第十一章
軟體專案與型態管理

執行單位：國立臺灣科技大學
軟體工程學程中心

大綱

- 軟體專案型態管理定義與工作內容
- 軟體專案型態管理標準與內容
- 軟體專案基準線與型態管理的關係
- 軟體專案變更管制的作業程序
- 軟體專案型態稽核目的與主要工作
- 軟體專案型態管理工具

學習目標

- 組織內軟體專案型態管理的內容
- 軟體專案型態管理限制與指引
- 軟體專案型態管理的規劃
- 監督與稽核軟體專案型態管理

軟體專案型態管理定義

- 一個系統型態是指可能是硬體、韌體、軟體或技術文件的結合以完成一個產品[Buckley]。
- 特定版本(硬體、韌體、軟體或技術文件)管控依據程序及目標。
- 所有版本現狀及修改皆可追蹤並可回溯性，藉以提升可維護性。
- 具體地說：型態管理是標識和型態識別項，以維護其完整性、可追溯性以及正確性的學科。

工作內容

- 工作內容
  - 型態識別(Configuration Identification)
  - 版本控制 (Version Control)
  - 變更控制(Configuration Control)
  - 型態狀況紀錄(Configuration Status Accounting)
  - 型態稽核(Configuration Audit)

型態識別

- 型態識別(Configuration Identification)選擇適當的記述方式，包括運用數字編號及其它識別名稱，以區別各類軟體產品（文件、程式、資料）。
- 使用已獲准之記述方式，用各種不同的形式（包含規格、藍圖、清單、界面控制文件…等），以區別、定義出各種項目的功能及實體特徵(functional and physical characteristics)。

識別方法

- 軟體產品的識別方法，主要是選擇適當的記述方式，包括運用數字編號及其它識別名稱，以區別各類軟體產品（文件、資料、程式）。「軟體產品識別」必須確保各項軟體產品的「命名」(Naming)均有意義，而且具有一致性。

文件的識別

- 專案名稱(或專案代碼)
- 系統/子系統名稱
- 承包商名稱
- 客戶名稱
- 合約編號

版本控制

- 版本控制(Version Control)
- 構型管理的核心功能。
- 構型識別庫中的元素都應自動予以版本的標識。
- 版本命名的唯一性。

變更控制

- 變更控制(Configuration Control)主要依據基線配置項。

構型狀況紀錄

- 構型狀況報告(Configuration Status Accounting)應根據報告應著重反映當前基線配置項的狀態，以作為對開發進度報告的參照。

構型狀況報告應該包括下列主要內容：

- 配置庫結構和相關說明
- 開發起始基線的構成
- 當前基線位置及狀態
- 各基線配置項集成分支的情況
- 各私有開發分支類型的分佈情況
- 關鍵元素的版本演進記錄
- 其他應予報告的事項

構型稽核

- 構型稽核(Configuration Audit)的主要作用是作為構型管制的補充手段，來確保某一變更需求已被確實實現。在某些情況下，它被作為正式的技術審查的一部分，但當軟體構型管理成為一個正式的活動時，該活動由SQA人員單獨執行。

軟體專案構型管理標準與內容（1/2）

- 軟體工程技術正吸引著越來越多關注的目光。CMMI為代表的先進的軟體工程理念在國內也正日益受到業界廣泛的重視。軟體構型管理（Software Configuration Management，SCM）作為CMM 2級的一個關鍵域（Key Practice Area，KPA），在整個軟體的開發活動中佔有很重要的位置。

軟體專案構型管理標準與內容(2/2)

- 軟體構型管理活動設計一個能夠融合現有的軟體發展流程的管理過程，甚至直接以這個軟體構型管理過程為框架，來再造組織的軟體發展流程
- 在CMM 二級中，其中最後一個KPA 軟體構型管理的目的就是在專案的整個軟體生命週期內建立並維護軟體專案產品的完整性，更好地管理開發。實際上，構型管理是大多數軟體工程和管理流程的一個重要構成部分。

軟體專案基準線與構型管理的關係

- 軟體專案基線與構型管理的關係
- 基線屬性
- 建立基線的好處

軟體專案基準線與構型管理的關係

- 在軟體發展過程中，由於各種原因，可能需要變動需求、預算、進度和設計方案等，儘管這些變動請求中絕大部分是合理的，但在不同的時機作不同的變動，難易程度和造成影響差別甚大，為了有效地控制變動，軟體構型管理引入軟體專案基線（baseline）的概念。

基線屬性

- 通過正式的評審過程建立
- 基線存在於基線庫中，對基線的變更接受更高許可權的控制
- 基線是進一步開發和修改的基準和出發點
- 進入基線前，不對變化進行管理或者較少管理
- 進入基線後，對變化進行有效管理，而且這個基線作為後繼續工作的基礎
- 不會變化的東西不要納入基線
- 變化對其他沒有影響的可以不納入基線

建立基線的好處（2/2）

- 重現性
  - 及時返回並重新生成軟體系統給定發佈版的能力，或者是在專案中的早期所具備開發環境的能力。當認為更新不穩定或不可信時，基線為團隊提供一種取消變更的方法
- 可追蹤性
  - 建立專案工件之間的前後繼承關係。目的是確保設計滿足要求、代碼實施設計以及用正確代碼編譯可執行檔。

建立基線的好處(2/2)

- 版本隔離
  - 基線為提供了一個定點和快照，新專案可以從基線提供的定點之中建立。作為一個單獨分支，新專案將與隨後對原始專案（在主要分支上）所進行的變更進行隔離。

