第六章
軟體專案時程管理

執行單位：國立臺灣科技大學
軟體工程學程中心

大綱

- 導論
- 時程表
- 時程管理架構
- 時程管理因素
- 時程發展技術
- 時程壓縮
- 時程調整
- 人員與時程的關係
- 結論

學習目標

本章提供您了解以下項目的知識

- 認識時程表的重要性
- 如何定義工作事項
- 各種時程規劃技術
- 時程調整的策略

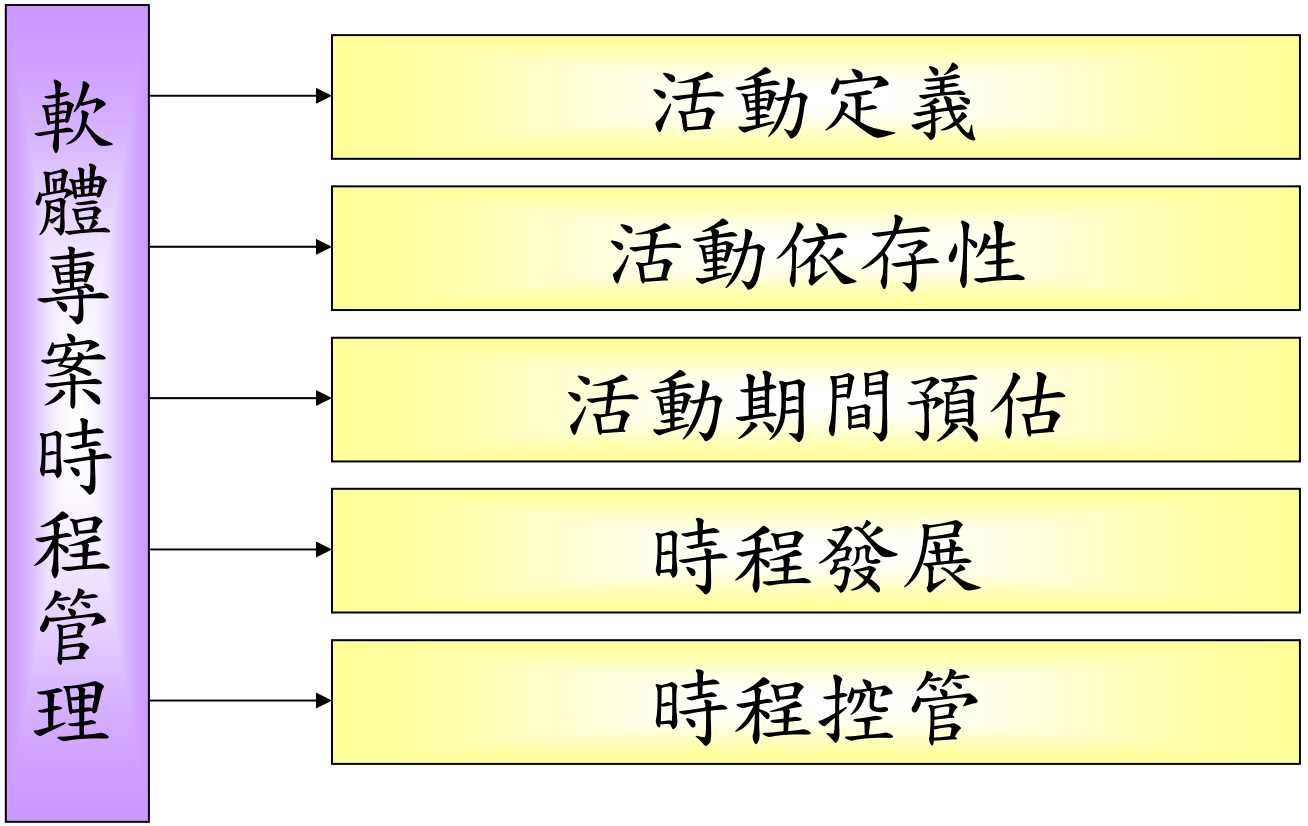
導論

- 時程管理就是軟體專案開發的「時間管理」
- 專案時程管理始於將專案分割成個別的工作項目，再確定各項目執行的先後順序，並估算所需時間和資源。
- 軟體專案的資源主要以人為主，因此時程管理即為將「人」分配至各工作項目的過程。

時程表

- 1995年CHAOS報告中提出，一個失敗的資訊專案平均花費時間為原先預估時間的222%。2001年CHAOS報告指出，平均花費時間已降低為原本的63%，但仍遠高於所預期的時間。【Joh95】
- 時程管理為確保專案能如期完成所需的工作項目，同時必須兼顧成本與時間的平衡。

時程管理架構



時程管理因素

- 活動定義：確立專案完成所需的工作項目
- 活動依存性：辨識各工作項目間的關係
- 活動期間預估：估計各工作事項完成所需時間
- 時程發展：根據活動依存性、期間預估及所需資源來建構整體時程表。
- 時程控管：控管時程表，使專案能依據所訂定之時程表來施行。

