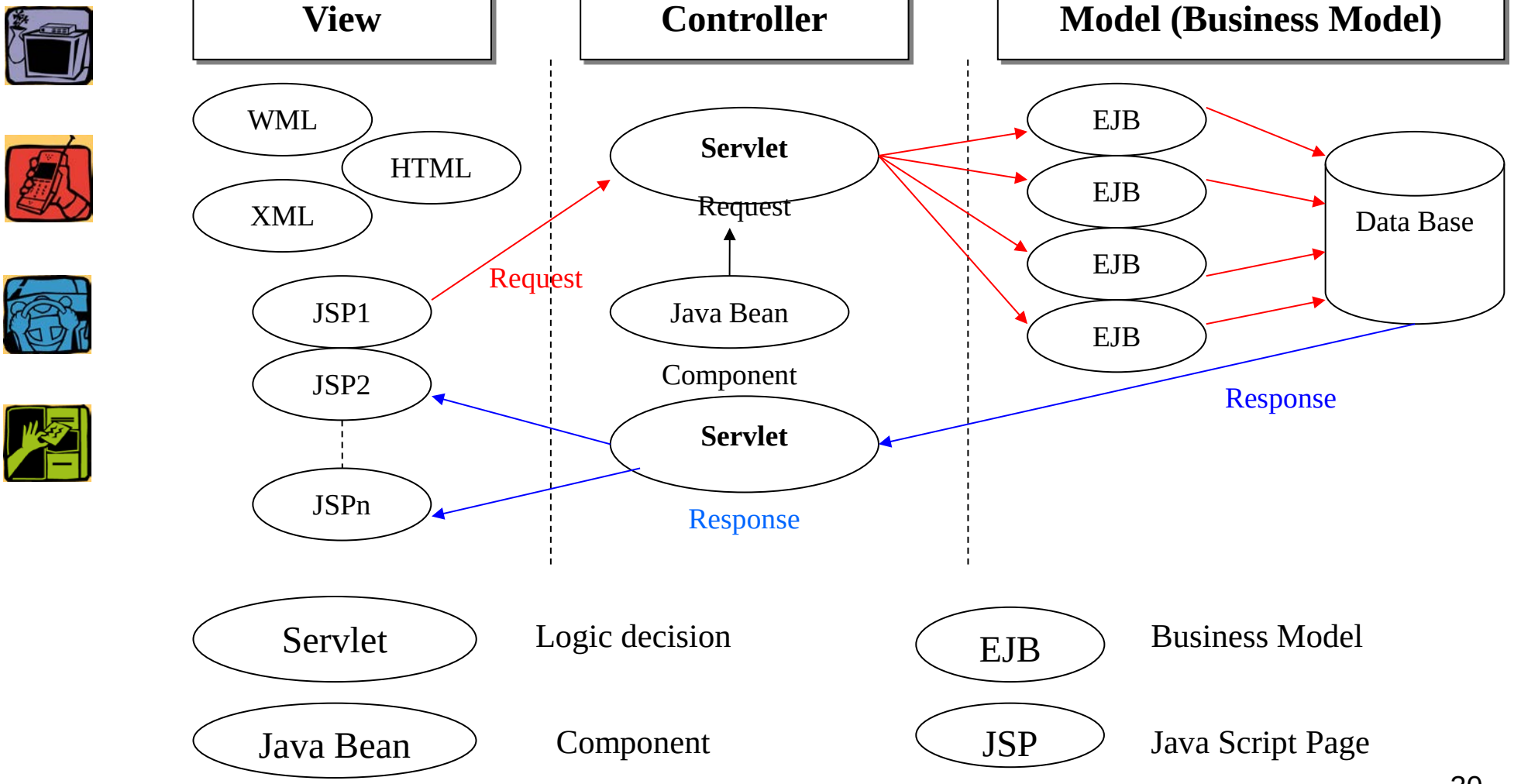
- 7.　COLLEAGUES
  - 軟體工程師應該公平的支援他們的同事。
- 8. SELF
  - 軟體工程師應該在他的專業實務上終身學習，並且應該在專業實務上提倡合乎道德的方法。

資訊系統的三種觀點




- 經營管理觀點 (Strategy View)
- 資訊技術觀點 (IT View) (互聯網絡觀點 ,IOT View)
- 經營流程觀點 (Business Process View)

Model View Controller Architechure (M.V.C)



30