軟體專案變更管制的作業程序

- 變更控制主要依據基線配置項，一般流程是：
  - （獲得）提出變更請求
  - 由CCB審核並決定是否批准
  - （被接受）修改請求分配人員為，提取SCI，進行修改
  - 復審變化
  - 提交修改後的SCI
  - 建立測試基線並測試
  - 重建軟體的適當版本
  - 復審（審查）所有SCI的變化
  - 發佈新版本

軟體專案構型稽核目的與主要工作

- 軟體專案構型稽核主要工作
  - 功能型態稽核(Functional Configuration Audit；FCA)
  - 實體型態稽核(Physical Configuration Audit；PCA)

功能型態稽核主要工作

- 審查「軟體測試報告」
- 審查「正式鑑定測試」(FQT)
- 確認所有的工程變更要求，均已完整納入管制，並已獲得解決。
- 審查相關的「運作及支援文件」(Operation and Support Document)
- 審查每一「檢驗與測試項目」及「檢驗與測試結果」

實體構型稽核主要工作

- 審查「軟體產品規格」(SPS)與「版本說明文件」(VDD)的一致性與完整性。
- 審查已經發展完成(as-built)的「軟體」與「軟體產品規格」(SPS)、「運作及支援文件」相互之間的一致性與完整性。
- 確認目前的「軟體產品規格」(SPS)是否與原來識別的內容一致。
- 確認正確的版本及修改資訊是否已納入基準文件，並與基準的構型狀況報告相符。

軟體專案構型管理工具與導入程序

- 軟體專案構型管理工具
  - Rational ClearCase
  - Hansky Firefly
  - Concurrent Versions System
  - Microsoft Visual Source Safe
  - Merant PVCS

軟體專案構型管理工具（1/5）

- **Rational ClearCase**
- Rational 公司是全球最大的軟體CASE 工具提供商，現已被IBM收購。它開發的構型管理工具ClearCase 也是深受用戶的喜愛，是現在應用面最廣的企業級、跨平臺的構型管理工具之一。

軟體專案構型管理工具（2/5）

- **Hansky Firefly**
- 為Ｈａｎｓｋｙ公司軟體發展管理套件中重要一員的Firefly，可以輕鬆管理、維護整個企業的軟體資產，包括程式碼和相關文檔。Firefly是一個功能完善、運行速度極快的軟體構型管理系統，可以支援不同的作業系統和多種層級開發環境，因此它能在整個企業中的不同團隊，不同專案中得以應用。

軟體專案構型管理工具（3/5）

- Concurrent Versions System
- Concurrent Versions System；CVS 的縮寫，它是開放源代碼軟體，由於其簡單易用、功能強大，跨平臺，而且免費，它在全球中小型軟體企業中得到了廣泛使用。其最大的遺憾就是缺少相應的技術支持，許多問題的解決需要自己尋找資料。

軟體專案構型管理工具（4/5）

- Microsoft Visual Source Safe
- Visual Source Safe；VSS，是微軟公司為Visual Studio配套開發的一個小型的構型管理工具，準確來說，它僅能夠稱得上是一個小型的版本控制軟體。VSS的優點在於其與Visual Studio產品結合，使用簡單。提供了歷史版本記錄、修改控制、日誌等基本功能。

軟體專案構型管理工具（5/5）

- PVCS
- MERANT 公司的PVCS 能夠提供對軟體配置管理的基本支援，通過使用其圖形介面或類似SCCS 的命令，能夠基本滿足小型專案開發的構型管理需求。PVCS 雖然功能上也基本能夠滿足需求，但是其性能表現一直較差，逐漸地被市場所冷落。

結論

- 構型管理本身無論從理論和實踐都在不斷豐富和發展。例如，型態管理應用於“知識庫”的管理就產生了“專案管理”這一新的領域。構型管理提供的狀態報告和資料統計也為軟體度量提供了決策依據。構型管理為專案管理提供了各種監控專案進展的視角，為專案經理確切掌握專案進程提供了保證。構型管理也為開發人員提供了一同協同合作的平臺，在此平臺上，大家能夠更有效率的交流和合作。可以說，構型管理是軟體發展的基石！

自我評量

- 軟體專案構型管理定義為何？
- 建立軟體專案基準線目的？
- 專案構態稽核目的與主要工作？
- 軟體專案構態管理工具有哪些？
- 軟體專案構態與導入程序為何？

參考資料

- Susan Dart, “Concepts in Configuration Management Systems” Software Engineering Institute Carnegie-Mellon University Pittsburgh, PA. 15123-3890
- UML軟體工程組織 http://www.uml.org.cn/pzgl/pzgl.asp
- 林信惠、黃明祥、王文良,”軟體專案管理”，智勝文化,2002.02.01

附錄（1/2）

- 版本控制(Version Control)是構型管理的核心功能。所有置於構型識別庫中的元素都應自動予以版本的標識，並保證版本命名的唯一性。版本在生成過程中，自動依照設定的使用模型自動分支、演進。除了系統自動記錄的版本資訊以外，為了配合軟體發展流程的各個階段，我們還需要定義、收集一些元資料（Metadata）來記錄版本的輔助資訊和規範開發流程，並為今後對軟體過程的度量做好準備。當然如果選用的工具支援的話，這些輔助資料將能直接統計出過程資料，從而方便我們軟體過程改進（Software Process Improvement，SPI）活動的進行。

附錄（2/2）

- 對於構型識別庫中的各個基線控制項，應該根據其基線的位置和狀態來設置對應的訪問許可權。一般來說，對於基線版本之前的各個版本都應處於被鎖定的狀態，如需要對它們進行變更，則應按照變更控制的流程來進行操作。