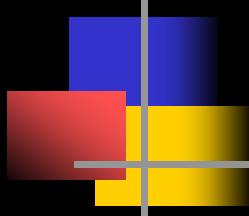


資料庫系統理論與實務 第 0 章 緒 言



Norway, by Frank S.C. Tseng
(<http://www2.nkfust.edu.tw/~imfrank>)

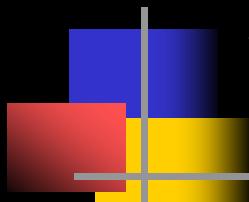


本章內容

- 0.1 前言
- 0.2 資料處理模式的演進
- 0.3 資料庫系統處理架構的演進
- 0.4 資料分析模式的演進過程
- 0.5 資料處理的未來發展

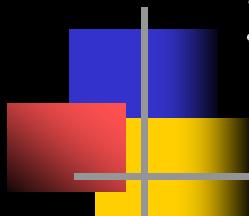
前 言

- 從以下三個維度探討資料處理的過去、現在與未來之演進，以涵蓋目前市面上的系統類型
 - 資料處理模式的演進
 - 資料處理架構的演進
 - 資料分析方式的演進
 - 資料庫系統的未來發展
 - 資料庫管理系統 (Database Management System) vs. 資料庫系統 (Database Systems) 兩者在本書中的區別
 - 前者指廠商直接販售的資料庫管理套裝軟體，後者範圍則擴大包含：使用者、資料、軟體、硬體等



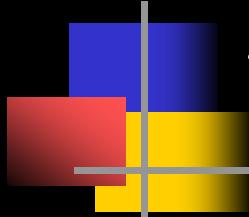
資料處理模式的演進

- 第一階段—人工檔案管理。
- 第二階段—電腦化檔案系統
 - 電腦化循序檔案系統。
 - 電腦化直接存取式檔案系統。
- 第三階段—記錄為處理單元的資料庫管理系統
 - 階層式資料模式 (Hierarchical Data Model)
 - 網路式資料模式 (Network Data Model)
 - 關聯式資料模式 (Relational Data Model)



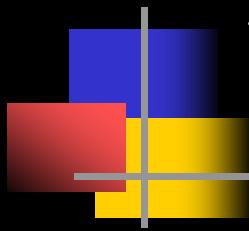
資料處理模式的演進 (續)

- 第四階段—以物件為處理單元的物件導向式資料庫管理系統
 - 物件導向式資料庫管理系統 (Object-Oriented Database Management Systems, OODBMS)
 - 物件關聯式資料庫管理系統 (Object-Relational Database Management Systems, ORDBMS)
- 從資料庫系統的角度看, ORDBMS 比較被看好



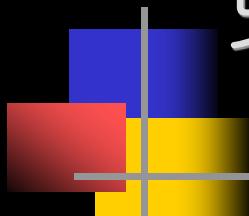
資料處理模式的演進（續）

- 第五階段—以 XML (Extensible Markup Language) 為處理單元的「XML 原生資料庫管理系統」(Native XML Database Management Systems)
 - 傳統的資料庫管理系統加上 XML 功能只能歸類為 XML-Enabled 系統
 - Native XML 是從根本上採用與傳統資料庫管理系統完全不同的架構來開發，以方便將 XML 文件做為處理單元的角度來做考量



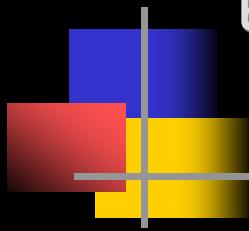
第一階段—人工檔案管理

- 如：戶政、病歷、圖書館藏書資料
- 限制與缺點：
 - 在眾多卡片中做搜尋、加總等動作
 - 儲存的檔案也越來越多，佔掉不少空間
 - 無法同時供眾多人調閱使用
 - 卡片之間的關係須由人工來記錄、管理
 - 容易被弄丟、毀損、竊取或拷貝帶出去
 - ...



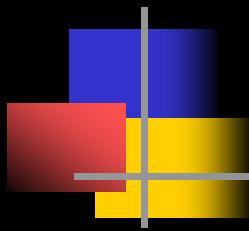
第二階段—電腦化檔案系統

- 依照儲存媒體又可分成以下兩類：
 - 電腦化循序檔案系統 (Computerized Sequential File Systems): 早期使用紙帶、卡片、磁帶，甚至錄音帶儲存資料
 - 電腦化直接存取檔案系統 (Computerized Direct Access File Systems): 近來則廣泛使用軟式磁碟片、硬式磁碟機、CD、DVD、隨身碟儲存資料



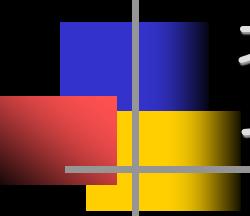
電腦化循序檔案系統 (缺點)

- 透過紙帶、卡片、磁帶等媒體來記錄資料
- 逐項在檔案中對特定項目做搜尋、加總動作
- 仍然要佔掉不少空間
- 資料重覆，維持資料的一致性不容易達成
- 檔案內部的相依關係 (Intra-file Dependency) 或 檔案與不同檔案之間的關係 (Inter-file Dependency)
難以記錄、處理
- 磁帶更容易被毀損、竊取或拷貝



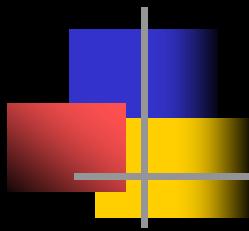
電腦化直接存取式檔案系統

- 磁碟取代磁帶，使得電腦具備直接存取式檔案系統 (“Direct Access” File System)
- 但存取的對象仍然僅限於「檔案」(Files)
- 所以仍然具有第二階段中的所有缺點
- 只不過存取速度較快罷了！



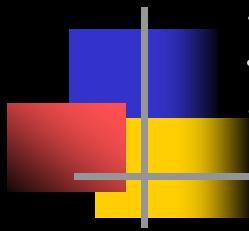
第三階段—以記錄為單元的 資料庫管理系統

- 將檔案再次細分，成為一筆筆的記錄 (Record)
- 形成「資料庫管理系統」 (Database Management Systems, DBMS)，依照其結構可以分成以下三類：
 - 階層式資料庫管理系統 (Hierarchical DBMS)：樹狀的結構組織 (**不可以有迴路**)
 - 網路式資料庫管理系統 (Network DBMS)：網路狀的結構組織 (**可以有迴路**)
 - 關聯式資料庫管理系統 (Relational DBMS)：關聯式的結構組織 (平面表格結構)



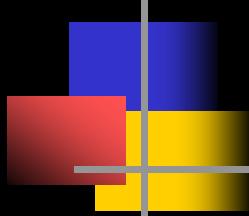
階層式資料庫管理系統

- 稱為 Hierarchical Database Management System
- 以單筆記錄為處理單位
- 將記錄以樹狀結構方式組織起來，以利進一步的處理。
- 詳見 2.2 節。



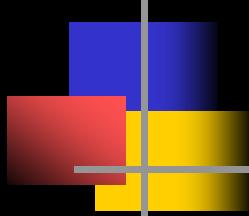
網路式資料庫管理系統

- 樹狀結構並不能反應某些真實情況
- 記錄之間可能具有網路的連結關係
- 所以將記錄以網路狀結構方式組織起來，稱為 Network Database Management Systems
- 詳見 2.3 節
- 資料相依性：記錄之間都是以鏈結方式串連在一起，因此在撰寫應用程式時，便必須要牽就鏈結之間的結構狀態



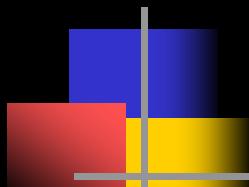
關聯式資料庫管理系統

- E.F. Codd 於 1970 年提出 Relational Database Model
- 以關聯式方式來組織資料間的關係
- 強調應用程式不應與資料的內部結構有依存關係的「資料獨立性」(Data Independence)，讓應用程式不須牽就資料結構的異動而做大幅度的修改
- 資料庫管理系統的研究與發展進入新的里程碑
- 獲 1981 年度 Turing Award — 電腦界最高榮譽
<http://www.acm.org/awards/taward.html>



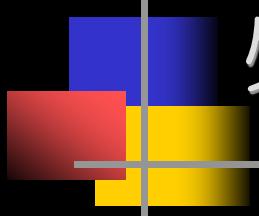
關聯式資料庫管理系統 (續)

- 將記錄集在概念上以表格方式組織起來
- 應用程式所面對的是表格般的結構，不需了解其內部組織
- 實際上表格中的記錄可能在內部組織上是以樹狀或網路狀的方式來鏈結
- 讓表格可以直接拿來做運算處理
- 表格之間的相關性則可以透過欄位的資料值來聯繫



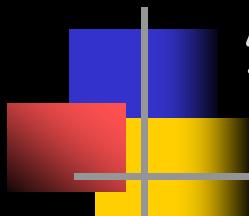
關聯式資料庫管理系統 (續)

- 有人曾批評系統的效能會變得其慢無比
- 就像「高階語言」(High-Level Language) 與「組合語言」(Assembly Language) 之爭一樣
- 電腦的速度越來越快，系統的複雜度也越來越高，效能變得不是問題
- 目前市場上的主流產品，其資料組織方式以集合方式運算與傳輸，也是讓分散式資料庫成為可行的主要原因
- 其主流地位未來可能會慢慢被物件關聯式資料庫管理系統所取代，但並不會消失，兩者各有其舞台



第四階段—物件為處理單元的 物件式資料庫管理系統

- 單純的表格方式無法表示真實世界裡的繼承關係與組合關係
- 當代的系統分析、設計、程式撰寫：
 - 「物件導向式系統分析」(Object-Oriented Analysis，簡稱 OOA)、
 - 「物件導向式系統設計」(簡稱 OOD)
 - 「物件導向式程式設計」(簡稱 OOP)
- 終究不能忽視物件導向化的趨勢

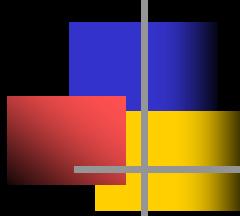


物件式資料庫管理系統

- 依照其發展方式又可分成以下兩類：
 - 物件導向式資料庫管理系統 (OODBMS : Object-Oriented Database Management Systems) ,
主要是從程式語言的設計觀點出發
 - 物件關聯式資料庫管理系統 (ORDBMS : Object-Relational Database Management Systems) ,
主要是從資料庫與查詢語言的設計觀點出發

物件導向式 資料庫管理系統 (OODBMS)

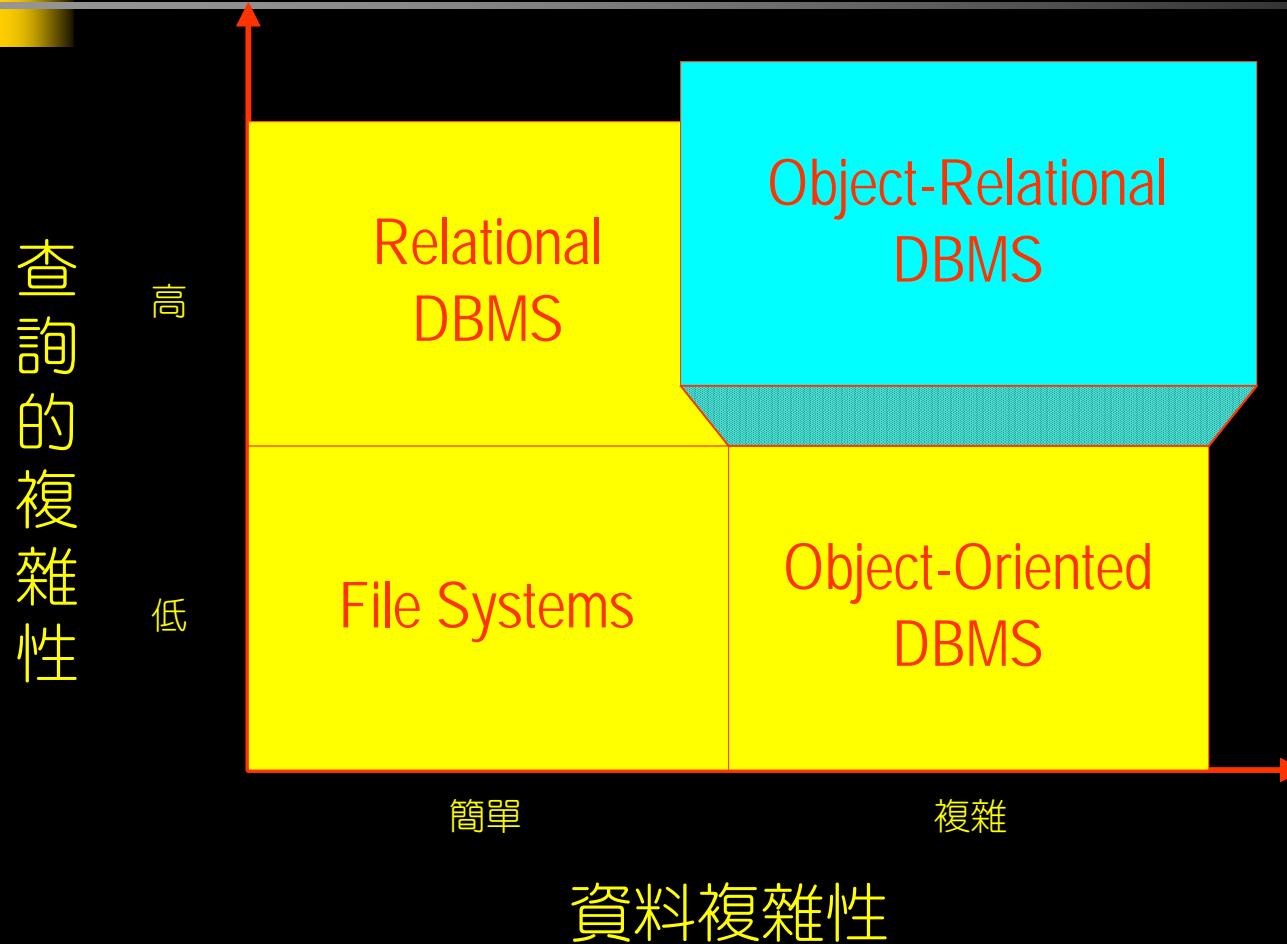
- 以物件 (Objects) 的觀念來代替記錄 (Records)
- 以 “永久式程式語言” (Persistent Language) 與 C++ 或 Smalltalk 做緊密結合的系統
- 主要是以物件程式語言為基礎朝資料庫管理方向發展。
- 原始目標是希望直接以程式語言的功能取代關聯式資料庫的功能，形成所謂的 “資料庫程式語言” (Database Programming Language)
- 並不會逼使得關聯式資料庫管理系統沒有發展空間
- 產品如 O₂, Versant, Ontos, Objectstore, Itasca, Objectivity, Gemstone, ...

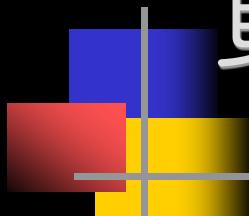


物件關聯式 資料庫管理系統 (ORDBMS)

- 主要是以關聯式資料庫為基礎朝物件化方向發展。
- 強調由關聯式資料庫進步到物件導向式資料庫的演進，不是如 OODBMS 所掀起的「革命」(Revolution)，而是「進化」(Evolution) 或「進一步的包容」
- 由於 90% 的商業應用已被「關聯式資料庫管理系統」佔據，預估「物件關聯式資料庫管理系統」只會慢慢成為未來資料庫管理系統的主流之一
- 產品如：PostgreSQL, UniSQL (Dr. W. Kim), Informix Universal-Server (Dr. M. Stonebraker) 、Illustra 、IBM DB2 、SQL Server 2008 、Oracle 10g ，... 等。

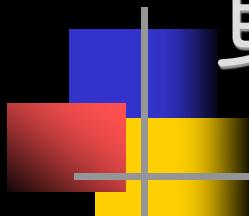
資料庫管理系統矩陣





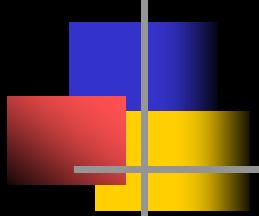
資料庫管理系統矩陣 (續)

- 左下角象限的應用以檔案系統來提供服務是最佳的選擇。這類應用如：文書排版系統，文字編輯系統，等
- 左上角象限的應用主要在商業資料處理。以關聯式資料庫管理系統來提供服務最佳。
- 右下角象限中的應用需要複雜資料的支援，但不常做搜尋動作，以物件導向式資料庫管理系統來提供服務最佳。



資料庫管理系統矩陣 (續)

- 右上角象限的應用需要處理與查詢複雜的資料，最好是用物件關聯式資料庫管理系統來提供服務
- 尤其是 Web-Based 系統在 Content-Based Management 上的應用：
 - 個性化 Homepage、客戶分析、...