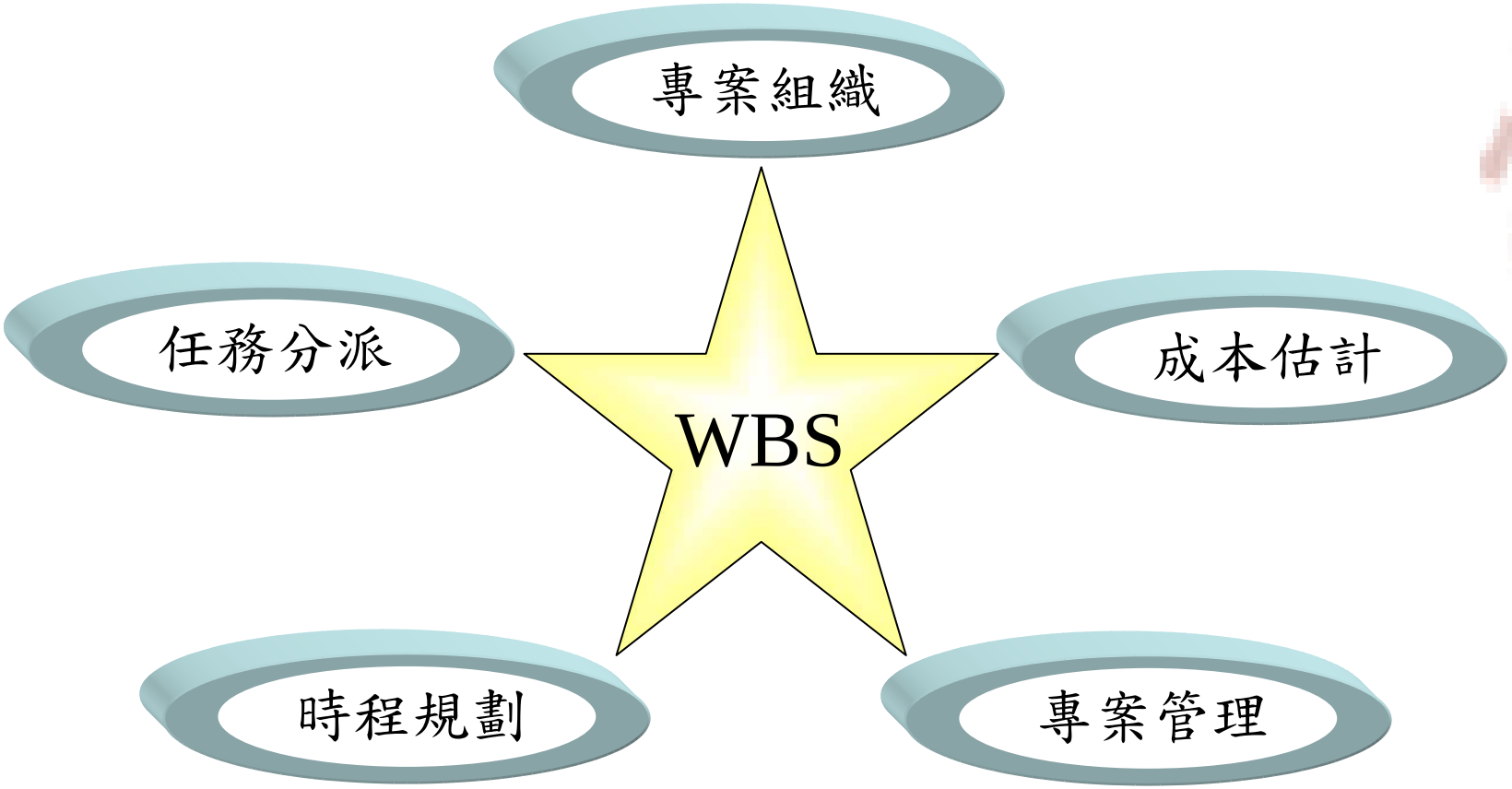
活動定義

- 專案負責人必須訂出專案起始與終止時間，同時也需估算專案所需金額。
- 專案管理者則依這些資訊來訂出時程表中的工作事項，若與原先預估差異過大，則需協調。
- 工作事項確定後，必須與使用者一同檢視，確定是否為使用者所要求。

活動依存性

- 詳閱工作分解結構(Work Break-down Structure, WBS)及產品說明、假設與限制來確定活動間的依存性。
- 三種建立相依性的因素：
  - 強制相依：表嚴謹邏輯，例在偵錯前要先將原始碼完成。
  - 任意相依：表鬆散邏輯???
  - 外部相依：專案活動與非專案活動間的關係。