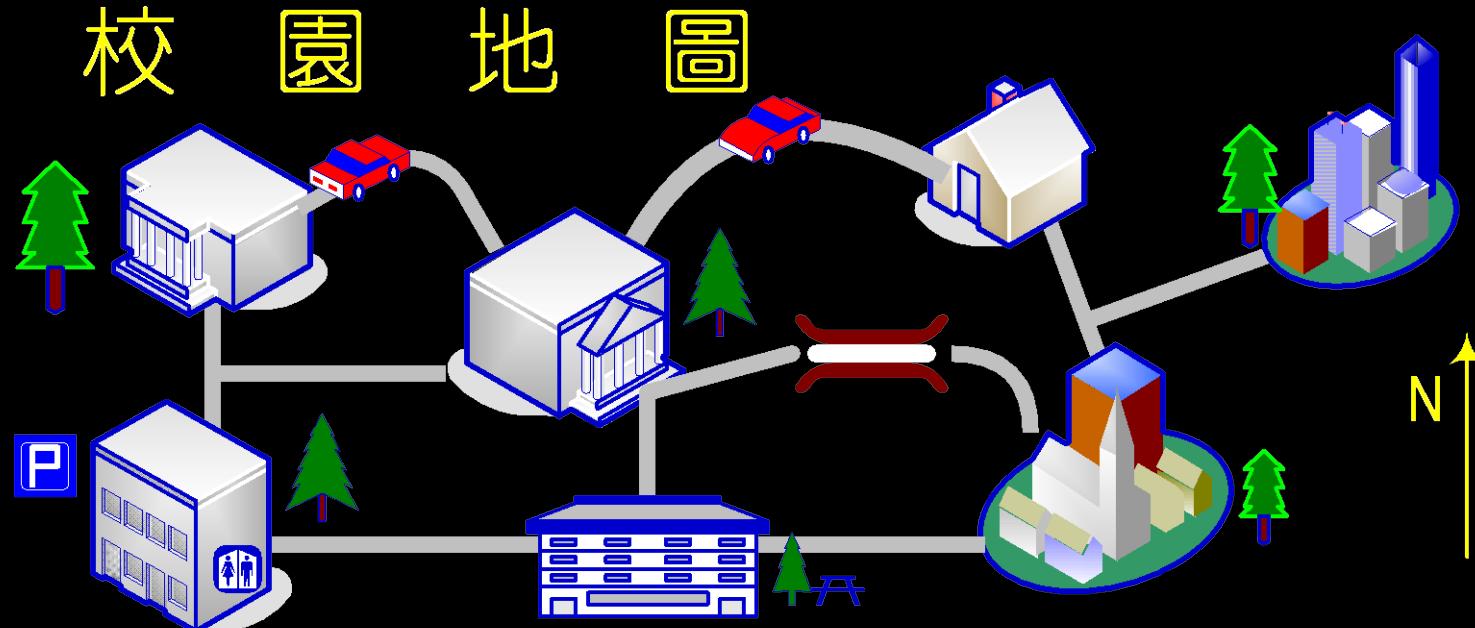


當代/未來處理複雜資料之需求

- 多媒體的全球資訊網 (WWW) vs. 資料庫
- 電腦輔助設計、製造 (CAD, CAM, CASE)
- 地理資訊系統 (Geographic Information Sys., GIS)
- 決策支援系統 (Decision Support System)
- 歷史/時間資料庫 (Temporal Database)
- 知識管理系統 (Knowledge Management)
- ...

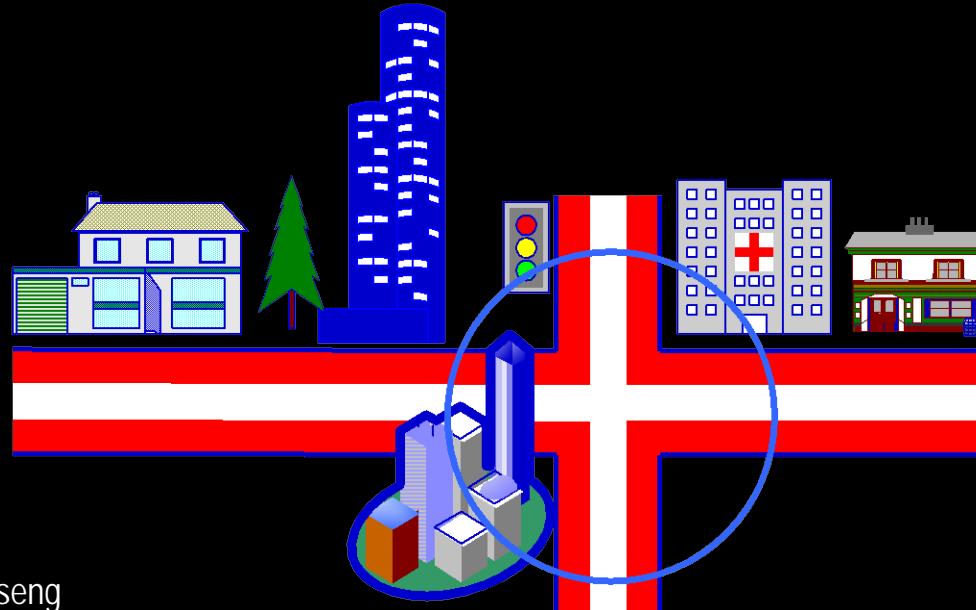
空間與地理資料查詢

- 請查出座落在校園東南方的建築物
- 請找出在科學館西方的建築物
- 請找出科學館到體育館的最短路程



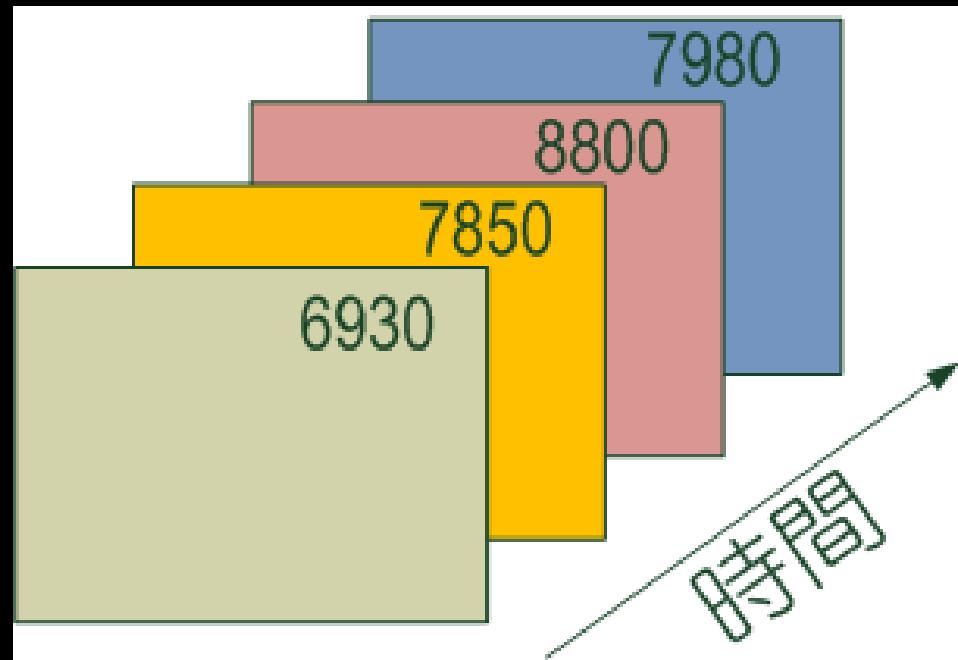
以內容調閱資料或照片 (Content-Based Retrieval)

- 請找出含有夕陽西下景色的照片
- 請找出居住在縣政府方圓兩公里內的員工清單
- 請找出居住在所有危險十字路口方圓五十公尺內的保險客戶，並提高其保費



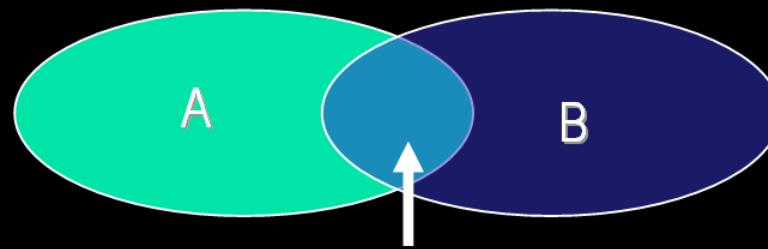
時間序列查詢 (時間資料庫)

- 請找出民國八十年到八十五年間的台塑股價波動情形



資料的互補:三個臭皮匠 vs. 諸葛亮

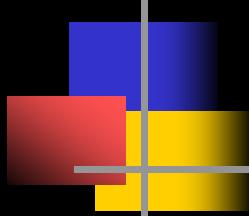
- 利用物件關聯式資料庫管理系統來整合各類資料庫管理系統 (異質性資料庫系統)
- 資料庫 A 顯示：超級颶風將襲擊墨西哥灣沿岸 (含美國德州, 路易斯安那州與佛羅里達州)
- 資料庫 B 顯示：德州與佛羅里達州地區集中了美國 24% 的煉油業和 30% 的石油產品出口企業
- 整合後得知：石油價格即將上漲



下一波無法抵擋的浪潮: ORDBMS

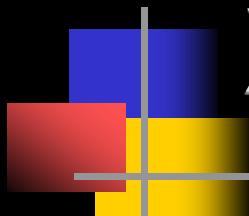
■ 兩股推動力量

- 新興多媒體電腦化趨勢
(WWW, 數位相機, 數位圖書館, 隨選視訊...)
- 商業與醫療，等資料處理漸漸向右上角靠攏
(商情分析、X光照片、超音波、保險理賠的照片分析)
- 物件化的資料可以使用 XML 來加以表示與包裝



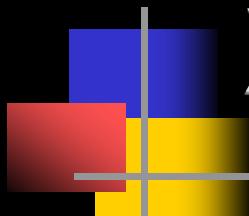
第五階段—以 XML 為處理單元的 XML 原生資料庫管理系統

- 在 Internet 上有大量資料交換的需求。使用 XML (Extensible Markup Language) 作為資料交換的媒介已經是業界的共識。
- 與 XML 相關的各種應用程式變化、組合與重組關係正快速地蔓延開來
- 經濟部技術處也成立了 XML 台灣資訊網 (<http://www.xml.org.tw>) 供各界參考



XML 原生資料庫管理系統

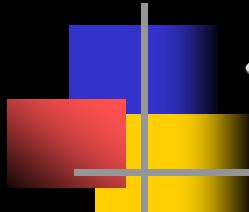
- 可以直接使用 XML 的格式與結構來存取資料
- 知名的 XML 原生資料庫管理系統 (Native XML DBMS):
 - Software AG (<http://www.softwareag.com>) 的 Tamino XML Server。
 - Apache 的 Xindice (<http://xml.apache.org/xindice>)。
 - Open Source eXist (<http://exist-db.org>)。
 - X-Hive Corp.'s X-Hive/DB (<http://www.x-hive.com>)。
 - Object Design (是 eXcelon 公司的一個研發部門) 所提出來的 eXcelon (<http://www.exceloncorp.com>) 產品 (目前稱為 XIS)。



XML 原生資料庫管理系統 (續)

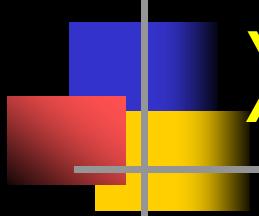
■ 知名的 XML 原生資料庫管理系統產品 (續):

- 4Suite Server (<http://4suite.org/index.xhtml>) 是由 Fourthough Inc, (<http://fourthough.com>) 所開發。
- dbXML (<http://www.dbxml.com>)。
- GoXML DB (<http://www.xmlglobal.com>)。
- Stanford 大學所發展的 Lore (<http://www-db.stanford.edu/lore>)。
- Ixia Inc 所發展的 TEXTML Server (<http://www.ixiasoft.com>)。
- OpenLink Software 的 virtuoso (<http://www.openlinksw.com/virtuoso>)。
- XDBM (<http://sourceforge.net/projects/xdbm>)。
- 由 B-Bop Associates 所發展的 Xfinity Server (<http://www.b-bop.com>)。
- XYZFind Server (原本由 XYZFind Corporation 所發展，後來被 Interwoven 併購)。



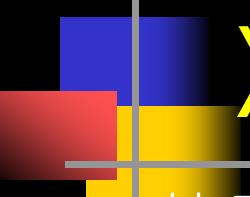
值得進一步討論的議題

- 傳統資料庫與 XML 原生資料庫之間的優、劣勢何在？
- 關聯式資料庫廠商對 XML 的擴充功能：
 - MS SQL Server 2000/2008 針對 SQL 擴充了 XML 的功能、
 - Oracle 提出 iFS (Internet File System) 架構 for XML 資料、
 - IBM 將 DB2 加上了 XML Extender，讓使用者能將資料以 XML 格式編碼後，透過傳統關聯式查詢介面來存取
- 以上產品也通稱為 **XML-Enabled DBMS**。
- **XML-Enabled** 與 **Native-XML** 兩類資料庫管理系統差異何在？
- 是否我們應該全面採用 **XML 原生資料庫**來取代傳統的資料庫存取方式？還是讓兩者並存？



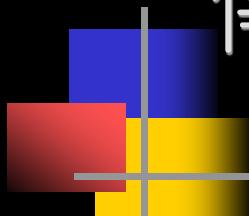
XML 原生資料庫的特點

- 針對 XML Documents 定義邏輯資料模式 (Logical Data Model)，並根據此資料模式來存取文件：
 - 該模式包含：XML 元素 (XML Elements)、XML 元素中的屬性 (Attributes)、PCDATA，以及文件的順序等組織方式。
 - 目前已經有針對此類資料模式的實作成品問世，例如：DOM (Document Object Model) 與 SAX (Simple API for XML)。
- 使用 XML 物件 (XML Objects) 與類別 (Class) 的型式來存放資料。



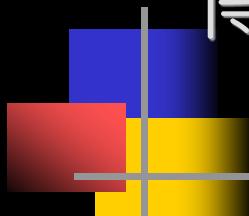
XML 原生資料庫的特點 (續)

- 使用 XML Documents 做為邏輯儲存的單元。不像關聯式資料庫使用「值組」(Tuples) 以及「關聯表」(Relations) 做為邏輯儲存單元。
- XML 原生資料庫使用專屬的儲存格式。例如：索引檔 (Index Files) 或壓縮檔 (Compressed Files)。不像關聯式資料庫使用 B-Tree 等方式來存放資料。
- 具有兩種不同的架構：
 - Data-Centric XML 文件 (也稱為 Text-Based)：適合用在銷貨資料管理、帳務管理、基本資料管理，等結構化資料 (Structured Data)
 - Document-Centric XML 文件 (也稱為 Model-Based)。適合用在半結構化 (Semi-Structured Data) 或非結構化 (Non-Structured Data) 的文件，如：專案文件、會議記錄、公文、廣告或型錄等。



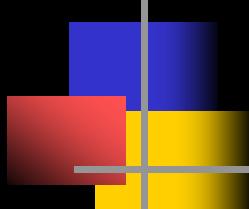
傳統資料庫 vs. XML 原生資料庫

- 關聯式資料庫在值組 (Tuples) 與屬性 (Attributes) 等單元資料值 (Data Elements) 上有很清楚的對應關係，因此容許快速而靈活的商業計算。
- 不過，直接的對應關係也造成了將關聯式資料庫資料對應到 XML 結構時，會面臨相當大的挑戰
 - 因為，我們必須隨著 XML 上所提供的各種不同排列與組合等擴充方式，來調整與增加關聯式資料庫中的表格關聯性 (Relationships)



傳統資料庫 vs. XML 原生資料庫(續)

- 短時間內 XML 原生資料庫並不能取代關聯式資料庫在商業上的應用範疇。原因：
 - 關聯式資料庫已經在商業應用上佔有相當龐大的應用範圍
 - XML 原生資料庫算是一種用來提供自由格式的 XML 文件強固儲存體，以及處理相關應用的工具而已。
- XML 原生資料庫有自己的舞台：
 - 公司內部資訊入口網站管理、會員資料庫管理、
 - 產品目錄管理、出版管理、病歷資料追蹤，及
 - B-to-B 文件交換，或跨組間的工作流程管理

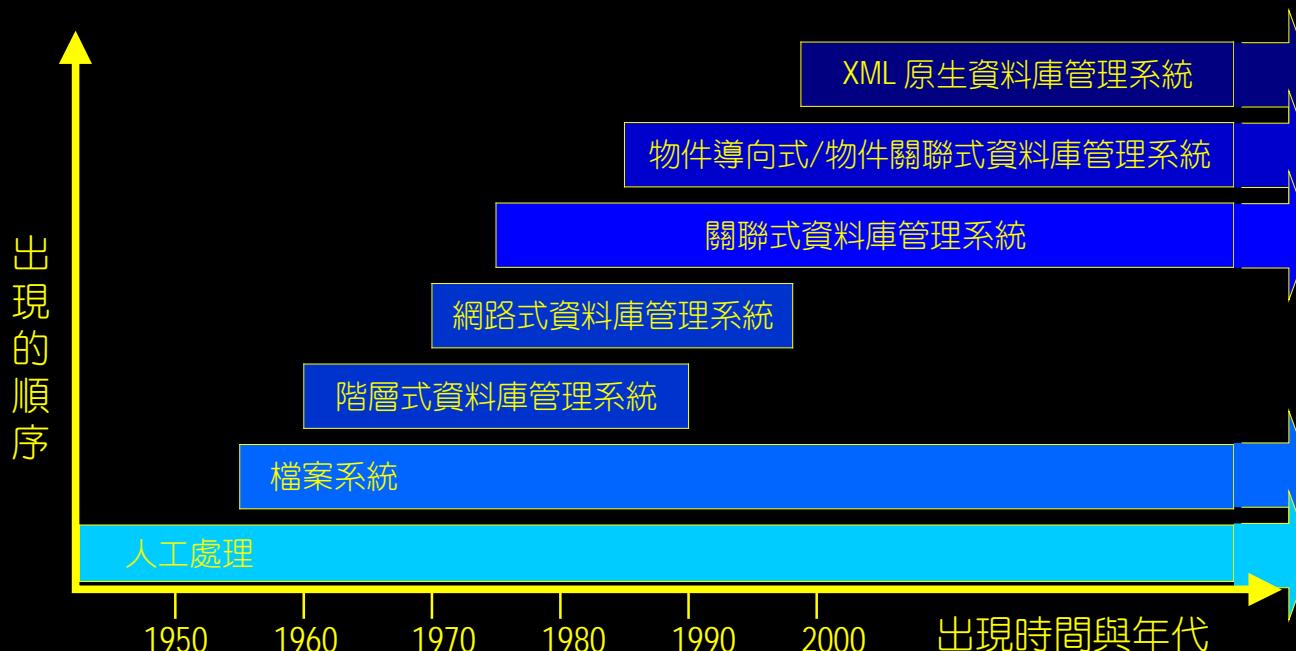


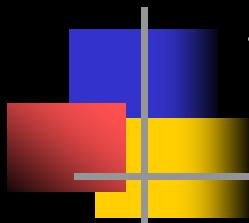
我們的結論 (僅供參考)

- 若資料的結構是一致的 (Homogeneous) 或是結構化的 (Structured) 則使用傳統關聯式資料庫較適合，
- 若資料的結構是不規則的 (Heterogeneous) 或是半結構化的 (Semi-Structured)，那就使用 XML 原生資料庫、文件管理系統 (Document Management Systems)，或是內容管理系統 (Content Management Systems)。
- 若能適當結合以上兩種資料庫，則將會能得到最佳組合，因此兩者應是形成互補而非競爭的關係

五個階段各種系統的圖示

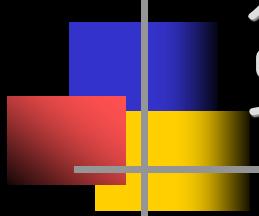
- 有些已經禁不起時代的考驗，面臨淘汰的命運，如：階層式與網路式資料庫管理系統已逐漸式微。
- 有些則仍有其應用範疇，如：檔案系統。





資料處理架構的演進

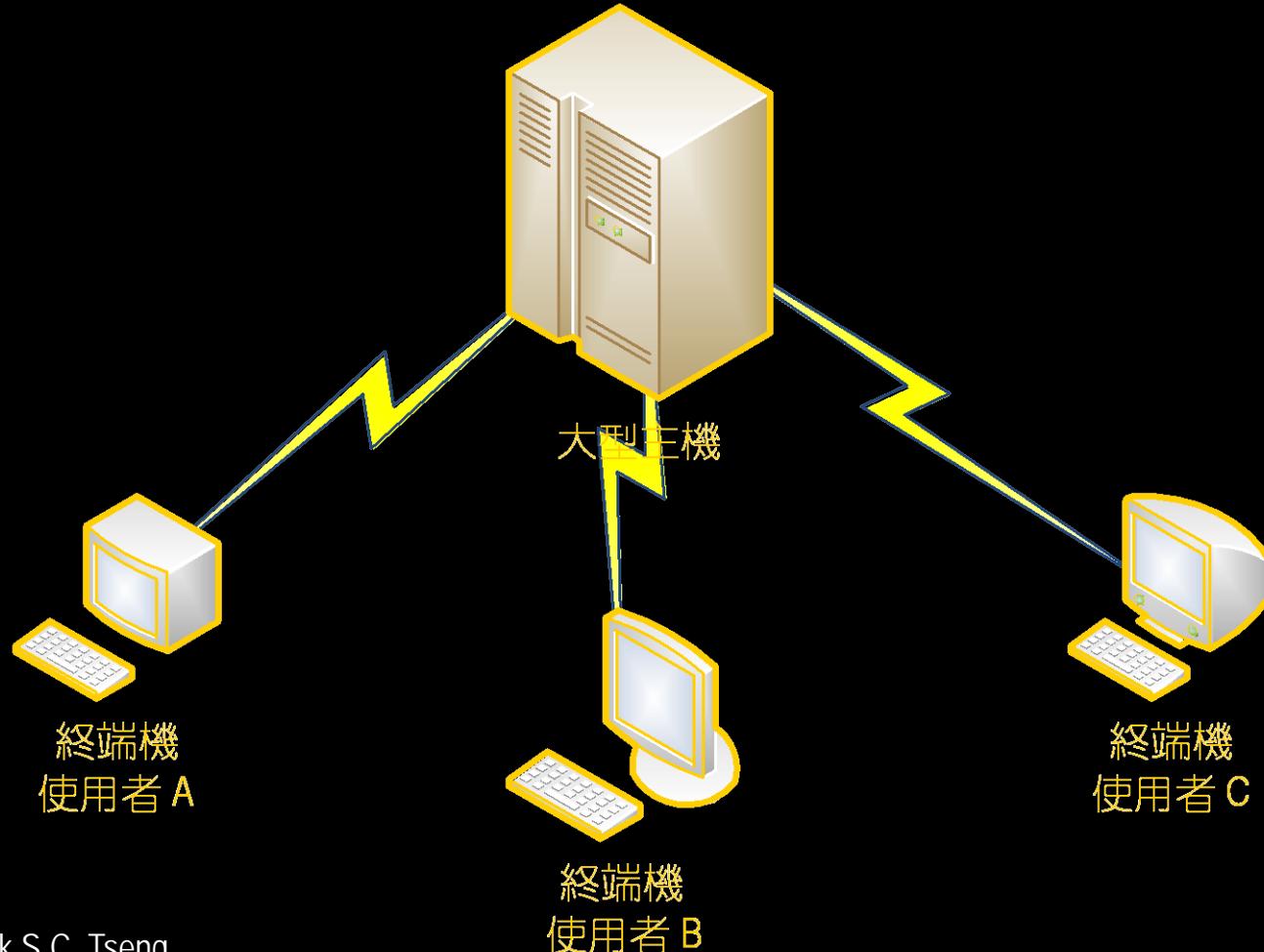
- 第一階段—集中式處理 (Centralized Processing)
- 第二階段—兩層主-從式架構 (2-Tier C/S Arch.)
- 第三階段—多層主-從式架構 (Multi-Tier C/S Arch.)
- 第四階段—同質性分散式處理 (Homogeneous Distributed Processing)
- 第五階段—異質性分散式整合 (Heterogeneous Distributed Processing)
- 第六階段—行動計算處理 (Mobile Computing)



第一階段—集中式處理

- 傳統大型主機所使用
- 前端客戶以終端機與主機相連
- 由主機一手包辦所有的工作。
- 不過當資料量與使用者增多時，即使是快如超級電腦者，如此包辦所有事務於單一主機上，也會有力不從心的時候

傳統「集中式處理」

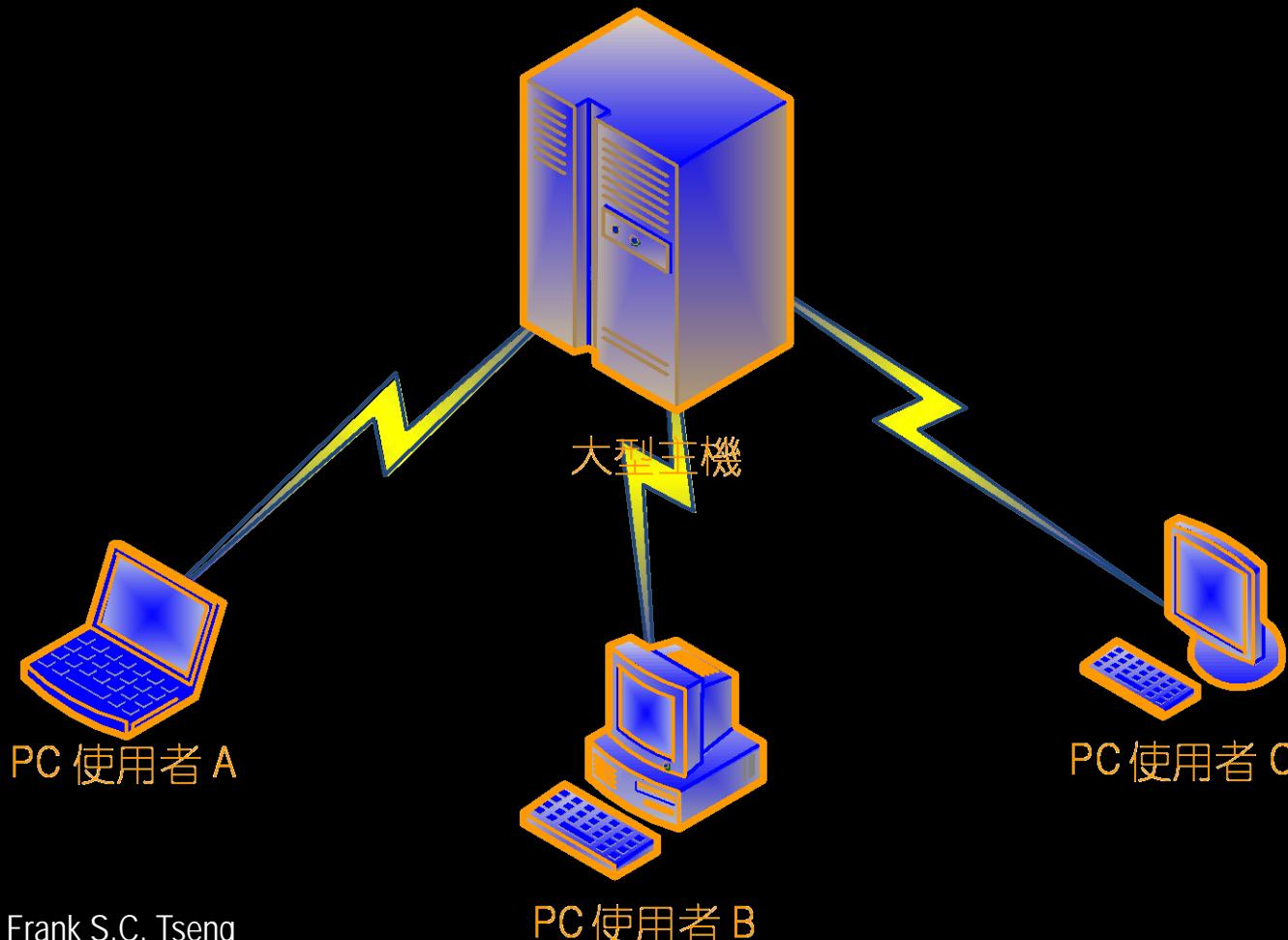


第二階段—

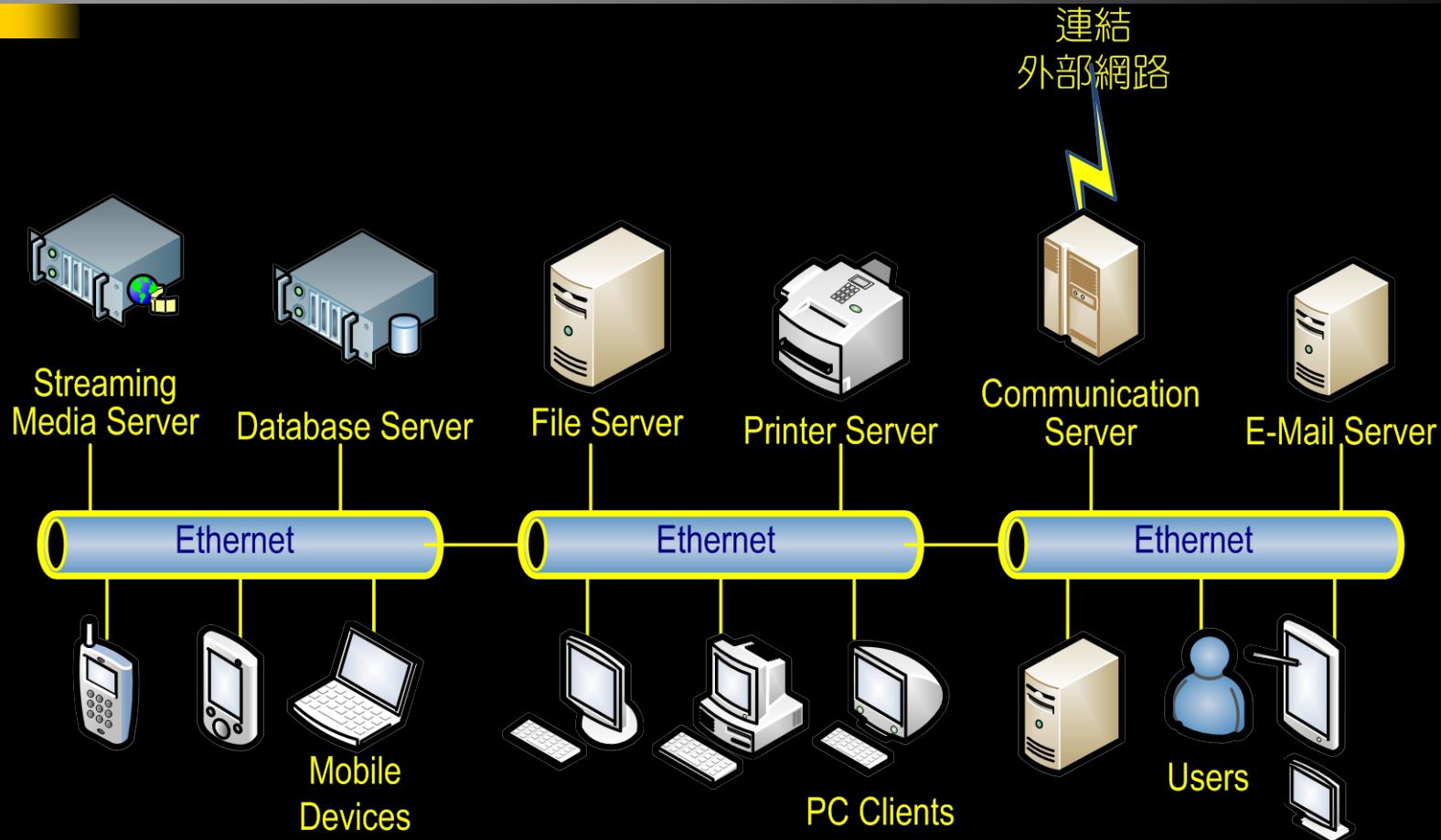
兩層式(2-Tier) 主-從式計算

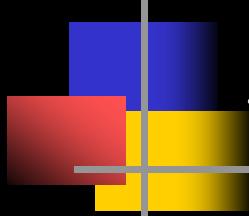
- 個人電腦與工作站的價格越來越低，且功能不弱，可以分攤部份主機的工作, e.g.,
 - 顯示畫面或視窗畫面的處理，
 - 排序主機送來的資料，或
 - 過濾不必要的錯誤訊息等。
- 「主機」慢慢變成了「伺服器」(Servers) 的概念並演變成: 電子郵件、檔案伺服器、資料庫伺服器

兩層式主-從式架構處理



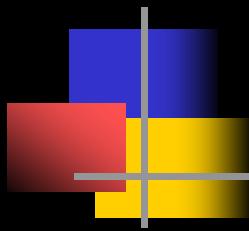
複雜的兩層式主-從式架構





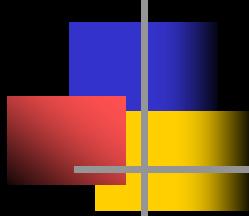
客戶端與伺服端的溝通方式

- 此種架構下的運算方式，大致上是以
 - 「客戶端」(Client) 透過標準的應用程式介面 (Application Program Interface, API)、或是
 - 利用遠端程序呼叫 (Remote Procedure Call, RPC) 的方式來與伺服端做聯繫，



兩層式主-從式架構的優點

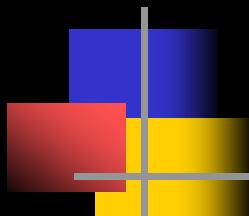
- 在企業組織方面
 - 企業再生工程 (Re-Engineering)，改變企業的流程與組織精簡化，再造企業的第二春
 - 易隨著組織的擴大而擴增新系統
 - 呼應使用者的呼聲，提昇工作效率
 - 提供多媒體與異質性整合環境
 - 跟隨時代脈動，改善企業形象



兩層式主-從式架構的優點

■ 在資訊系統方面

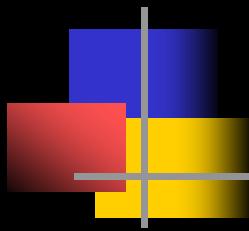
- 妥善率 (Availability) 提高了
- 整體系統的可靠度 (Reliability) 也增加了。
- 可隨時加入各種不同的新伺服器。
- 資源共享更易達成。
- 可充分調整與運用處理效率。
- 親和性的多媒體使用者介面，
- 大大地降低了企業在電腦硬體上的投資等。



兩層式主-從式架構的缺點

■ 在管理方面

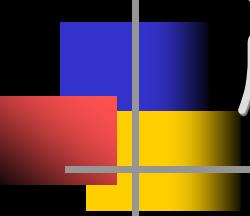
- 軟、硬體執行平臺的不明確與產品更新的快速，造成了產品供應的不穩定性。
- 產品間的不相容問題，增加採購時的不確定性。
- 專業人員的訓練不易以及無法長時間為企業效力的隱憂。
- 資料的安全性策略不易掌握以及網路的管理政策不易推展。



兩層式主-從式架構的缺點

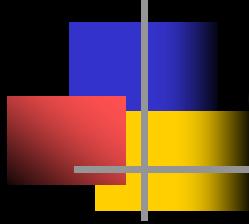
■ 資訊系統方面

- 軟、硬體的「異質性」（Heterogeneity）導致複雜度提高，造成管理上的困難，與調整不當所造成的效能低落。
- 資料的保密工作是一項挑戰。
- 初期系統開發成本高，優勢無法立現。
- 圖型化的介面可能使得操作速度減慢
- Right-sizing 而非 Down-sizing



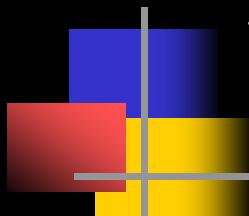
產品走向與必備知識

- 系統產品紛紛走向
 - 「開放性」(Open Systems)
 - 「分散性」(Distributed Systems)
 - 「整合性」(Integrated Systems)
- 電腦專業人員的必備知識
 - 「電腦網路」
 - 「資料庫系統」
 - 「資訊安全與保密」



業界開放性共同介面 (*de facto* standard)