工作分解與專案開發的關係



【林信惠02】

活動期間預估

- 活動期間預估需考量實際工作時間加上額外時間，其中額外時間需視實際狀況隨時調整。
- 須與專案團隊人員討論，同時也可檢閱相關計劃與諮詢專業人員來決定完成時間。

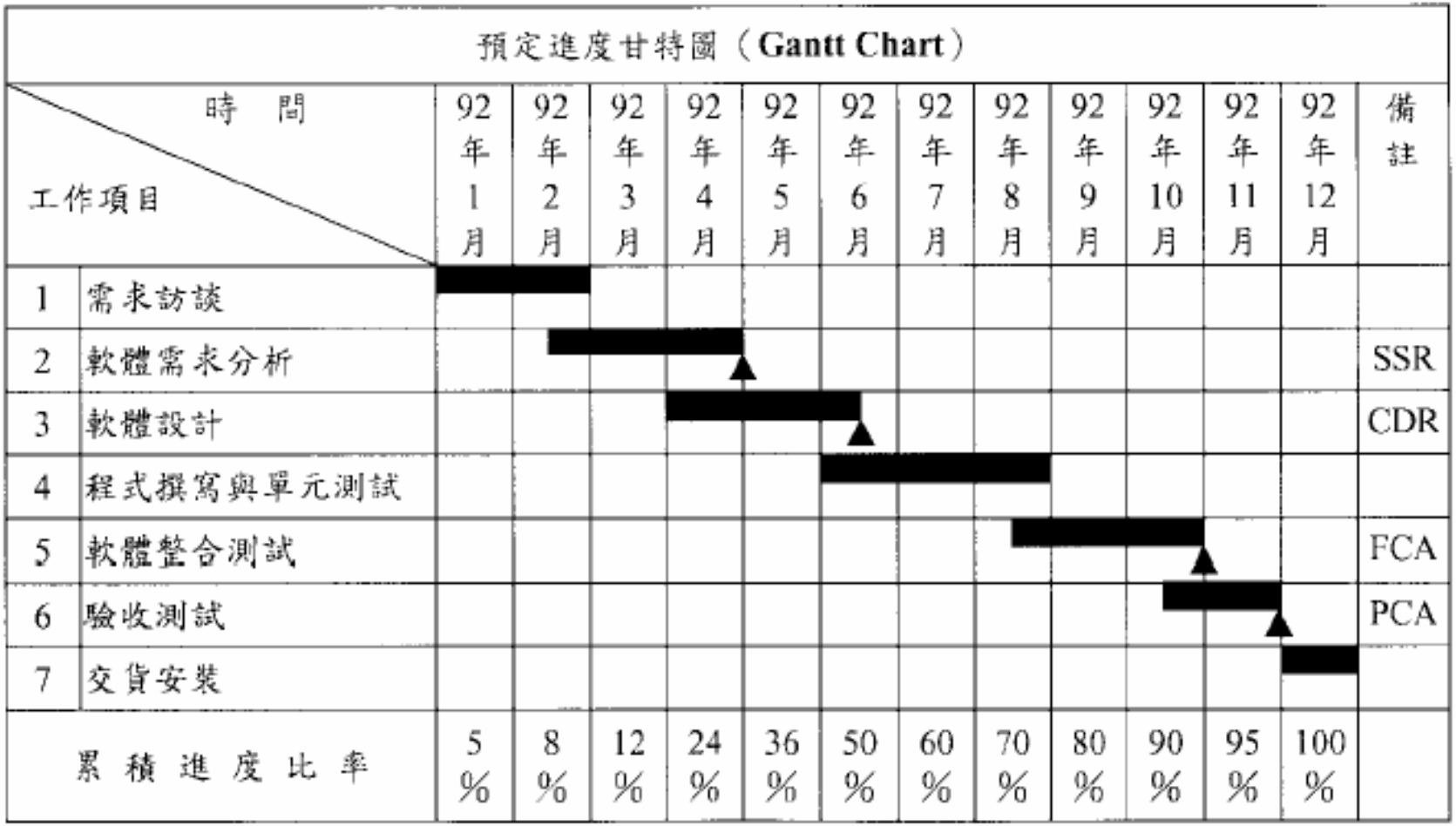
時程發展

- 在專案完成前，時程表經常需要多次修改。
- 時程表目的是要產生一個合理時間表，然後以此為基礎來控管專案的進度。

時程發展技術

- 甘特圖(Gantt chart)
- 計劃評核術(Program Evaluation & Review Technique，PERT)
- 要徑法(Critical Path Method，CPM)
- 資源平滑(Resource Leveling)
- 里程碑(Milestone)
- 同步點(Synchronize point)

甘特圖範例



預定進度甘特圖（Gantt Chart）

工作項目 \ 時間		92年1月	92年2月	92年3月	92年4月	92年5月	92年6月	92年7月	92年8月	92年9月	92年10月	92年11月	92年12月	備註
1	需求訪談													
2	軟體需求分析													SSR
3	軟體設計													CDR
4	程式撰寫與單元測試													
5	軟體整合測試													FCA
6	驗收測試													PCA
7	交貨安裝													
累積進度比率		5%	8%	12%	24%	36%	50%	60%	70%	80%	90%	95%	100%	

計劃評核術 (1/3)

- 使用有順序關係的網路邏輯圖以及加權後的平均活動時程來估算專案時間
- 事件導向分析方法，用來估計當個別活動時程有高度不確定性，整體專案所需之總時程。
- 利用機率模式來預估完成時間，由範圍預估來取代點預估。
- 最主要的功能是將風險放入考量

計劃評核術 (2/3)

- 計畫評核術估計係採用下列三因素來估計時程期望值
  - 樂觀時間 (O)　(Optimistic time)
  - 最可能時間(M) (Mostly likely time)
  - 悲觀時間 (P)　(Pessimistic time)

$$\text{PERT Weighted Average} = \frac{O + 4M + P}{6}$$

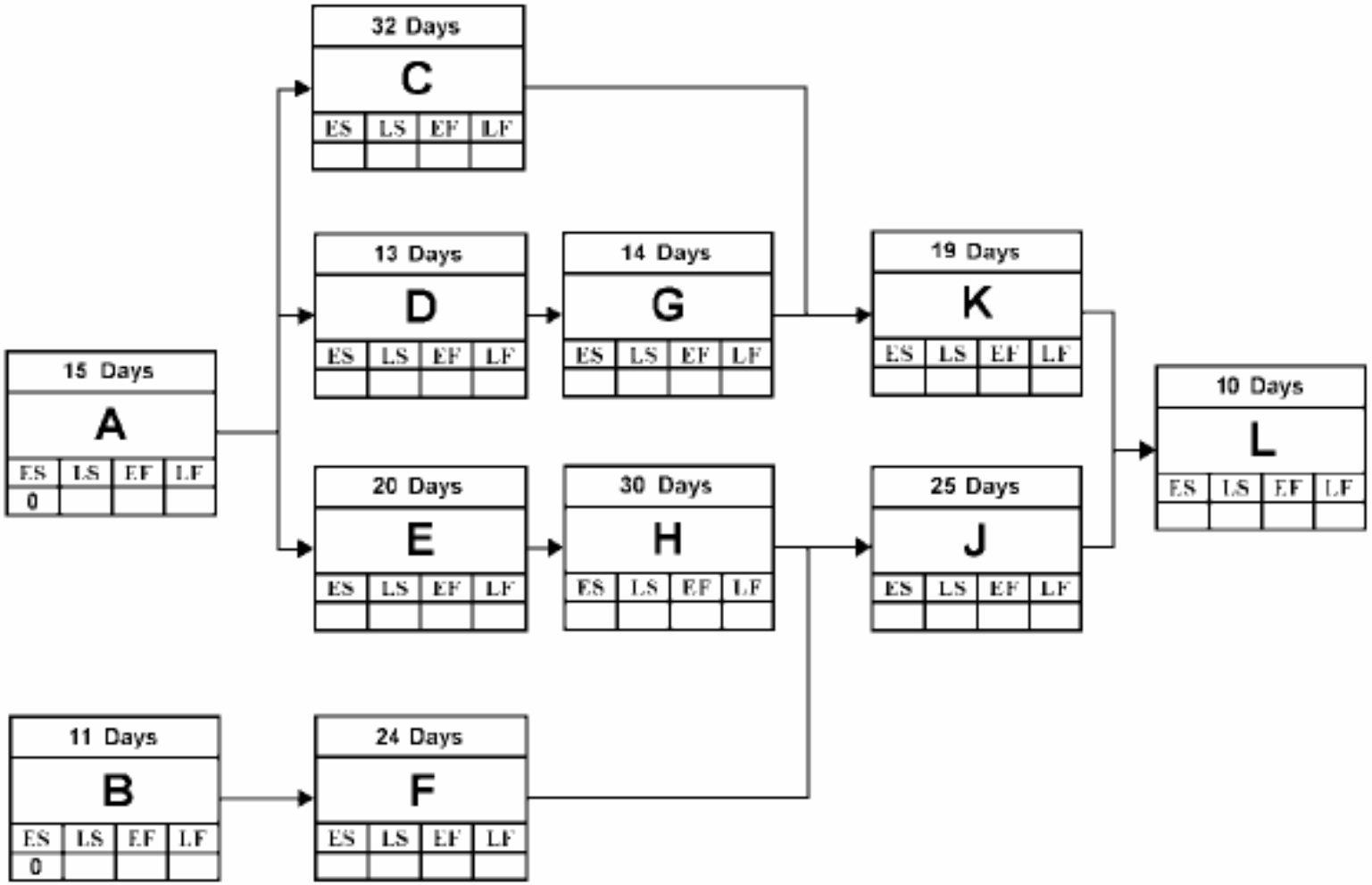
- 另一估計方法

$$\text{PERT Weighted Average} = 12 \text{ workdays}$$

計劃評核術 (3/3)

- 計畫評核術實行步驟
  - 估計樂觀時間、最可能時冒及悲觀時間。
  - 求得每個工作項目的時程期望值。
  - 求得每個工作項目的時程變異數。
  - 將關鍵路徑上每個工作的時程期望值相加，求得專案整體時程期望值。
  - 將每個工作的時程變異數相加，求得專案時程變異數。

計劃評核術範例


15 Days
A
ES LS EF LF
0
32 Days
C
ES LS EF LF
13 Days
D
ES LS EF LF
14 Days
G
ES LS EF LF
20 Days
E
ES LS EF LF
30 Days
H
ES LS EF LF
11 Days
B
ES LS EF LF
0
24 Days
F
ES LS EF LF
19 Days
K
ES LS EF LF
25 Days
J
ES LS EF LF
10 Days
L
ES LS EF LF

要徑法 (1/2)

- 要徑法又可稱要徑分析法(Critical Path Analysis，CPA)
- 關鍵路徑(Critical Path)指決定專案可最快完成的一連串事件
- 在找出關鍵路徑前，必須要先完成專案網路圖，同時也要估計來決定關鍵路徑。
- 要徑法只以時間面來作考量

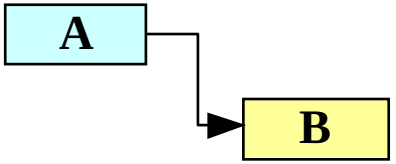
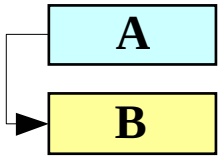
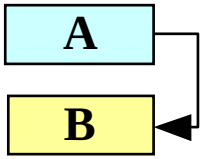
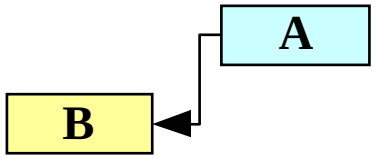
要徑法 (2/2)

- 關鍵路徑與寬鬆時間(Slacking Time)
  - 專案開始至結束有多條路徑，其中可能有一或數條路徑為專案完成的瓶頸，稱之為為關鍵路徑，若關鍵路徑上的工作進度超前或延遲將影響整體專案的完工時間。
  - 非關鍵路徑上的工作有寬餘時間可以容許工作的延遲而不會影響到整個專案的完成時間，此寬餘時間稱為寬鬆時間或浮動時間(Floating Time) 。