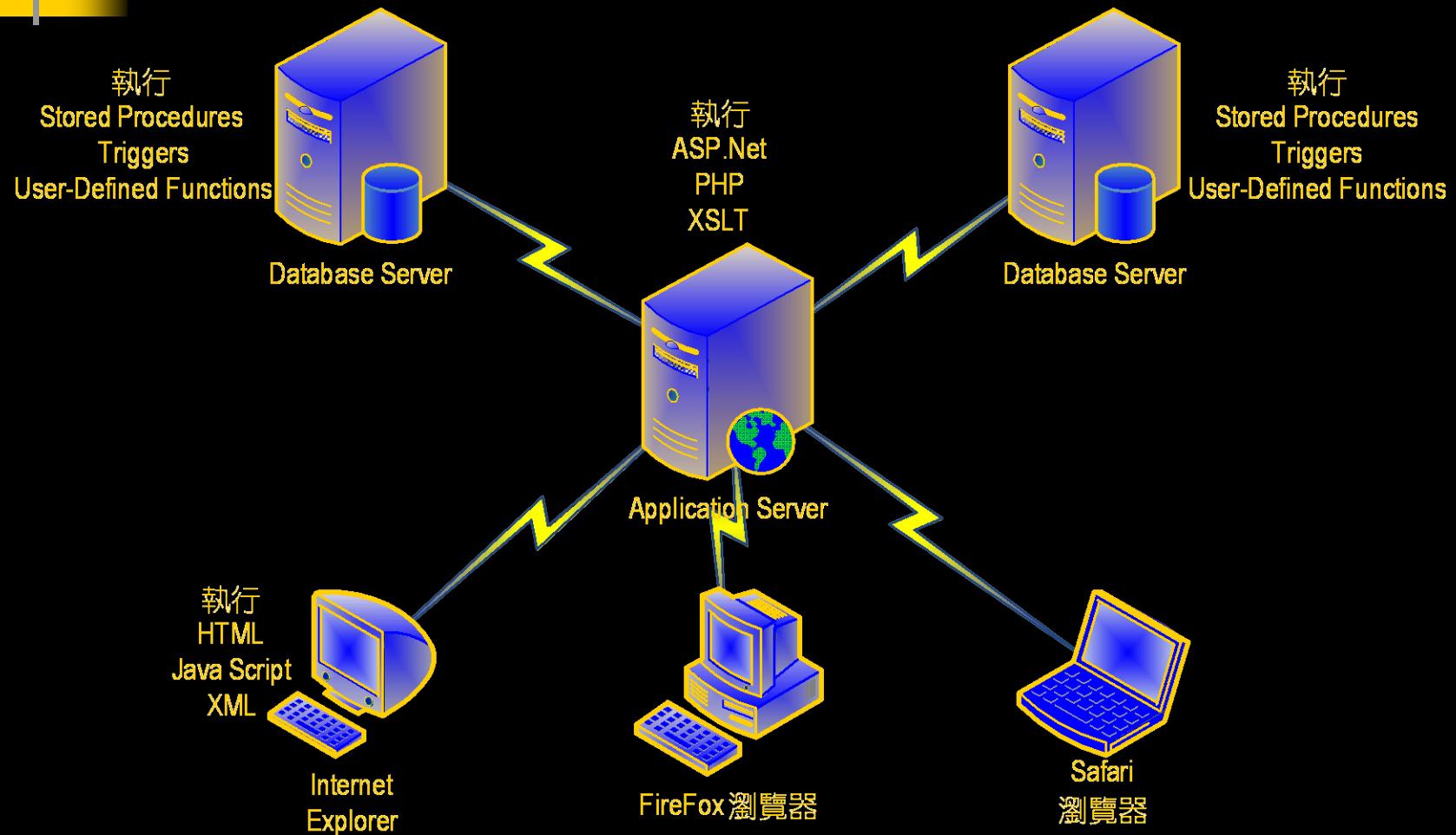
- 作業平臺 (Hardware Platform)
- 作業系統層次 : POSIX, UNIX, Windows, ...
- 跨作業系統層次 : DCE, ONC, Java, ...
- 網路層次 : TCP/IP
- 資料庫互通介面 : SQL, ODBC/JDBC, IDAPI, ...
- WWW Browser, XML,
- ...

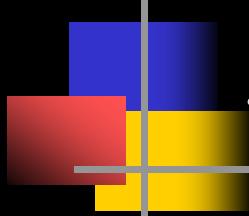


第三階段—多層式架構

- World-Wide Web (WWW) 舉起,
- 瀏覽器 (Web Browser) 與相關發展工具造成了多層式主-從架構的發展
- 主要以「三層式發展架構」(3-Tier Arch.) 為主一分成客戶端 (Client)、應用伺服器 Application Server (通常與Web 伺服器 (Web Server) 整合在一起)，與資料庫伺服器 (Database Server) 三個層次
- 融合了集中式與兩層式主-從架構的優點
- 只是效能較差，但近年來已漸漸改善與提升效能

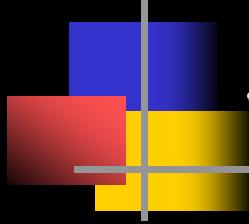
三層式應用系統發展架構





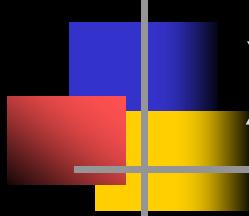
三層式發展架構的優點

- 融合了集中式管理與兩層式主-從架構的優點
- 客戶端只要裝上瀏覽器便可使用各種應用程式。
- 資料均由伺服端集中掌控，管理上較不會有兩層式主-從架構所引發的缺點。
- 客戶端可選擇瀏覽器，並可由伺服端透過ODBC/JDBC等共通介面，存取不同類型資料庫。



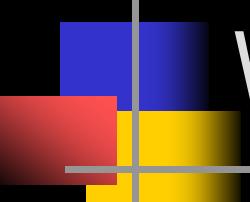
三層式發展架構的優點 (續)

- 應用程式一旦改版只要將伺服端的「首頁」更換，完全不需要重新安裝前端。
- 在 Intranet 讓企業內部的應用程式在使用上不受外界干擾，同時又可以讓企業內部的員工得以與外界的網際網路溝通。
- 兩層式主-從式架構容易轉換到此架構中，不會阻礙企業再生工程 (Re-engineering) 的進展。



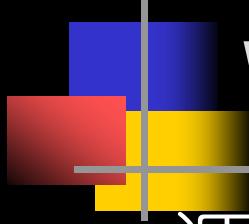
XML 技術成熟引發的網路改革

- SOA and Web Services: UDDI, SOAP, WSDL
 - SOA: Loosely-coupled, Implementation Neutral, location transparency
 - Http + RPC + XML → Web Services
- 邁向 Semantic Web/Web 2.0 的應用技術
 - Ajax (Asynchronous JavaScript and XML, Jesse James Garrett 提出))
 - RSS (Rich/RDF Site Summary : RSS 1.0, Really Simple Syndication:RSS 2.0) : 透過 RSS Reader 主動接受訊息
 - 由使用者主導的「集體創作」時代來臨 : Wikipedia, YouTube, ...



Web 2.0 的新時代已經來臨

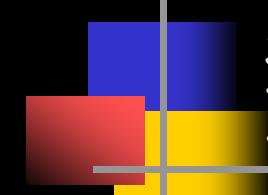
- Ajax 技術讓網站介面更豐富，互動能力大幅增強，增加群眾參與度，實現 Web 2.0 新時代，
- Web 2.0 打破過去由網路媒體主宰網站內容的單向傳播模式，朝向由使用者主導資訊傳遞與整合的「集體創作」模式，
- 由網站使用者主動提供內容與分享，共創網路新時代的知識經濟體。
- 將來的軟體可望漸漸朝向結合傳統桌面應用軟體與網際網路應用程式兩者的優點，形成所謂的 RIA (Rich Internet Applications) 概念。



Web 2.0 的特性

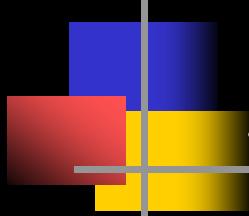
■ 源自 O'Reilly 以及 MediaLive 的會議共識：

- 網路就是作業平台 (The Web as Platform)
- 利用集體智慧 (Harnessing Collective Intelligence)
- 網站資料 (Web Content) 是下一個 Intel Inside
- 終結軟體發佈週期 (End of the Software Release Cycle)
- 輕量化程式設計模型 (Lightweight Programming Models)
- 軟體超越單一設備 (Software Above the Level of a Single Device)
- 豐富的使用者體驗 (Rich User Experiences)



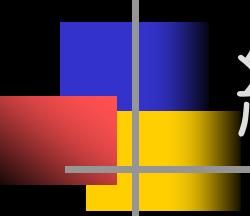
邁向 Web 2.0 所需的技術

- RSS (Really Simple Syndication)：利用 XML 標籤描述發布內容，以主動的方式通知使用者
- CMS (Content Management Systems)：用自動化方式管理由集合眾人智慧之社群所發佈的內容，著重於內容管理
- Ajax (Asynchronous JavaScript and XML)：利用合成之 Script 語言，以非同步傳輸方式與伺服器溝通 (使用者無法查覺)
- SOA (Services-Oriented Architecture)：把不同主機的應用程式以服務的方式提供給使用者，藉由網路服務 (Web Service) 等技術之整合，實現以服務為導向的系統架構。



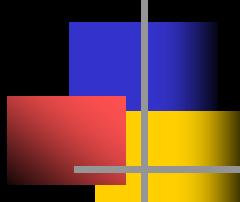
主-從架構造就的三大特性

- 「開放性」：都能夠支援眾多的作業平臺、網路通訊協定等。
- 「分散性」：伺服器彼此之間可以互相溝通，透過網路互相存取彼此的資料或交互運。
- 「整合性」：能整合不同廠商所出品的伺服器，或前端開發工具、軟體等，達成異質性交互運算的目的。



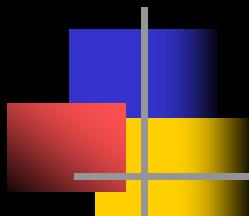
網路服務與雲端計算

- 「網路服務」：
 - 藉助 WSDL (Web Service Description/Definition Language) 進行功能描述，
 - 透過標準化的 SOAP (Simple Object Access Protocol) 技術，提供分散式應用環境上同步或非同步呼叫。
 - 使用 UDDI (Universal Description, Discovery, and Integration) 註冊標準登錄到「服務中介伺服器」(Service Broker)，讓需要服務的應用程式搜尋到，
- 目前業界朝向建構「雲端計算」(Cloud Computing) 環境
 - 將軟、硬體與儲存空間都變成服務 (SaaS, Software as a Service)，
 - 以網際網路的方式來服務所有客戶。



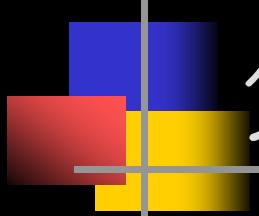
第四階段—分散式處理

- 未來的機關、企業組織、經營型態都將會跨越省、縣、市，甚至國界
- 機關、企業已將資料庫分散到網路上的各個站
- 真正反應其組織結構，並加速各資料站的處理速度與提高資料存取的可靠度
- 各個資料站具有某種自主性 (Autonomy) 但卻又要與其它資料站互相合作
- 各站間不共用記憶體，也不共用同一個計時器
- 若多個 CPU 共用記憶體，也共用同一個計時器則稱為“平行處理架構” (Parallel Processing)



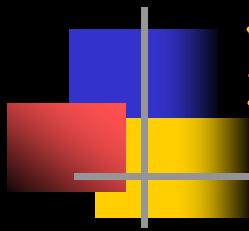
同質性分散式資料庫系統

- 使用者所存取的資料來源不再侷限於本身的資料站
- 企業以整體組織運作為主要考量，所以資料站的組成結構會有絕大部份的同質性
- Top-Down Construction
- 各資料站之間地位相等 (Peer-to-Peer, P2P)
- 可以跨越網路存取遠端的其它同質資料站
- 美國在 1980 年代便著手進行各項研究



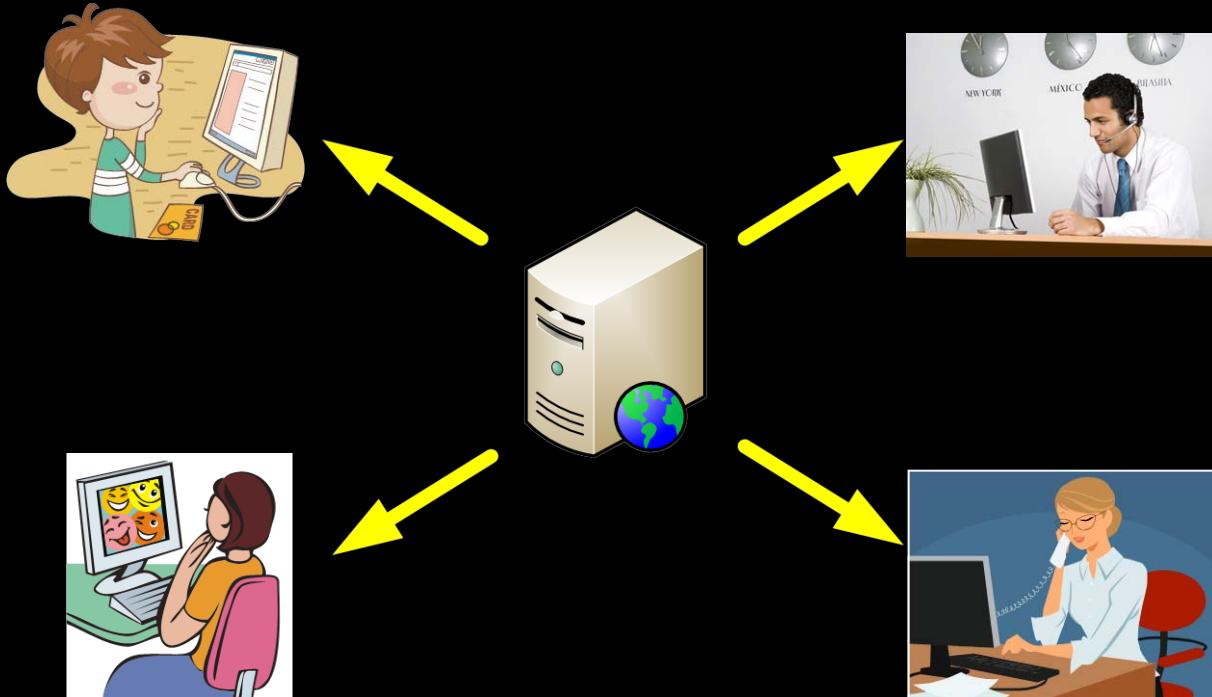
分散式架構 vs. 集中式架構

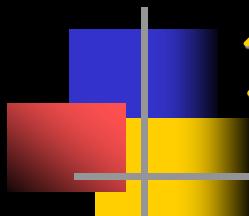
- 兩者的理念完全相反
- 所以相對於集中式系統 Centralized Systems, 分散式又稱為「反集中式系統」 De-Centralized Systems
- 集中式的優點變成了分散式的缺點
- 分散式的優點變成了集中式的缺點
- Moreover, ... (See the Next Two Pages)



集中式架構中的人機互動角色

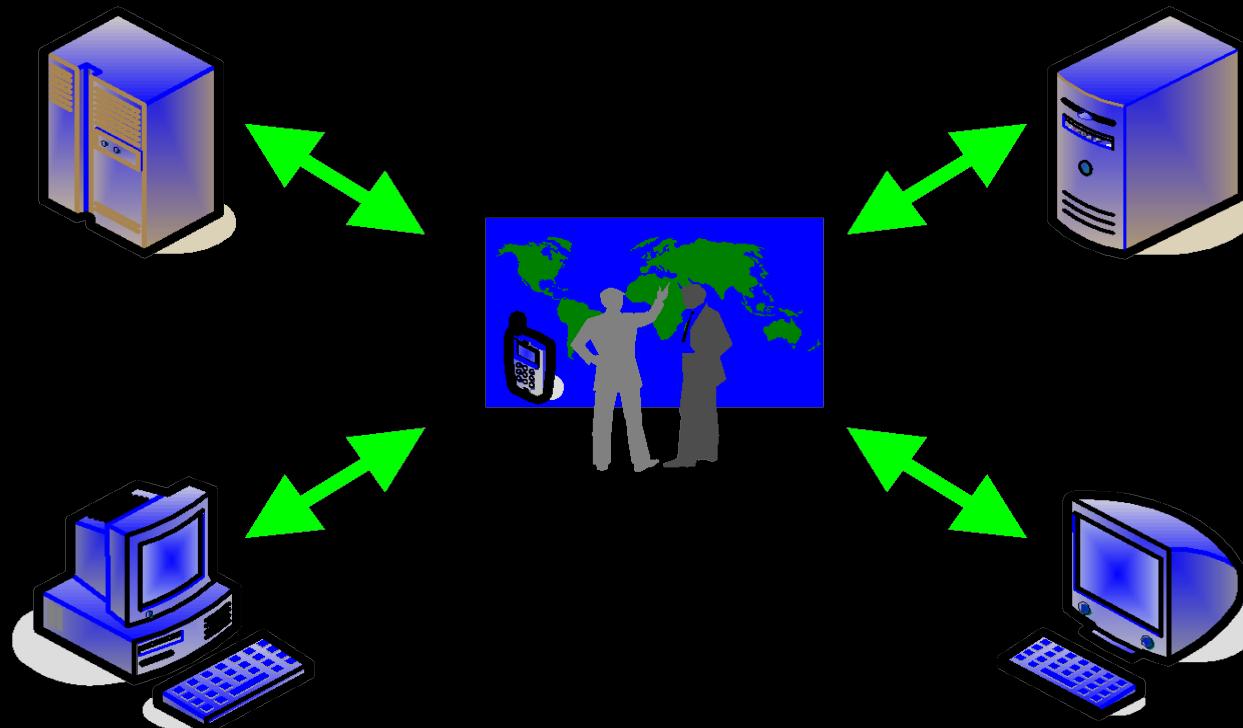
- 以電腦為中心，使用者是配角

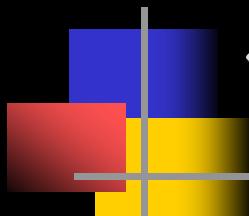




分散式架構中的人機互動角色

- 以使用者為中心，電腦變成配角

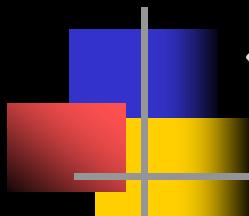




分散式架構的優、缺點

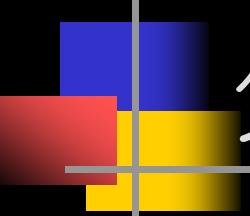
■ 優點 (與主-從式架構一樣) :