緩衝時間

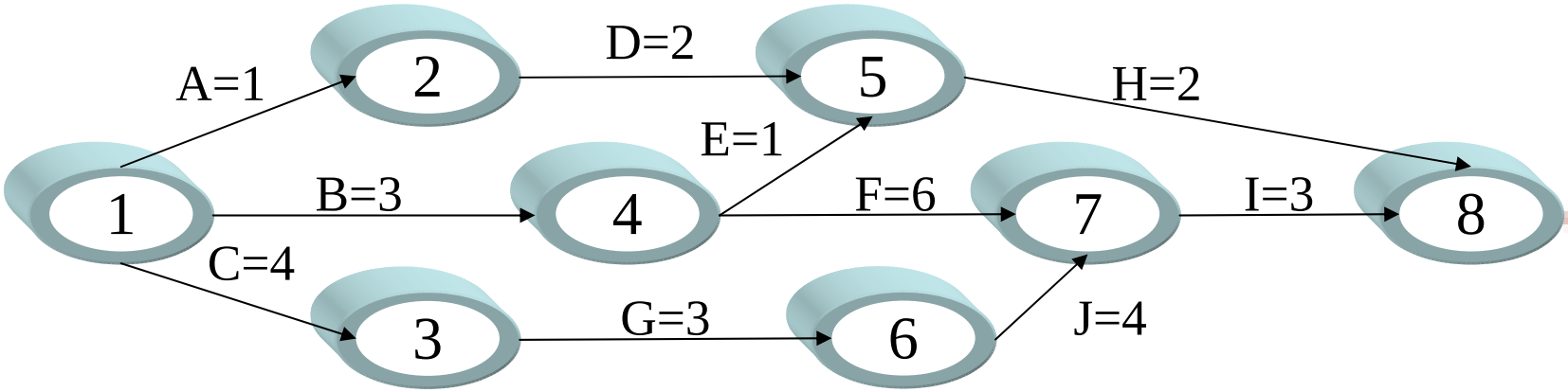
- 緩衝時間
  - 通常開發人員總是把專案時程想得很樂觀，於是延誤的情形也一再地上演著。
  - 軟體開發有著不確定的特性，為了因應非預期的延遲而加入的時間。
- 緩衝時間的形式
  - 正式的緩衝時間
  - 非正式的緩衝時間
  - 臨時性的緩衝時間
  - 分割性的緩衝時間

任務先後關係

任務順序	範　例	敘　述
完成－開始(FS)		任務B必須在任務A完成之後才能開始
開始－開始(SS)		任務B必須在任務a開始之後才能開始
完成－完成(FF)		任務B必須在任務A完成之後才能完成
開始－完成(SF)		任務B必須在任務A開始之後才能完成

專案網路圖

- 活動-箭號圖(Activity-on-Arrow，AOA)
  - Node：為事件起始點或終點
  - Arrow：表事件



資源平滑 (1/2)

- 當資源過度分配或是不平衡時，可考慮進行調配、資源限制等因素，再利用寬延時間的彈性進行資源平滑。
- 資源平滑方法有：
  - 浮動時間法
  - 任務分割法

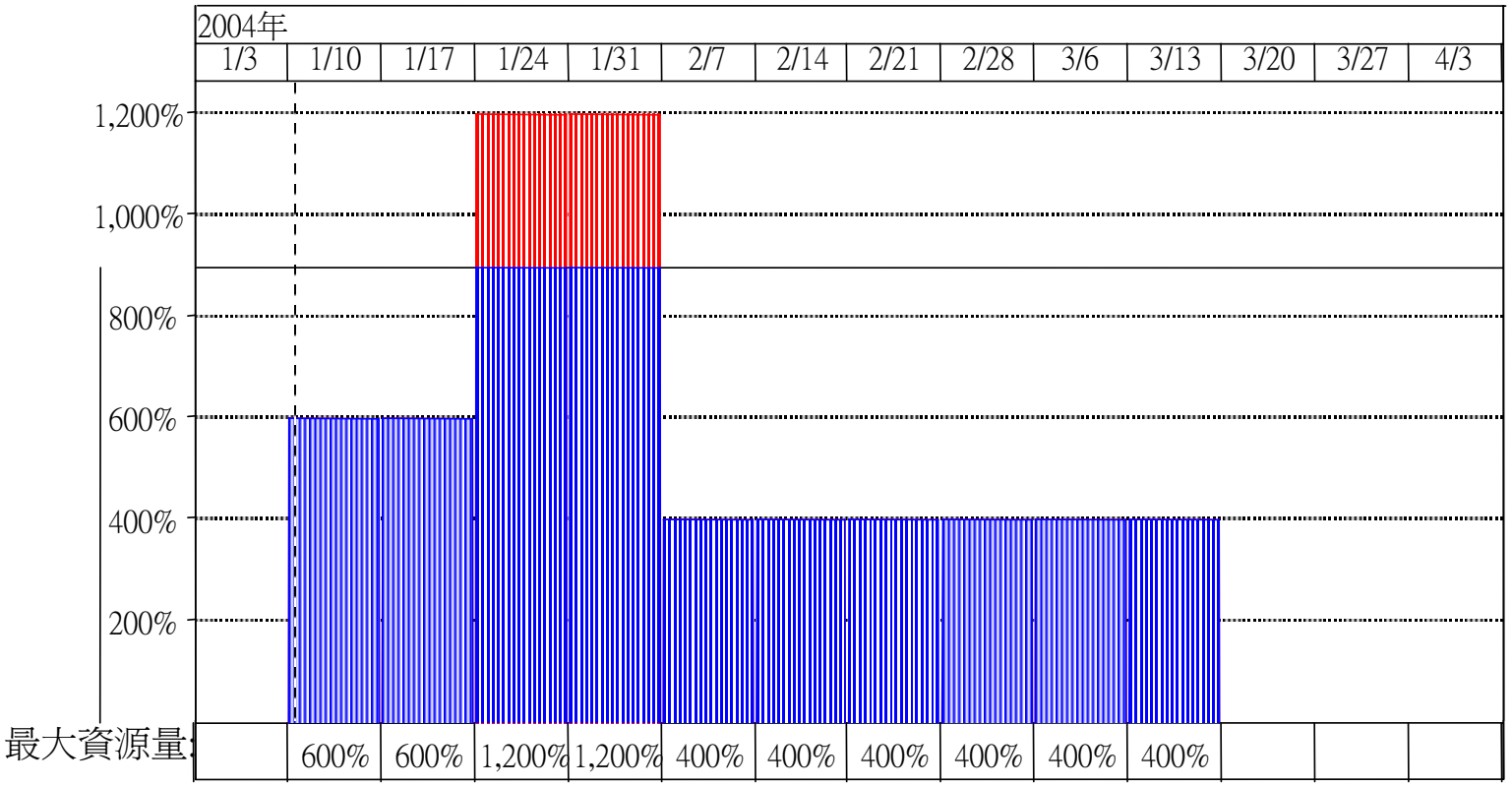
甘特圖

- 甘特圖由Henry L. Gantt於1917年所發展出來之管理工具
- 甘特圖提供標準格式來顯示專案時程資訊
- 甘特圖用來作為規劃、控制及評量專案各項工作進度，為計劃與實際進度之時序圖。

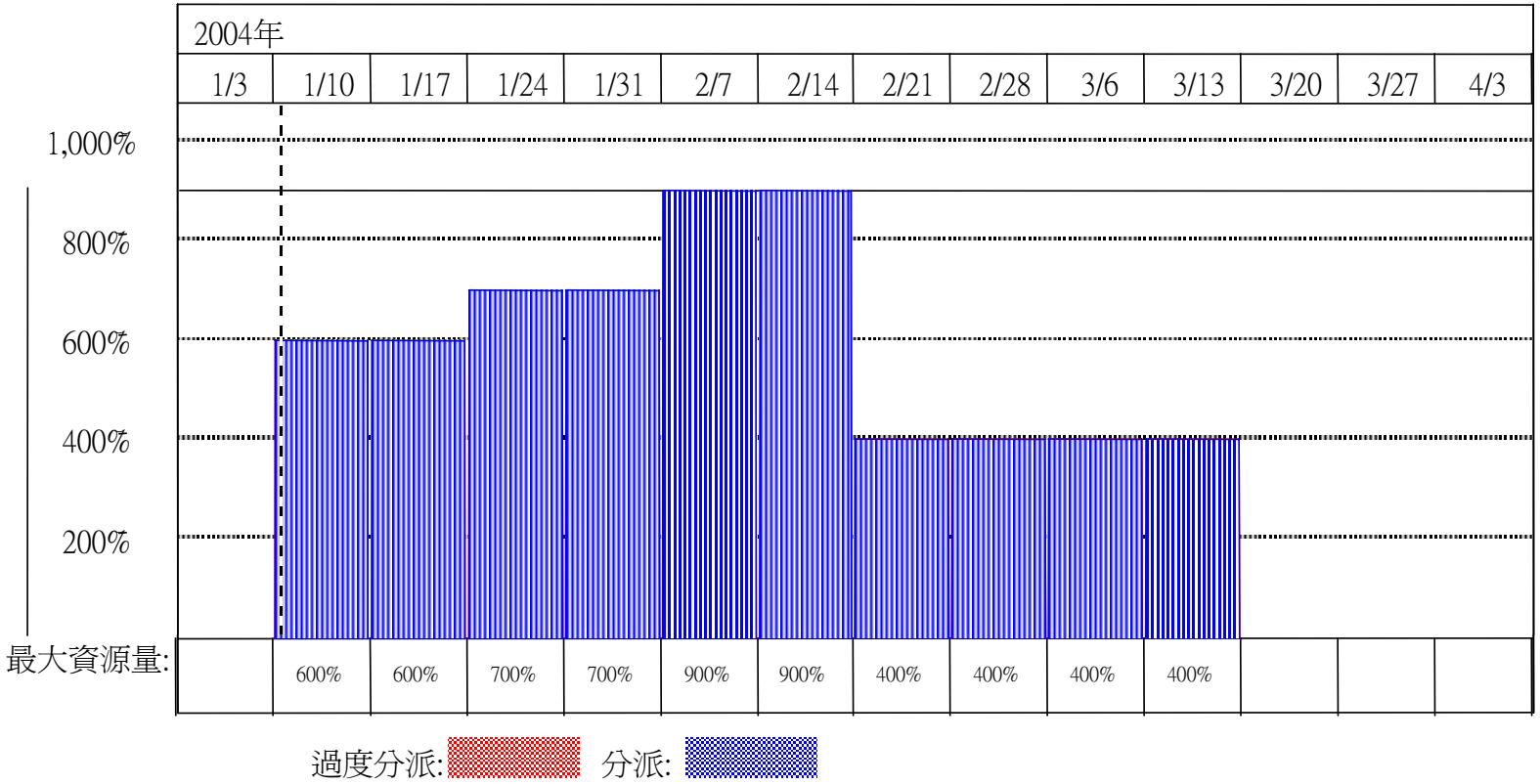
資源平滑 (2/2)

- 在有限資源下，工作分派可適度地作以下的調整：
  - 將一個人的時間分割以支援一個以上的工作。
  - 將活動細分以增加分配的彈性。
  - 指派能力強或有經驗的人負責要徑上或人力分配不足的工作。
  - 以加班、外包、購買等方式尋求內部或外部的額外資源。
  - 與顧客協商，請求配合。

過度分配下之人力負荷圖



資源平滑後之人力負荷圖



【林信惠02】

里程碑

- 在尚未製作時程表前，可能已經存在一些里程碑，例季報、展示、對顧客簡報等。
- 一旦里程碑決定後，必須要確知如何才能達成里程碑
- 估計每個里程碑功能總數前，可以由下列幾項問題著手
  - 哪些功能是團隊所有?
  - 哪些任務必須必達成?
  - 需要多少時間?
  - 需要花費多少心力?