- 可增強整體的運作效能 (Performance) ,
- 提高妥善率 (Availability) 。
- 具有更多的彈性 (Flexibility) ,
- 資料站具有某程度的自主性 (Autonomy) ,
- 增進系統的穩定性 (Robustness) 與可靠度 (Reliability)



分散式架構的缺點

- 缺點 (與主-從式架構一樣)：
 - 軟體的開發價格跟著提高，，
 - 應用系統的維護不易，容易產生問題，
 - 增加網路上的處理負擔。
 - 安全性上的維護也是一項挑戰。



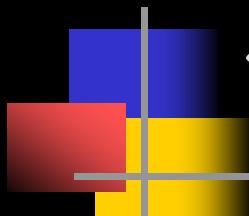
分散式架構 vs. 主-從式架構

■ 分散式架構

- 資料是分散在各部門的資料站 (Site) 上
- 各部門便可以對內部資料的容量、安全性與維護性做一番妥善的規劃，各資料站之間的地位是對等的 (Peer-to-Peer)
- 也稱為「對等式系統」(Peer-to-Peer, P2P Systems)。

■ 主-從式架構

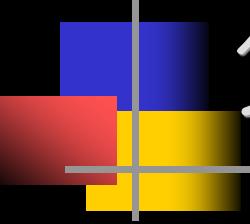
- 資料主要放在伺服端 (Server)
- 有「主」(Client，或稱 Master) 與「從」(Server，或稱 Slave) 的關係
- 稱為 Client-Server Systems 或是 Master-Slave Systems



分散式處理所引發的問題

■ 從技術的角度來看

- 除了內部處理效率以外，仍要確保跨越資料站之間的資料處理能更有效率地完成
- 資料站之間的資料轉換與整合問題
- 如何最佳化地將資料分散
- 如何在網路上對存取權限做有效的控制
- 安全而快速地確保系統回復到原來的狀態
- 跨越資料站的異動管理

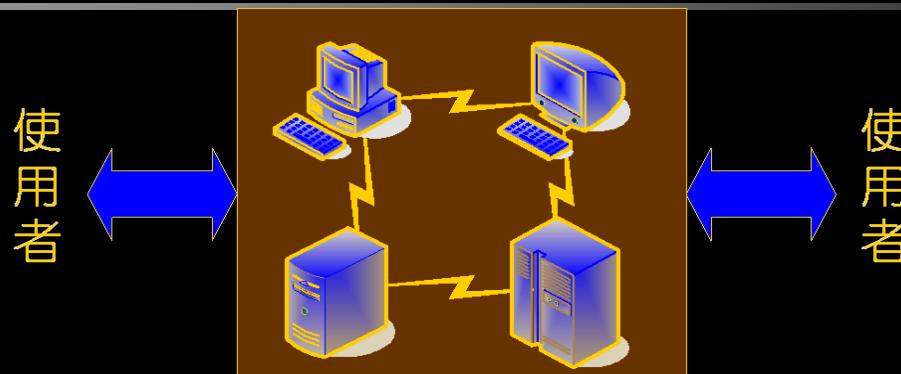


分散式處理所引發的問題

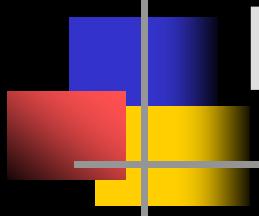
■ 從管理的角度來看

- 部門與部門間的資源管理與分配問題。
- 跨部門與部門間的資料之管理與分散問題。
- 透過網路傳輸資料後，資料安全性與保密性的管制問題
- 各部門爭取自主權的權力下放範圍

分散式資料庫系統的最大目標

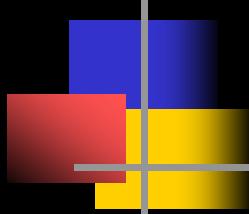


- 讓使用者用起來感覺是面對一個集中式資料庫，不需要去考慮資料存放的地點。
- 分割的透通性 (Fragmentation Transparency)
- 複製的透通性 (Replication Transparency)
- 位置的透通性 (Location Transparency)



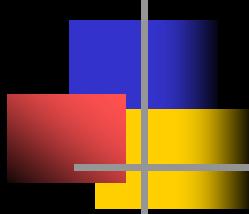
Peer-To-Peer 架構的新觀念 (註)

- 近年來「對等式計算」(Peer-to-Peer Computing, P2P Computing)已漸漸從分散式計算的概念脫離，成為一個獨立的課題
- 參與者節點散佈在各處，透過底層網路互相連接，並扮演相同的角色，
- 靠著互相合作的機制，提供資源給其他參與者，並從其他參與者獲得資源做為回報，
- 實現交換服務與資源的目的，同時提昇系統可靠度並消除瓶頸。



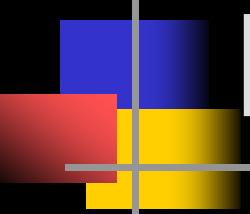
P2P 架構的興起理念

- 雖然：網際網路連結了整個世界，但是…
- 透過集中控管或事先規劃的管理架構無法讓世界各地的不同資源充分共享，
- P2P 架構運用槓桿原理的效率，捨棄伺服器的觀念，讓所有參與者均為同儕對等，
- 達成讓大多數連線於網際網路上的資源獲得充分共享的目標



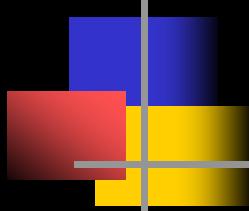
P2P 架構的特色

- 這種計算模式將對網際網路產生徹底的改變，
- 它能架在網路核心骨幹之上運作，提供更具彈性、更獨立自主的服務
- 參與伺服器之間有其獨立的定址系統與交換協定，
- 不一定要仰賴「網域名稱服務」(Domain Name Service, DNS) 或某個集中式的伺服器才能運作，
- 可以應付節點隨時加入與隨時離開的不穩定動態連線服務 (即使網路斷線或頻寬改變)，
- 功能相當強大，但是難以管轄



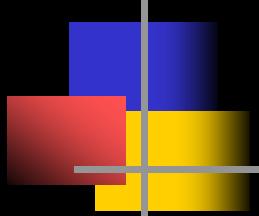
P2P 架構 (續)

- 目前盛行於特定網路用戶族群當中，大多是各種電腦檔案的交換，如: skype 網路電話，PPStream 網路電視，或透過 eDonkey、eMule、BT (BitTorrent) 等 P2P 軟體交換 mp3 或影音檔，
- 後續可能延伸成為各種資料庫的資料交換，
- 注意：開放未經授權的檔案供他人下載的行為有觸犯智慧財產權之虞。



第五階段— 異質性分散式處理

- 當代的軟、硬體均朝向：異質性、開放性與整合性來開發
- 銀行、電信、圖書等服務業，會漸漸朝向彼此整合，以及異業結盟的型態來發展，以提高其服務品質與服務對象
- 人類在溝通、交流演進下的產物：電話系統→Internet 電腦網路系統→異質性資料庫系統



異質性資料庫系統

- 為何會走向異質性系統的整合？
- 同業購併也必須走向資料整合的路線
- 不同組織整合後所面臨的是：
 - 獨立組織的內部運作條件、型態所形成的差異，
 - 使得整合所要面對的問題是異質性的 (Heterogeneous)，
 - 故這類資料庫的整合稱為異質性資料庫的整合。

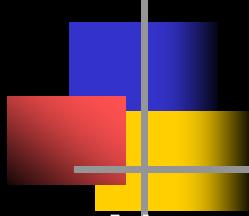
同質性 vs. 異質性 分散式資料庫系統

■ 同質性分散式資料庫系統

- 單一機關或企業自行建立
- 架構的設計乃是由上至下 (Top-Down)
- 各個資料站需要分工合作時，便可以捨棄某些自身的自主性
- 可以充分合作達成最佳化的目標
- 通常是資料的整合非系統的整合

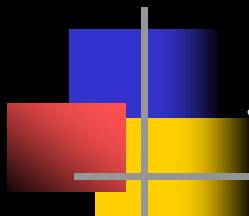
同質性 vs. 異質性 分散式資料庫系統 (續)

- 異質性資料庫系統
 - 整合不同機構的資料庫
 - 由下至上 (Bottom-Up) 整合
 - 各個機關仍有其完全獨立的自主性
 - 不傷害內部利益的情況下，酌量開放某些資訊供異質性資料庫系統來整合
 - 難以達成最佳化的運作
 - 可以是資料的整合或系統的整合



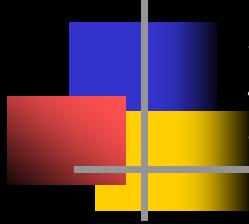
Internet 電腦網路系統

- Network as a Computer → Internet as a Computer → Software/Platform as a Service (SaaS/PaaS, On-Demand Software, **Cloud Computing**):
 - IBM: Blue Computing
 - Amazon: EC2 (Elastic Compute Cloud)
- 分散式計算環境可能需要在作業系統層次透過「中介軟體」(Middleware) 來做為整合的介面
- 異質性資料庫系統則是在 Database System Level 上運作



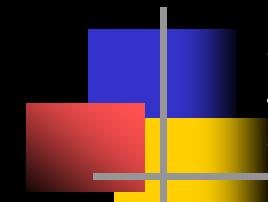
異質性分散式資料庫系統

- Network As a Global Database.
- 整合原先既存的各種資料庫管理系統
- Buttom-Up Construction (vs. Top-Down Construction)
- 必需解決不同程度的「異質性」(Heterogeneity)
- 透過網路與各類系統做彼此間的交互運作
- Loosely-Coupled (Interoperability) vs. Tightly-Coupled (Heterogeneous Integration)



整合觀點 1 (資料 vs. 系統整合)

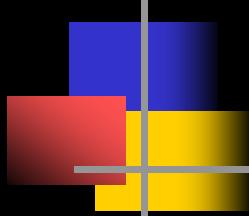
- 「資料整合」與「系統整合」
 - 資料整合：以緊密的結合方式將所有資料融合在一起，稱為「緊密整合」(Tightly-Coupled)。
 - 系統整合：僅做系統介面上的整合，以形成為所謂的「交互運作系統」(Inter-operable Systems)，我們稱此種整合方式為「鬆散式的整合方式」(Loosely-Coupled)



整合觀點 2 (縱向 vs. 橫向整合)

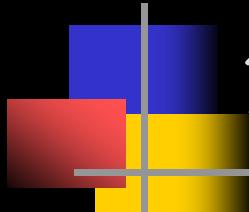
■ 「縱向整合」與「橫向整合」

- 縱向整合 (Vertical Integration)：將許多資料庫進行垂直整合，並以緊密結合方式將資料融合在一起，提供整體觀點，也就是「分散式資料庫系統」的主要概念。
- 橫向整合 (Horizontal Integration)：透過商業程序 (Business Process) 將不同資料庫中的資料串接起來，完成企業內部或跨組織之間的作業程序，形成所謂的「工作流程」 (Workflow)。整個程序中可能會牽涉到：使用者 (Users)、活動 (Activities)、程式 (Programs)、以及資料 (Data) 等概念。



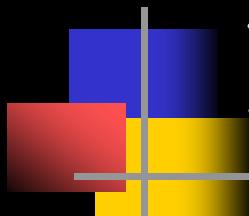
格網計算 (Grid Computing): 另類的異質性分散式計算模式

- 格網計算 (Grid Computing), 或稱「網格計算」
- 透過強力的協調運作，讓大量異質性計算資源得以共享，並以動態加入與重組的方式解決各種需要大量計算資源的科學與商業問題，
- 目前朝標準化制定邁進，讓尋找計算單元、資料的存取、資源的配置與監督等工作，能夠透過網際網路 (Internet) 在單一的虛擬環境下完成，
- 希望其效能足以媲美，甚至於超越超級電腦 (Super Computer) 的能力。E.g., 尋找外星生物研究計畫 SETI@Home



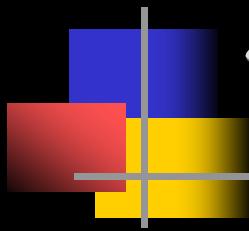
分散式計算的整合式目標

- 希望讓各類型的計算裝置 (Computing Device)，如：
 - 個人數位助理 (Personal Digital Assistant, PDA)、
 - 汽車上的電腦設備、
 - 配有電腦計算功能的先進家庭電器設備 (Appliance)、
 - 行動電話 (Cellular Phone)、甚至
 - 「自動射頻辨識標籤」 (Radio Frequency Identification, RFID) 的
讀寫機 (RFID Reader) 等
- 都能夠結合下一節的「行動計算處理」讓計算能力無所不在，
就像現在隨處可以取得的電力一般得以隨插即用，
- 衍生出了所謂的「普及式計算」 (Pervasive Computing)，以及前述的「對等式計算」 (P2P Computing) 這些新觀念與之相互結合



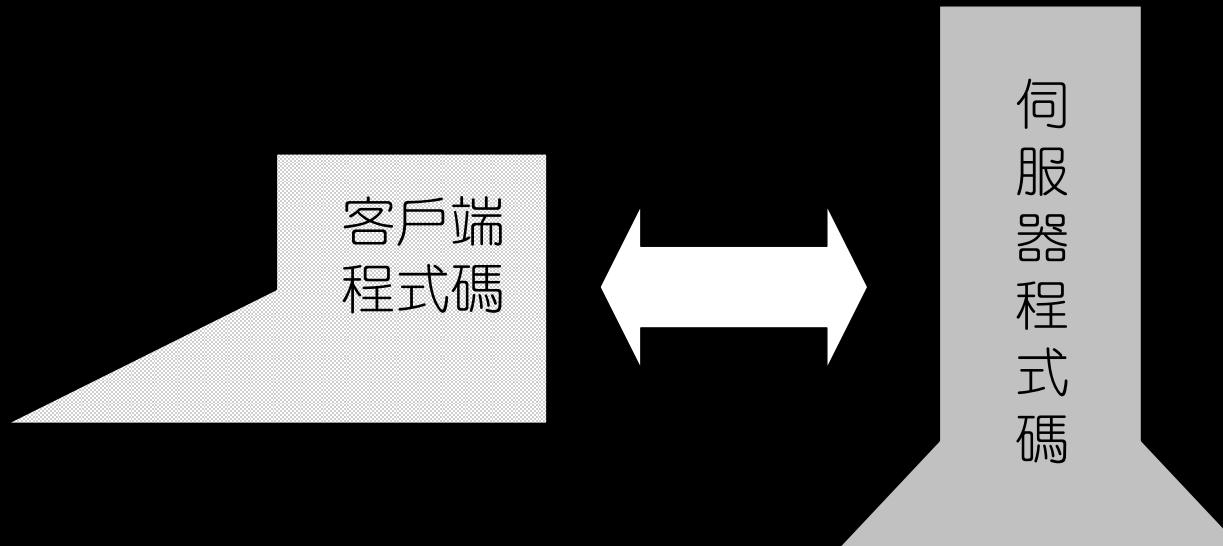
第六階段—行動計算

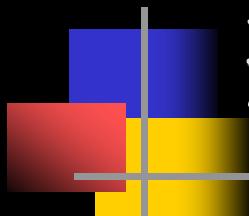
- 攜帶型電腦與無線電腦通訊的發展促成
- 「行動計算」(Mobile Computing) 又稱為「遊牧計算」(Nomadic Computing)
- 新一代的電腦作業系統中，已經開始加入了支援行動計算的各種特性，如：Windows XP
- 行動計算是否會引發更多有關資料庫方面的理論與實務研究課題？
- GRPS, 3G, IEEE 802.11b (802.11g), 802.16...



傳統的主從式運算環境

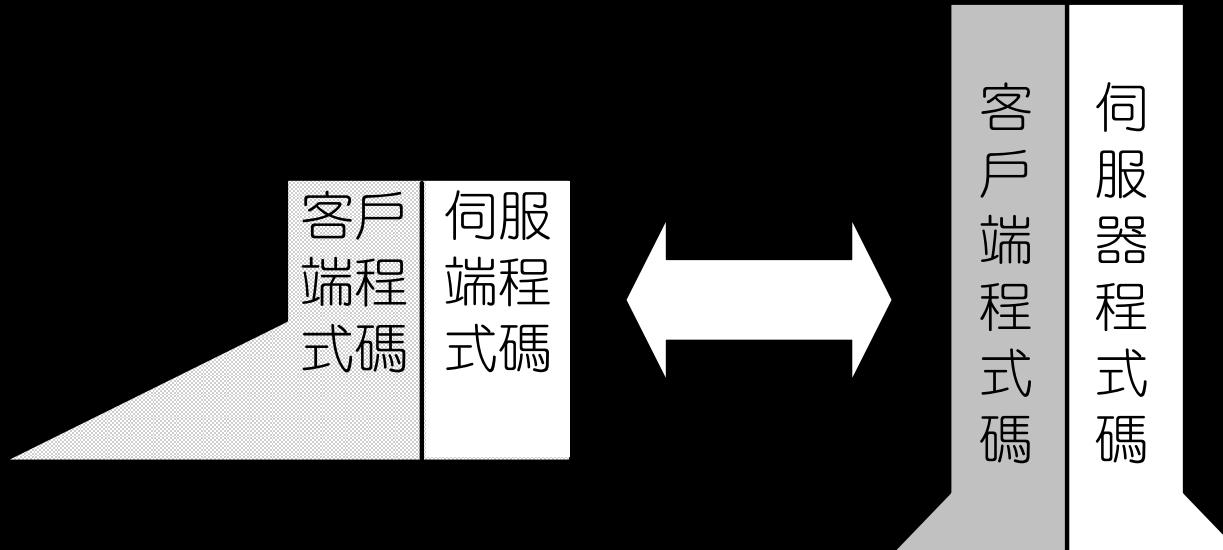
- 傳統的攜帶型電腦只能當客戶端 (Client)





網路傳輸量以驚人速度成長

- 將客戶端的電腦作為運算處理的主體，而伺服器只是做為提供資料的配角
- 某些網路的節點上已經沒有客戶、伺服器之分

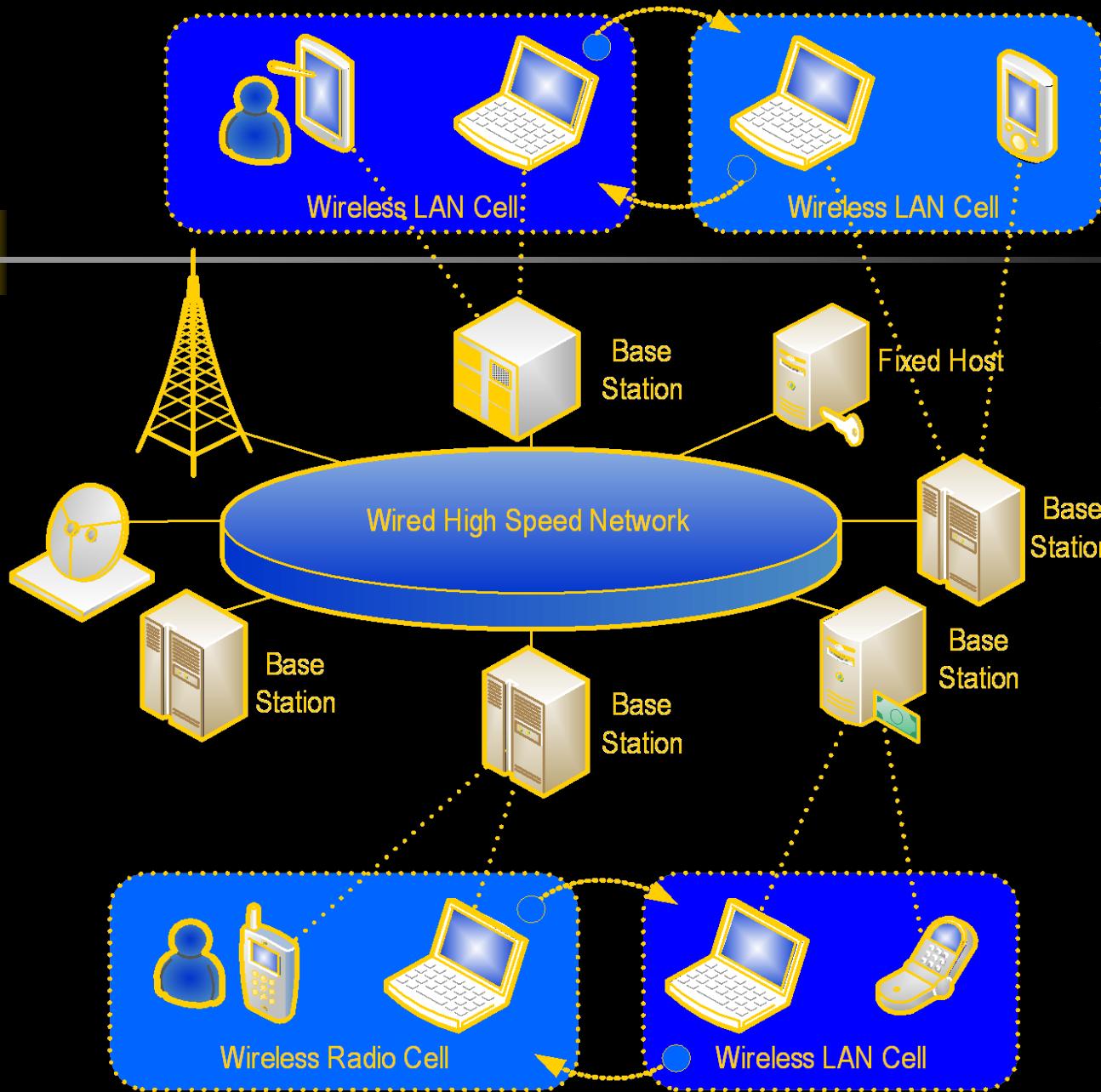


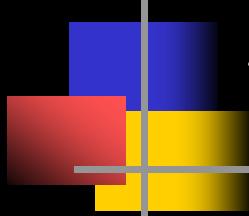
行動計算的環境裡 攜帶型電腦的雙重角色

- 變動的資料站：可透過電話線或經由其它媒介與其它電腦相連
- 固定資料站：直接與主電腦固定連接
- 無論是那一種角色，它都是分散式計算環境中的一份子

行動計算的架構圖

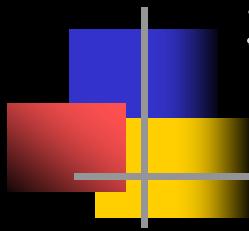
Wi-Fi/3G/
WiMax





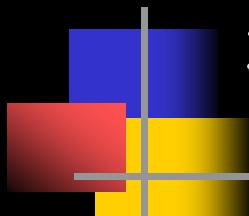
行動計算 (續)

- 固定主機 (Fixed Host)：固定的中、大型主機
- 基底站臺 (Base Station)：又稱為「行動支援基底站臺」 (Mobile Support Stations): Wi-Fi (IEEE 802.11系列) → WiMax (IEEE 802.16 系列，號稱可以涵蓋 40~50 km)
- 行動計算單元 (Mobile Cell)：攜帶型電腦、掌上型電腦或 PDA
- 分散式系統看成是行動計算的一個特殊情況
- 在行動計算的環境上，其網路連線是動態變化



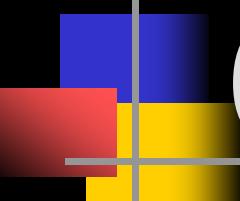
資料庫管理系統所受到的影響

- 節省行動計算單元電力的考量
- 查詢最佳化處理
- 異動管理 (Transaction Management) 模式也要做適當的調整
- 使用者介面的設計
- 資料正確性、穩定性與安全性的課題
- 網路溝通上的不確定性 (MQ-Series)



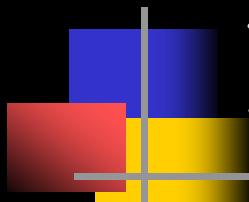
行動計算中的幾種新查詢觀念

- 不要求百分之百的「近似查詢結果」
(Approximate Query Answering)
- 「概略性查詢結果」(Intensional Query Answering)
vs. Extensional Query Answering (具體性查詢結果)
- 「與位置相關的查詢」(Location-Dependent Queries)
- 「最佳停止點之查詢結果」(Optimal Stop Query Answering)



行動計算 vs. 普及式計算 (Pervasive Computing) 的未來

- 要求動態的連接方式，動態地選擇各項功能、資料的傳輸
- 客戶端也有主動權可以主動調整本身的狀態、伺服端的各項功能以及資料的傳輸方式等。
- 硬體技術進步，造就行動計算裝置越來越小型化，
- 透過行動計算模式整合後形成了「普及式計算」(Pervasive Computing)的概念。
- 將來，當我們打開冰箱時，可能牛奶瓶上的 RFID 標籤已取代條碼 (Barcode)...
- 有能力發射無線射頻電波到使用者身上的行動電話 (或電子手錶)，告知其有效期限的日期...
- 若快過期，則顯示發射來源於電話螢幕上，並發出「嗶」聲以警示使用者儘速飲用。



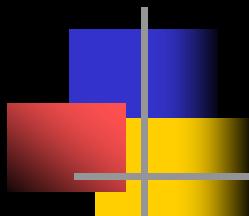
第七階段—雲端計算

■ 雲端計算的興起：

- 各種計算的理論發展或產品研發，已趨於完備，
- 網路速度大幅提升，因此漸漸興起了雲端計算 (Cloud Computing) 的概念。

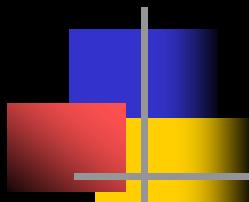
■ 雲端計算 (Cloud computing) 的重點

- 「雲」(Cloud)：指電腦網路，或網際網路 (Internet)
工程師常以一朵雲來代表「Internet 網路」
- 「端」(Client)：筆記型電腦、手機、平板電腦等
「終端裝置」(Terminal Devices)



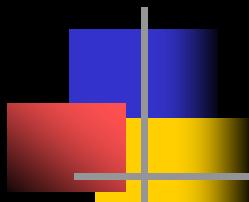
雲端計算簡介

- 「雲端計算」：應用「分散式架構」虛擬化伺服務器設備和平台，以降低使用成本。
- 各類裝置連上網際網路以後，透過遠端伺服器或資料中心 (Data Center)，便可以快速存取資料或執行各種計算服務
 - 如：資料搜尋與檢索、觀賞影音、玩電腦遊戲等，
- 未來電腦、手機等終端產品將走向精簡型，只要能連上網際網路便可滿足絕大部份的需求。



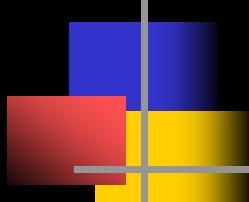
雲端計算簡介

- 「雲端計算」並不是新技術，而是一種新概念，
- 目的是利用電腦網路使電腦彼此的合作或所衍生的服務更加無遠弗屆，
- 運用前述的六大架構，以及已經發展完備的技術來進行改良，
- 在實現「雲端計算」概念的過程中，產生相對應的「雲端技術」與「雲端服務」



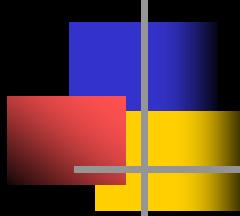
雲端技術 (Cloud Technology)

- 研究重點：應用資訊科技 (Information Technology) 提升雲端服務的效能、穩定性與安全性，並降低成本。以提供
 - 動態、虛擬化、易擴充的套件，
- 讓使用者不需要了解「雲端」基礎設施的細節，
 - 不必具有對應的電腦系統專業知識，
 - 更不需要直接對伺服器進行控制，
 - 只要關注自己真正需要什麼樣的資源，以及如何透過網路來得到相應的服務即可。



雲端服務 (Cloud Service)

- 重點在於服務的創新，而非資訊科技本身。探討在 3P 上面的 3C 服務。3P是指部署模式：
 - 特定雲 (Peer Cloud)：由特定族群所架設的雲端計算網路，或是某供應鏈的上、下游廠商們自行建立的服務管道，又稱「社群雲」(Community Cloud)
 - 私有雲 (Private Cloud)：指企業內部因為基於資料保密因素所自行架設的雲端服務環境；
 - 公眾雲 (Public Cloud)：政府、法人團體或企業 (像：Google、Yahoo 或 Facebook) 為了服務大眾所建立的雲端服務環境
 - 以上部署模式也可以混合後，形成「混合雲」(Hybrid Cloud)



雲端服務 (Cloud Service)

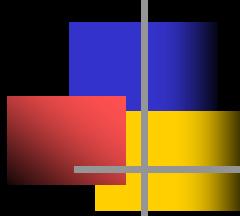
■ 3C 是指：

- 計算服務 (Computing)：提供運算環境平台 (Platform as a Service—PaaS)，如：VMWare；在網路上提供軟體應用的租賃服務等 (Software as a Service—SaaS)，例如：E-mail、Google Docs、搜尋引擎服務等。
- 儲存空間服務 (Capacity)：像 Dropbox 或 Database as a Service—DaaS 等。
- 內容服務 (Content)：像網路電子書、Youtube 或 E-Learning、趨勢雲端掃毒服務，不需在個人電腦上更新病毒碼等。



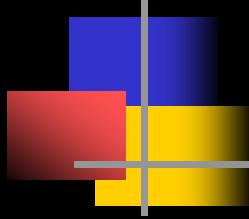
服務科學 (SSME) 的出現

- 透過以上雲端核心服務，後續可以衍生出各種類型的網路服務商業模式
 - 結合實體服務模式後可形成**創新服務型態**。
 - 配合**實體或虛擬服務人員與服務制度**，未來可望形成一個蓬勃發展而且創新的服務新局面，
 - 讓「資訊系統」漸漸變成「服務系統」的概念；
 - 讓各行各業都漸漸演變成為某種型式的服務業。
 - IBM 力圖轉型的目標，其重點在於整合與創新，並引出了新興的研究議題：「**服務科學**」(SSME—Service Science, Management, and Engineering)。



雲端計算總結

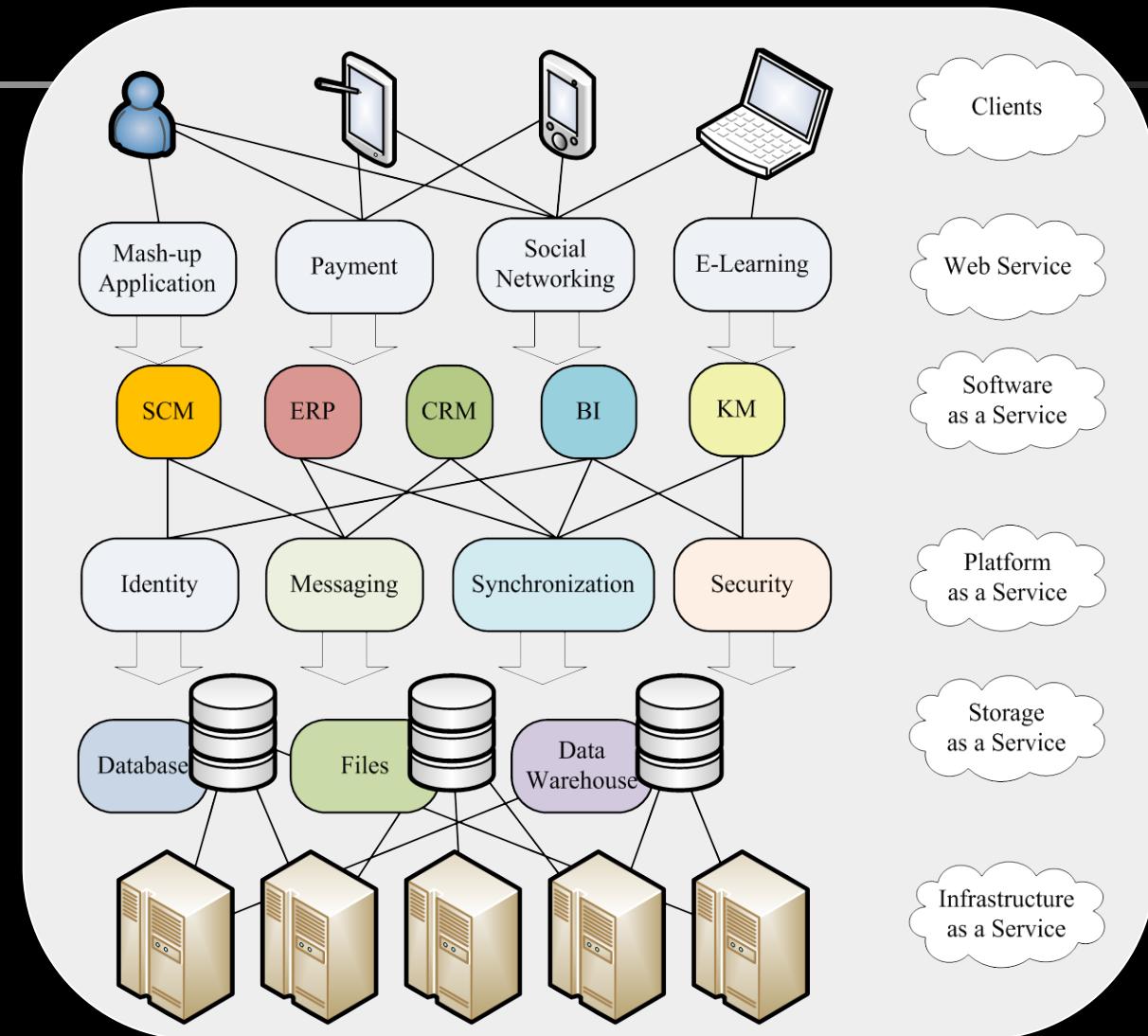
- 網際網路與應用服務未來就像水、電一般隨插即用那麼方便，
- 讓我們時時仰賴電腦，但是卻忘了它的存在
 - 個人的計算能力：筆、個人電腦，延伸到雲端；
 - 個人的記憶庫將從自己的腦袋、記事本，延伸到雲端
 - 企業的服務能力也將從第一線面對客戶的員工，搭配後勤支援作業，一直延伸到與雲端整合。

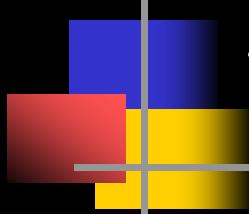


雲端計算的特色

- 超大規模 (High Scalability) 、
- 具通用性 (Universal) 、
- 虛擬化 (Virtualization) 、
- 高可靠度 (High Reliability) 、
- 高可用性 (High Availability) 、
- 成本低廉 (Low Cost) 以及
- 用多少付多少 (Pay-As-You-Go) 等。