同步點

- 同步點是用來達成同步性、狀態評估及提供產品檢測的時間點。
- 太少同步點，將會增加任務風險性；太多同步點又會使得團隊浪費過多時間在準備同步點。
- 可考慮下列四項因素來設立同步點：
  - 在設立同步點前先確定里程碑
  - 在作重大決議或風險評估前放置同步點
  - 在同步點時，確保產品為可測試、整合及重要部分功能都已完成。
  - 利用同步點評估產品狀態，降低潛在威脅性。

時程壓縮 (1/6)

- 時程壓縮的理由
  - 爭取市場先機、市場占有率
  - 因應使用者時程需求
  - 錯估時程，避免合約上的賠償
- 軟體專案三大關鍵因素
  - 時間
  - 成本
  - 品質

時程壓縮 (2/6)

- 時程壓縮的方法
  - 提高結構化與標準化的程度
  - 提高員工努力程度與人員生產力
  - 工作減量
  - 增加可用人力資源
  - 加強專案人力調配與技術選擇的最佳化
  - 修改專案結構與目標
  - 增加專案整體開發時間
  - 激勵

時程壓縮 (3/6)

- 壓縮的工作應有助於縮短整體的工期
  - 關鍵路徑上的工作項目無法縮短工期時，便無法縮短時程。
  - 避免某一工作的時程縮短，卻造成其它工作的時程延長。
- 壓縮後不會影響合理的先後關係
- 合理的成本效益
  - 部份壓縮時程的方法將增加成本的支出
  - 長時間工作可能造成效率的低落

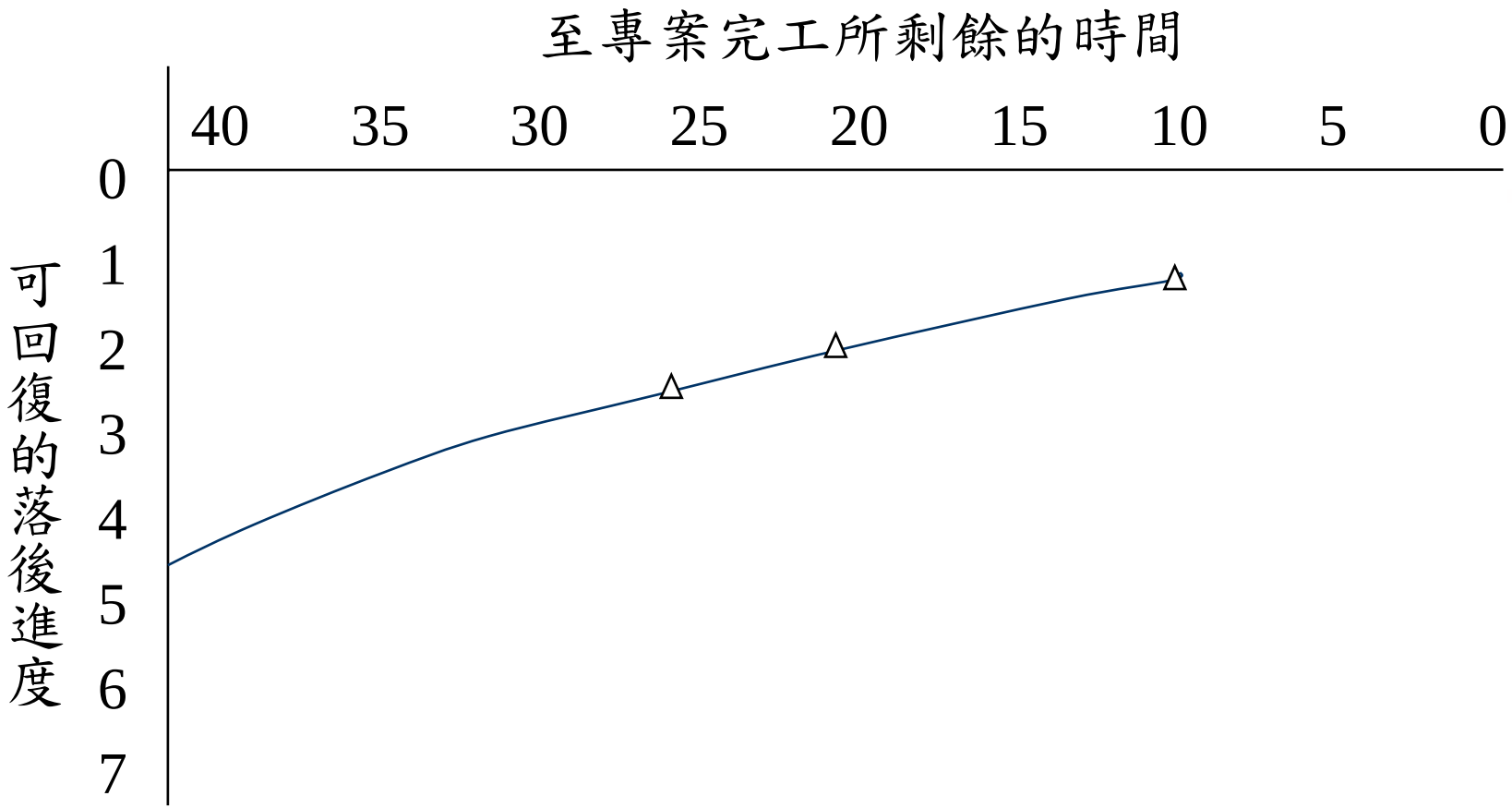
時程壓縮 (4/6)

- 不同開發階段的壓縮成效
  - 不同開發階段的壓縮成效有顯著的差異
  - 分析、編碼階段的時程縮短效果高於整合測試階段。
  - 編碼階段的壓縮成本大於分析階段
  - 分析階段壓縮時程對專案品質的影響明顯小於編碼與整合測試階段的影響

時程壓縮 (5/6)

- 壓縮成效
  - 專案延遲愈少、困難度愈低，資源愈充裕、規模愈大、目標愈明確，其時程壓縮成效也愈好。
  - 如果兼顧成本與品質因素，則激勵為最有效的時程壓縮策略。

時程壓縮的限制



不同開發階段之適用與避免的壓縮方法 (1/2)