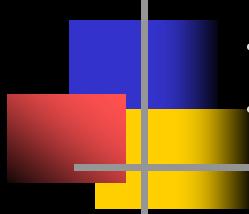
雲端服務的範圍





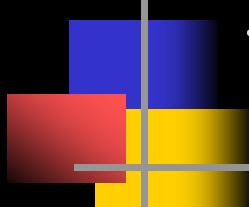
資料分析模式的演進

- 報表系統 (Reporting Systems) ,
- 試算表 (Spreadsheets) ,
- 決策支援系統 (Decision Support Systems) ,
- 資料倉儲系統 (Data Warehouse Systems) ,
- 資料採礦 (Data Mining) ,
- 文件倉儲系統 (Document Warehouse Systems) ,
- 文件採擷 (Text Mining) ,
- 知識管理系統 (Knowledge Management Systems)—可以歸納、管理與分享「商業智慧」 (Business Intelligence)



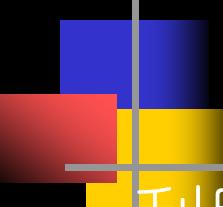
報表系統

- 缺乏分析方法與工具，
- 大多是將平時業務所需的各類資訊印成報表以供檢討。
- 報表的型式繁多，往往一變再變，造成大量的系統維護工作都是用在報表的格式修正與調整上，
- 效率低落，而且所能提供的分析結果相當有限



試算表系統

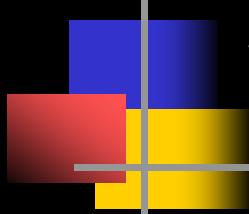
- 試算表提供了靈活的報表組織能力，在 Lotus 1-2-3 與 Excel 出現之後，幾乎完全取代了以往的報表系統，甚至於成為個人資料整理的最佳工具。可惜的是，這類系統所能應付的資料量相當有限。
- 以 Excel 來說，在 32 位元的作業系統下，通常資料筆數如果超過 65535 筆時就無法進行處理，大大地限制了它們在企業營運分析上的應用範圍。



決策支援系統 (DSS)

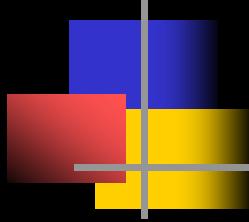
利用電腦分析資料的能力，協助管理者針對

- 一組事先定義的決策敘述 (Decision Statement) ，
- 提供各種選擇方案 (Alternatives) 的組合分析，並
- 依據「決策制定準則」 (Decision Making Criteria) 予以最佳化 (Optimize) ，
- 以輔助人類在面對半結構化 (Semi-Structured) 或非結構化 (Unstructured) 的問題時，做最終的決策
- 配合各種不同的決策模式 (Decision Model) ，以交談式的介面輔助管理者制定決策
- 主要工作算是知識的前置處理 (Knowledge Pre-processing)



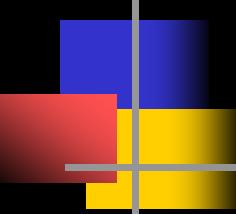
決策支援系統 (續)

- 決策 (Decision) 是從一組選擇方案 (Alternatives) 中，進行理性判斷後選擇其一
- 決策會用三個部分來描述其特徵：
 - 對決策的描述 (Decision Statement)、
 - 一組可行的選擇方案 (Alternatives)，以及
 - 一組決策制定準則 (Decision Making Criteria)
- 最簡單形式的 “決策” 是 (Yes/No) Question!
- 另有所謂的「群體決策支援系統」(Group Decision Support System, GDSS)：提供群體決策的參考。



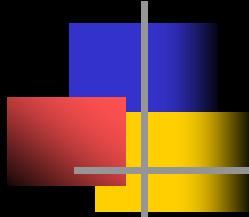
制定決策的三個階段

- 「決策」會歷經三個階段 [H. Simon (1960)]
 - 理解階段 (Intelligence Phase)：發現 (Find)、確認 (Identify)，以及有系統地規劃 (Formulate) 所要決定的問題。
 - 設計階段 (Design Phase)：設計與開發各種可行的替代方案 (Alternatives) 組合。
 - 選擇階段 (Choice Phase)：評估各種替代方案的優劣程度，並做出最後的決策。
- 三個階段並不是可以一氣呵成的，有時候可能因為資訊不足而會有回到前一個階段，甚至於從頭來的情況發生。



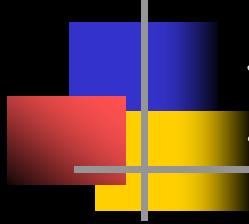
決策的分類

- 「決策」依資訊齊全與否可分成以下三大類: [Gorry and Morton (1971)]
 - 結構化的決策 (Structured Decisions) : 前述三個階段的Input、Output 與 Internal Procedure 都可以明確定義出來時 (此種決策可以完全以電腦程式來實現並解決)。
 - 半結構化的決策 (Semi-Structured Decisions) : 當某些Input、Output 與 Internal Procedure 無法明確定義出來時 (可以使用電腦程式來加以輔助，企業面臨的大多屬於此類決策)。
 - 非結構化的決策 (Unstructured Decisions) : 當所有階段的Input、Output 與 Internal Procedure 都無法明確定義出來時



「決策」應用層面的分類

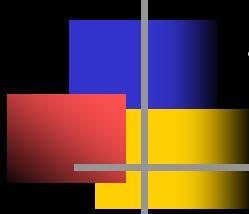
- **策略型決策** (Strategic Decisions)：從整個組織的目標與政策面出發，影響層面擴及全組織，影響時間長。由高層主管來制定之。
- **戰術型決策** (Tactical Decisions)：從整個組織的某個部分出發，要在策略型決策的規範中，影響層面僅為少數部門或人員，影響時間較短。由中階的主管來制定之。
- **操作型決策** (Operational Decisions)：從組織的某個特定活動出發，影響層面小，時間也最短。由基層的工作人員來直接實現。



決策的九大分類 [Gorry & Morton]

應用範疇 結構	操作型 (Operational)	戰術型 (Tactical)	策略型 (Strategic)
結構化 (Structured)	1	2	3
半結構化 (Semi-Structured)	4	5	6
非結構化 (Unstructured)	7	8	9

通常企業面對的決策問題大多是落在 1、5、9 這幾種類型中



資料倉儲系統

- 決策品質完全是取決於「資料品質」(Quality of Data) — 「垃圾進，垃圾出」(Garbage In, Garbage Out)。
- 資料以多維度呈現能讓使用者獲得更多資訊。
 - 綜合了這兩項需求，造就了「資料倉儲」(Data Warehouses) 的出現，
- 資料必須要經過品管，也就是要經過淨化、修補與增強後才能進到資料倉儲中。

資料以多維度呈現

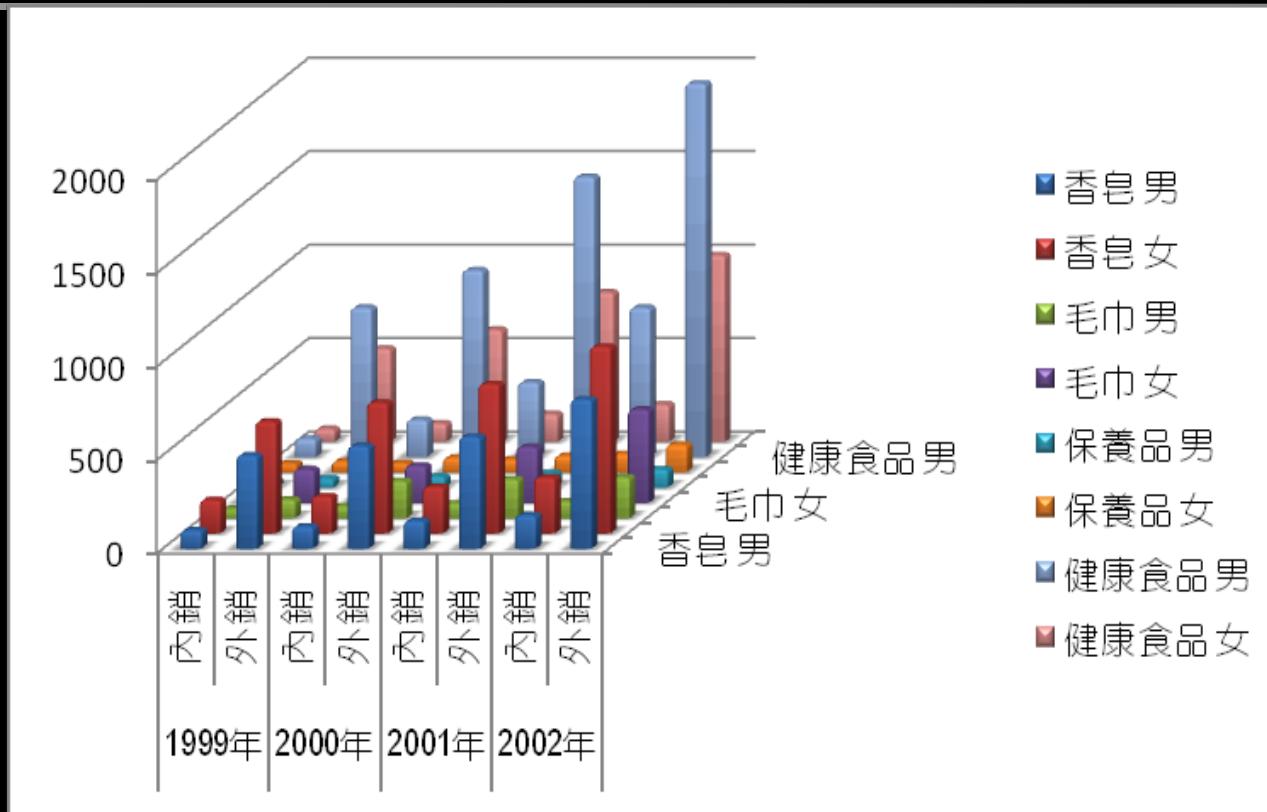
維度 1
維度 2

維度 3
維度 4

以 4 個維度看第 5 個維度 (銷售量)

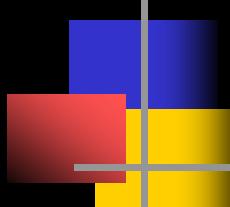
年度/銷別		1999 年		2000 年		2001 年		2002 年	
		內銷	外銷	內銷	外銷	內銷	外銷	內銷	外銷
產品/性別	男	100	500	120	550	150	600	180	800
	女	180	600	200	700	250	800	300	1000
毛巾	男	50	100	65	200	80	210	90	220
	女	80	180	90	200	100	300	150	500
保養品	男	38	50	45	60	50	80	70	100
	女	48	60	50	80	70	90	100	150
健康食品	男	100	800	200	1000	400	1500	800	2000
	女	70	500	100	600	150	800	200	1000

多維度配合統計圖表一目了然



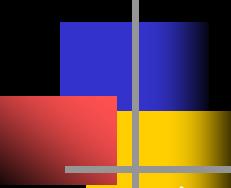
應用廣泛: 例如, 想知道職棒對社會有無貢獻?

調查顯示: 日本職棒球季期間，整體犯罪率降低兩成



資料倉儲系統

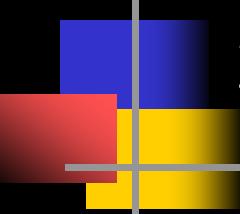
- 決策品質取決於「資料品質」(Quality of Data)的好壞。
「垃圾進，垃圾出」(Garbage In, Garbage Out)。
- 綜合了這兩項需求，造就了「資料倉儲」(Data Warehouses)的出現，
- 建置過程要求內存的資料必須要經過品管，也就是要經過淨化、修補與增強後才能進到資料倉儲中，
- 以多維度的結果呈現
- 目前在資料倉儲領域影響力最大的兩人：
 - W.H. Inmon (提出整體架構)
 - R. Kimball (實務經驗豐富)



資料倉儲的四個要點

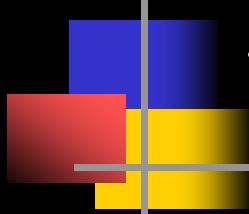
■ 資料倉儲之父 W.H. Inmon (1994) 定義：

- 整合的 (Integrated)：集中而統合的資料庫
- 主題導向 (Subject-Oriented)：傳統資料庫的規劃大多以功能來劃分 (Function-Oriented)，整合後的資料則以主題 (Subject) 來區分。
- 隨時間變動的 (Time-Variant)：資料倉儲儲存整個組織的運作歷程記錄，
- 非暫存性 (Non-Volatile)：保存每個階段當時的狀態，所以資料一旦存入資料倉儲即被保留



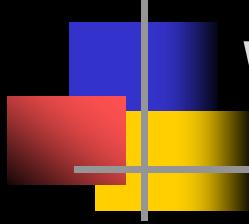
從實務觀點看資料倉儲

- 建立資料倉儲的實務考量 [R. Kimball (1996)]:
 - 存取介面必須立即、可以隨時取用，而且要具備高效率的存取能力。
 - 具備一致性而且不可以有不確定或錯誤的資料夾雜其中。
 - 讓我們根據不同的維度對不同的觀察值做組合或拆散的動作。
 - 要包含查詢、分析與展示的工具。
 - 存入前必須要有正式編制人員嚴格把關，不容許錯誤的情況發生。
 - 要求更完備的資料庫系統，以及更佳的管理品質，會讓企業具備全面品質提升的動力

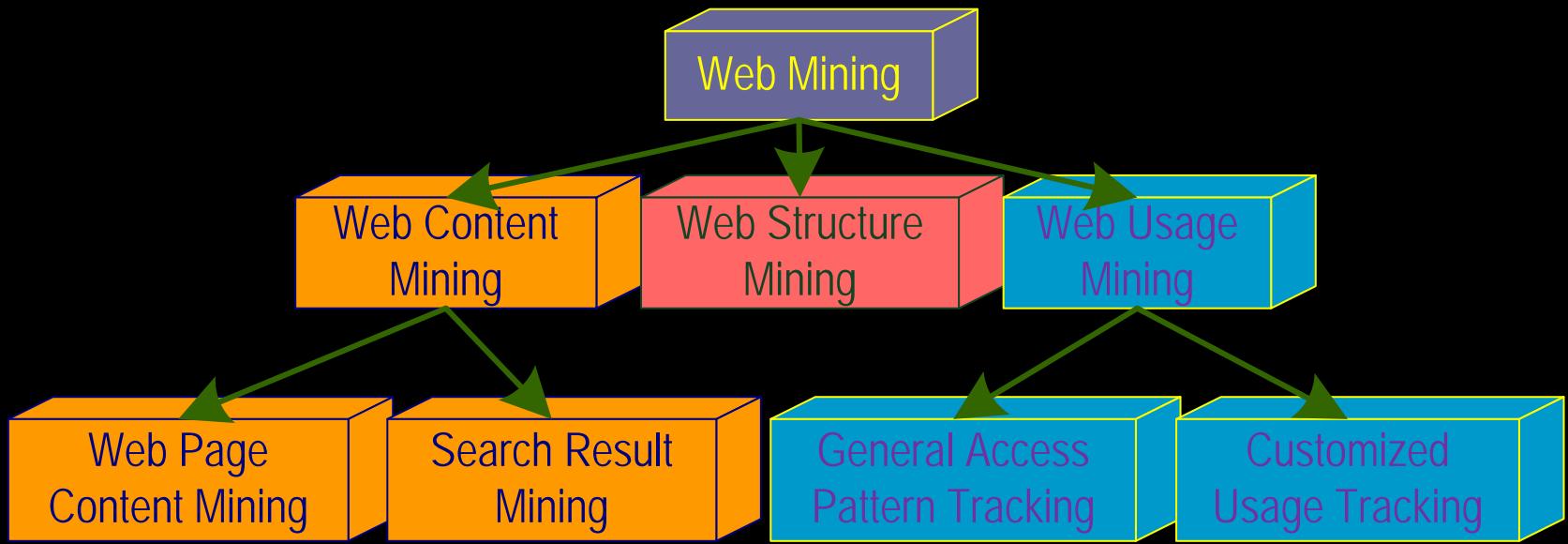


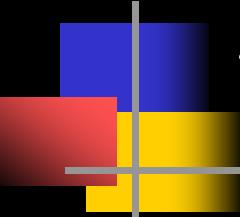
資料採礦 (Data Mining)

- 從雜亂的資料當中整理出某些頭緒，以得到以往所無法觀察得知的現象
- 結合了資料庫系統、人工智慧、統計學、高效能運算架構與視覺化等技術所產生的新興領域？
- Web Mining 利用此一技術來了解
 - 網站內容的特徵 (Web Content Mining)
 - 網站結構的特徵 (Web Structure Mining)
 - 使用者的行為模式 (Web Usage Mining)



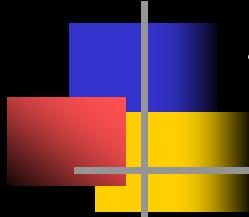
Web Mining 的分類





資料採礦技術的分類

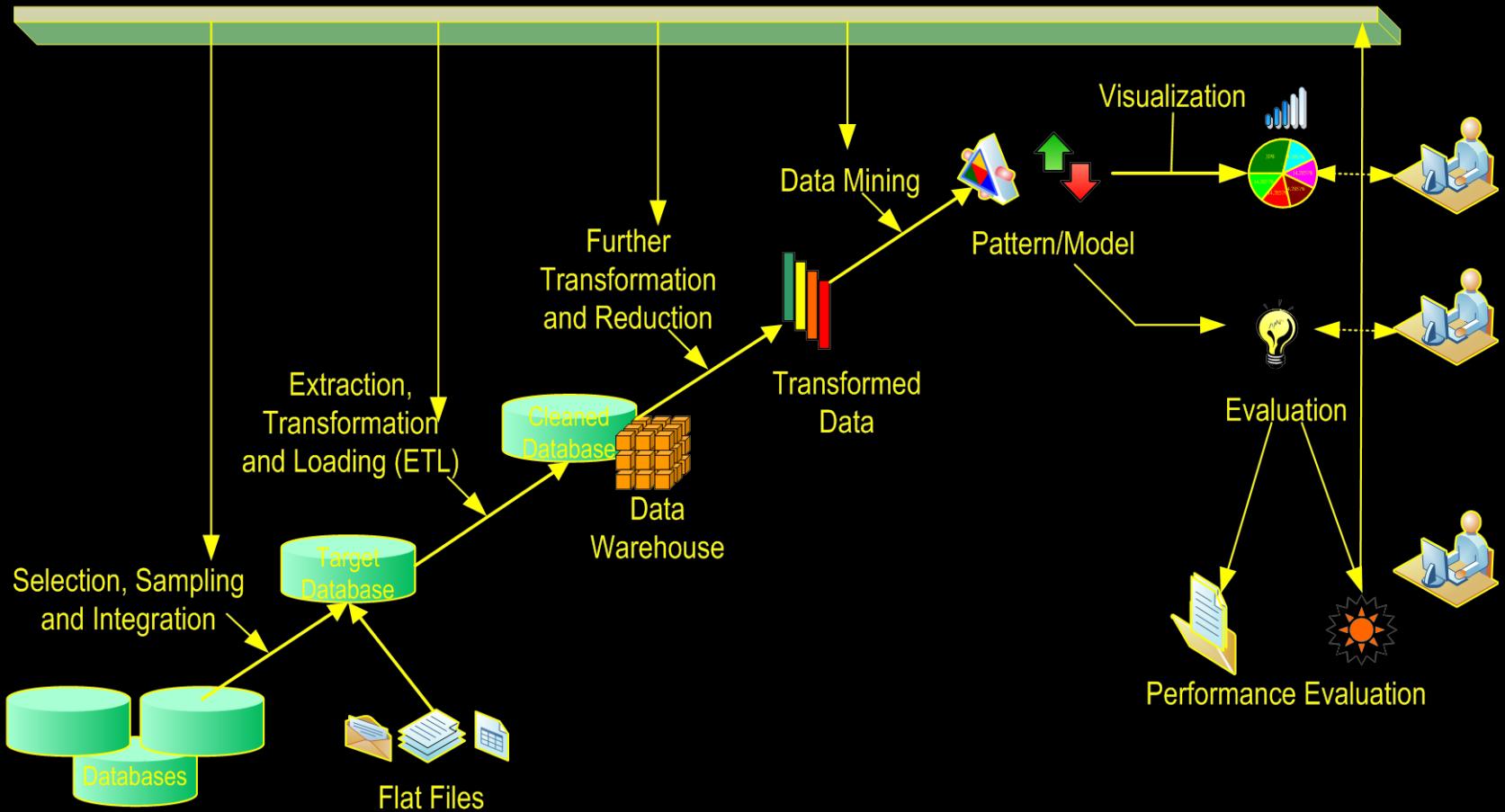
- 找出分類的規則 (Classification Rules)
- 找出關聯性規則 (Association Rules): 啤酒 vs. 尿布
- 找出順序規則 (Sequence Rules): 阿諾電影→湯姆漢克?
貝克漢進球數→全球小麥價格 (球迷啤酒喝越多, 小麥漲價!)
(蝴蝶效應 (The butterfly Effect): "Does the flap of a butterfly's wings in Brazil set off a tornado in Texas?"
by Edwards N. Lorenz (1979))
- 找出同質時間序列 (Similar Time Series)
- 找出群集規則 (Clustering Rules)
- 完成連結分析 (Linkage Analysis)
- ...

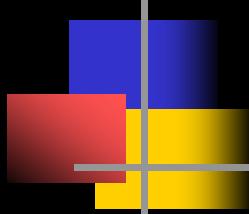


資料採礦的步驟（續）

- 最早由 U.M. Fayyad 在 (1996) *IEEE Expert Intelligent Systems & Their Applications* 中描繪出大綱
- 後續有些學者則認為在做資料採礦時應該先整合各種資料並架在資料倉儲上來實行
- 完整的步驟稱為 KDD (Knowledge Discovery in Databases), 其中 Data Mining 是其中的一項步驟，說明如下頁所示。

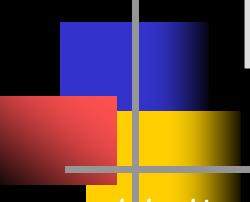
KDD 的步驟 [U.M. Fayyad (1996)]





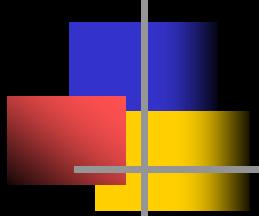
KDD 的步驟（續）

- 針對應用領域，瞭解該領域的先導知識，及最終目標。
- 先將資料庫與檔案資料整合，並透過資料篩選、取樣以建立一個採擷目標資料集。
- 將資料做前置處理，去除錯誤或不一致的資料 (Data Cleaning and Preprocessing)。此步驟有時候可以與上一個步驟互換順序。



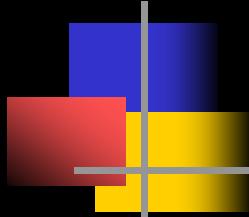
KDD 的步驟（續）

- 簡化/轉換資料 (Data Reduction & Transformation)
- 決定資料採礦的工作：
 - 找出分類規則 (Classification Rules)、關聯性規則 (Association Rules)、順序規則 (Sequence Rules)、同質時間序列 (Similar Time Series)，還是群集規則 (Clustering Rules)。
- 根據上述決定選擇演算法或處理模式
- 尋找「樣型」 (Pattern)、做迴歸分析 (Regression) 或找出分類型態等。然後透過視覺化工具軟體 (Visualization Tools) 展現分析結果。
- 評估上述結果是否有用並符合效率 (需專家參與)。
- 這些步驟已經有標準化的規範了：稱為 CRISP-DM (<http://www.crisp-dm.org>) 是由 NCR, SPSS, 等廠商所推動



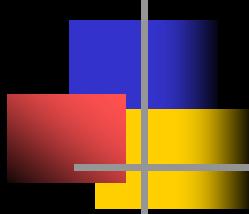
應用層面

- 超市客戶分析 (動線設計、貨架、音樂, ...)
- 美國 NBA、職棒對手與戰況分析
- 股市、期貨分析
- 痘情上的醫療分析
- 科學與工程上的分析
- 保險與信用卡市場，洗錢與逃、漏稅防制
- 電信資料的商機開發 (e.g. 與航空公司結盟)



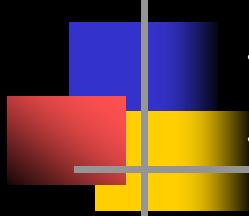
文件倉儲系統

- Survey.com 的分析結果顯示：
 - 其實企業所需要的商業智慧大約只有 20% 是由存放在傳統關聯式資料庫中的結構化資料所推導出來的。
 - 其餘 80% 左右的商業智慧必須要到各式各樣的商業文件中去找尋
- 例如：市場調查報告、專案進度報告、會議記錄、客戶的抱怨信件、專利申請書、競爭對手的廣告內容等，都是以文件形式儲存。



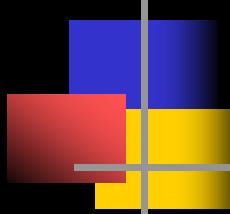
文件倉儲系統

- 自動萃取文件特徵內容 (Key Feature Extraction)、自動做文件分類 (Automatic Document Categorization)、自動做文件內容總結與歸納 (Automatic Text Summarization) 的操作環境
- 資料倉儲的建置僅能協助企業找出結構化資料中的商業智慧，並協助決策人員了解某些營運現象中所產生的 Who, What, When, Where, 以及 Which，
- 而文件倉儲的建置目標是協助使用者了解 Why？



文件採擷 (Text Mining)

- 以文件關鍵字索引的建立、文件特徵的擷取、文件的分類、文件的總結及文件的分群等文字分析的技術來對以純文字構成的文件做分析，產生可結構化的資訊，
- 盡可能的以結構化的形態將已分析過的文件資料加以表示，
- 最終則是根據企業模式來建立文件倉儲
- 以前稱為 “Information Retrieval”



論文投稿信件應用範例

啟者：

欣逢第七屆國際資訊管理學術研討會在 貴校舉辦，謹寄上由本人所撰寫的論文投稿。我們的論文題目為：

從資料處理的演進過程看資料庫系統的發展走向

我們投稿的組別為“資訊管理組”，可以歸類為下列幾個領域之一

1. Database Management
2. Data Engineering

期盼能參與盛會，共襄盛舉。敬祝

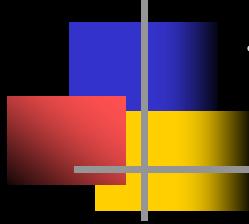
教 安

國立高雄第一科技大學

資訊管理系 曾守正 敬上

E-mail: imfrank@ccms.nkfust.edu.tw

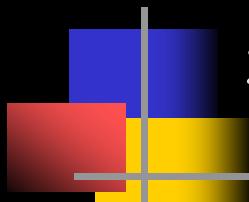
URL: <http://www2.nkfust.edu.tw/~imfrank>



論文投稿信件轉入資料表格

屬性	內含值
研討會名稱	第七屆國際資訊管理學術研討會
作者	曾守正
論文中文題目	從資料處理的演進過程看資料庫系統的發展走向
論文組別	資訊管理組
歸屬領域	Database Management, Data Engineering
電子郵件信箱	imfrank@ccms.nkfust.edu.tw

如何萃取這些資料並存入結構化的資料表是新興的研究領域!

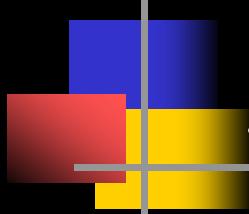


資料倉儲 vs. 文件倉儲

	Document Warehouse	Data Warehouse
相同點	<ol style="list-style-type: none">具有相同的建置步驟，但是處理的資料一為 Documents，一為 Formatted Data。都要面對大量的商業資料，並產生有用的資訊。使用者可以快速瀏覽這些有用的資訊，萃取所需的資訊並進行比較。都必須要能吸納各種不同的異質性資訊來源。兩者形成互補關係 (Data Warehouse 協助我們看出 Who, What, When, Where, Which； Document Warehouse 則協助我們了解 Why?)。	

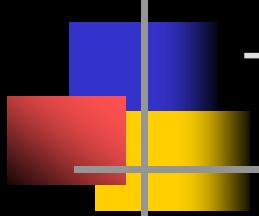
資料倉儲 vs. 文件倉儲

	Document Warehouse	Data Warehouse
相異點	1. 希望得到 Text-oriented business intelligence。	1. 希望得到 Numeric-oriented business intelligence
	2. 資料來源為：企業內部文件檔案（或是文件庫）、會議記錄、研討會論文集、市場調查報告、市場研究報告、產業公報、政府機構發行之文件、E-mail、合約書、廣告信件。	2. 資料來源為：企業內部資料庫、POS (Point-of-Sale) 系統、ERP (Enterprise Resource Planning) 系統、財務或會計系統。
	3. 可以協助使用者快速過濾大量文件，但是難以精確分析出精準的結果，目標是萃取出某些問題的原因 (Why)。	3. 分析結果相當精準，並且可以依據人 (Who)、事 (What)、時 (When)、地 (Where)、物 (Which)，動態切換與檢視。
	4. 將大量文件的特徵與摘要萃取出來後予以分類歸納。	4. 將交易後的數字資料依據所需的維度加以統計、加總後呈現出來。
	5. 比較難以使用固定結構的 Relational Database 來儲存。或許可以利用 UML 設計，並配合 Native XML 資料庫管理系統，以物件導向的型式來儲存。	5. 可以使用固定結構的 Relational Database 來存放原始交易資料。
	6. 需配合 Text Mining 技術來做文件特徵的萃取。	6. 需配合 Data Mining 技術來做各種資料分佈與群聚特徵的萃取。



知識管理系統

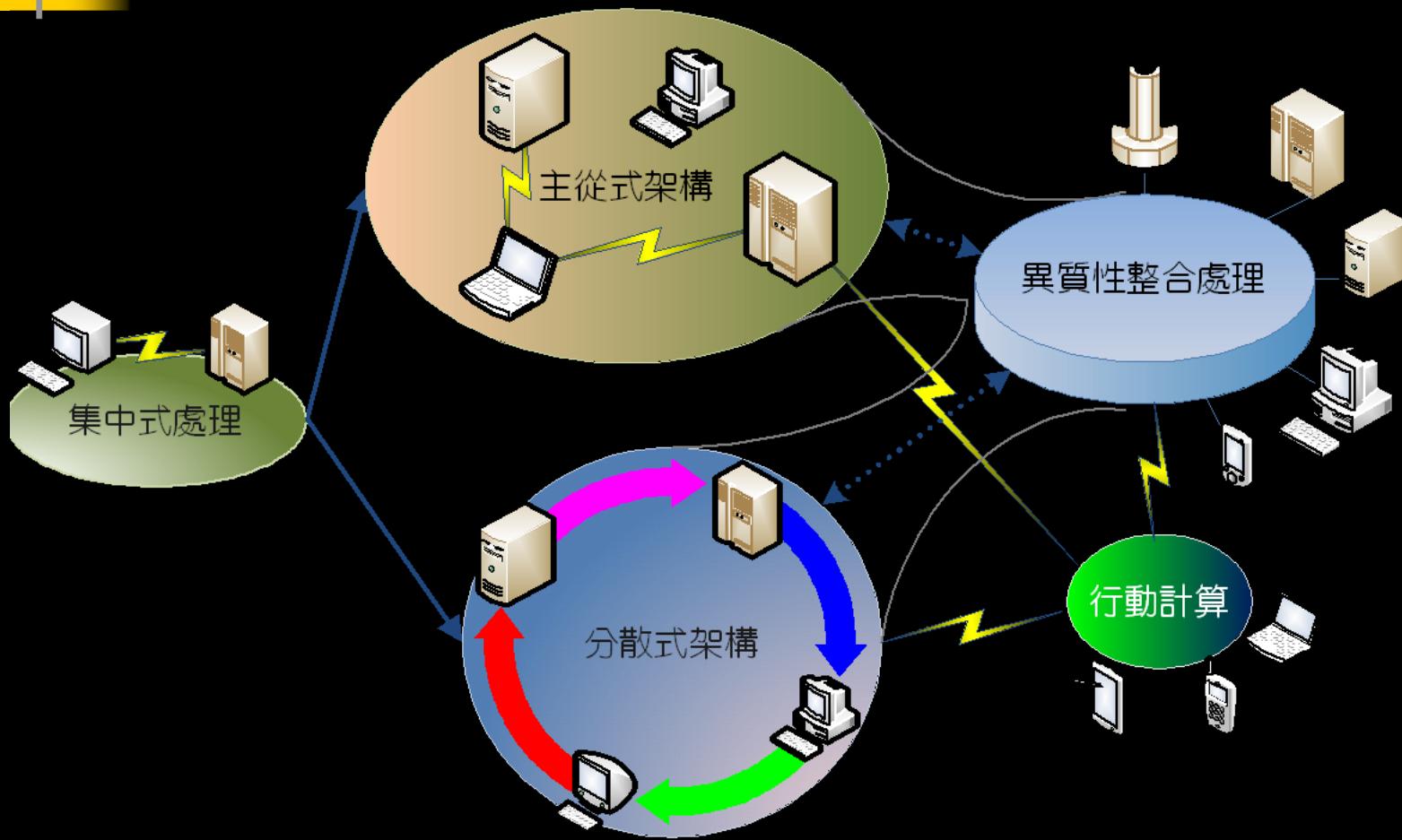
- 隱藏在人與人之間 (People-to-People) 的「隱含知識」 (Tacit Knowledge)
- 隱藏在人與文件資訊 (People-to-Information) 之間的「明確知識」 (Explicit Knowledge)
- 隱藏在人與資料 (People-to-Data) 之間的「潛在知識」 (Potential Knowledge)
- Domain Knowledge 可透過建置 Domain Ontology, XML Topic Map (XTM), Resource Description Format (RDF) 等模式來共享



The Essence of Knowledge Management

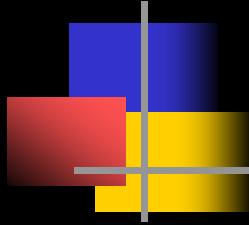
- Arthur Anderson: $K = (P + I)^S$,
 - K : Knowledge,
 - P : People,
 - + : Technology
 - I : Information,
 - S : Sharing.
- 透過科技將人的經驗與資訊進行整合, 透過分享機制, 創造更多的知識

資料處理的未來發展



資料處理的未來發展(續)

	1965 ~ 1975	1975 ~ 1985	1985 ~ 1995	1995 ~ 未來
資料模式	階層式、網路式	關聯式	物件導向式	物件導向式?物件關聯式?
主要硬體架構	大型主機	大型主機與迷你電腦	工作站、個人電腦	多重處理機、平行處理?
計算環境架構	集中式	集中式	主從式/分散式	分散式/異質性、無線式行動計算、雲端計算?
資料存取單元	檔案	表格、記錄、欄位並存	表格、記錄、欄位、物件並存	表格、記錄、欄位、物件並存?
使用者介面	表格式(透過應用程式)	結構化查詢語言(SQL)	圖形化介面/SQL查詢語言	圖形化介面/SQL查詢語言/類自然語言查詢/語音查詢?
資料分析方式	報表系統 (Reporting System)	試算表系統 (Spreadsheet)	決策支援系統 (Decision Support System)	資料倉儲/資料剖析 (Data Warehousing and Data Mining)?



本章結束
(The End)