專案現況特性 / 壓縮績效		分析階段	編碼階段	整合測試階段
時程縮短	適用	• 提高結構化與標準化的程度	• 提高結構化與標準化的程度 • 激勵	• 激勵
	避免	• 工作減量	• 工作減量	• 增加可用人力資源 • 工作減量
專案成本	適用	• 激勵	• 激勵	• 工作減量 • 激勵
	避免	• 增加可用人力資源	• 增加可用人力資源 • 增加專案整體開發時間	• 增加專案整體開發時間

不同開發階段之適用與避免的壓縮方法 (2/2)

專案現況特性 / 壓縮績效		分析階段	編碼階段	整合測試階段
專案品質	適用	• 提高結構化與標準化	• 激勵 • 工作減量	• 激勵
	避免	• 修改專案結構與目標	• 增加可用人力資源 • 提高員工努力的程度與生產力	• 其它方法

【林信惠02】

不同延遲情況下之適用與避免的壓縮方法

專案現況特性 / 壓縮績效		低延遲專案	高延遲專案
時程壓縮	適用	• 提高結構化與標準化的程度 • 增加專案整體開發時間	• 修改專案結構與目標 • 工作減量
	避免	• 工作減量	• 提高員工努力程度 • 增加專案整體開發時間

【林信惠02】

不同專案困難度之適用與避免的壓縮方法

專案現況特性 / 壓縮績效		困難度低	困難度普通	困難度高
時程壓縮	適用	• 提高員工生產力與努力程度 • 激勵	• 提高員工生產力與努力程度 • 激勵	• 沒有適當的壓縮方法
	避免	• 工作減量	• 工作減量	• 提高員工生產力與努力程度 • 增加專案開發時間

【林信惠02】

不同資源充裕程度之適用與避免的壓縮方法

專案現況特性 / 壓縮績效		資源充裕	資源不充裕
時程壓縮	適用	• 所有壓縮方法皆有效 • 以激勵的效果最好	• 所有壓縮方法皆不佳
	避免		

【林信惠02】

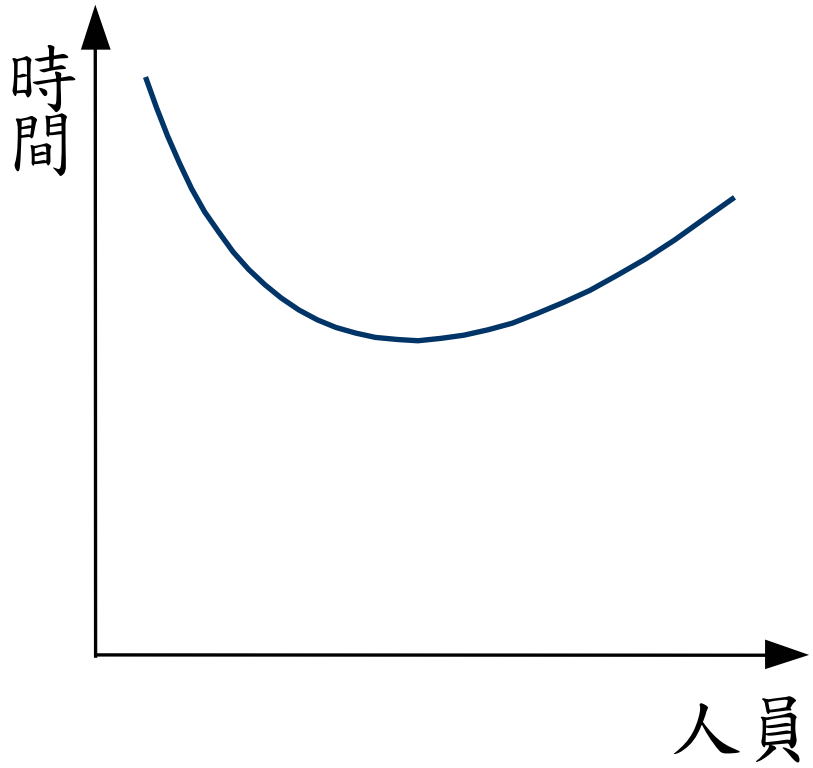
時程調整

- 分段完成
- 縮減功能
- 尋求外包

人員與時程的關係

- 專案管理者必須具備下列領導特質：
  - Enpowerment
  - Incentives
  - Discipline
  - Negotiation
- 專案管理者必須讓團隊成員能有責任感，且利用相關軟體來協助達成時間管理。
- 軟體專案愈到末期，新增人員反而愈不利專案的執行。
- 軟體專案開發過程之人力需求呈現動態的變化，必須事先規劃因應。

時間-人員關係圖



軟體專案

結論

- 軟體專案日益複雜與龐大，應用時程管理技術可有效增加專案規劃的效率與效能，也有助於追蹤與控制。
- 注意工作分派與資源平滑，以避免過度分派或負荷過重的情形。
- 因應上市時機或顧客需求，軟體專案經常會縮短開發時程，然而時程壓縮有許多策略，使用時需考量其適合度。
- 若案無法在期限內完成時，專案管理者應與使用者協商，並採取必要調整策略因應之。

自我評量

- 時程管理架構為何
- 時程發展技術共有幾種
- 何謂計劃評核術
- 甘特圖的作用為何
- 時程調整有幾種作法

參考資料

- 【Joh95】Johnson, Jim, “CHAOS-The Dollar Drain of IT Project Failures,” Application Development Trends, Jan., 1995(www.standishgroup.com/chaos.html)
- 【林信惠02】林信惠, 黃明祥, 王文良, “軟體專案管理”,初版, 第六章, 頁165-200, 智勝文化, 2002
- Joel Henry, “Software Project Management, A Real-World Guide to Success”, Addison Wesley, Chapter7, pp.125-156, 2004
- Kathe Schwalbe, “Information Technology Project Management”, Course Technology, Chapter 5, pp.120-156, 2002
- “PMBOK”, 2000
- “PMBOK中文版”, 2001