軟體元件技術與 ERP

中央大學資訊管理系　許智誠
khsu@mgt.ncu.edu.tw

軟體元件化(software componentry)是目前先進企業軟體(包括 ERP 系統)建構技術的大方向。這門技術是以先前的軟體物件(software objects)、軟體架構(software architectures)、軟體結構(software frameworks)、軟體設計模式(software design patterns)以及所有這些概念的物件導向程式撰寫與設計等等相關的理論爲其基礎。它強調軟體元件就像一個硬體元件一樣，最終應該可以重複交替地、穩定地被使用 (例如在電信中的各個硬體元件，即是具備了這樣的特色)。

軟體元件是一項封裝軟體功能的軟體技術。從物件導向程式設計的觀點來說，一個軟體元件常常是以一個物件(Object)的形式存在著(可能是二進位可執行檔或是文字形態的檔案)，並遵從介面定義語言(Interface Definition Language， IDL)的定義，以致於可以和其它元件獨立共存於一台電腦之中。

元件導向的程式設計觀念較著重於撰寫一個可以有效地被重複使用的軟體元件，而這個元件應該可以滿足下列的需求：

- 能被完全的予以文件化(documented)。
- 能夠更徹底地進行測試。
- 對於輸入的參數能夠進行更完固的有效性驗證。
- 能夠適當地傳回有用的錯誤訊息。
- 在往後的程式之中也能夠重複地被使用。

軟體元件的歷史

軟體應該被元件化的觀念起源於預先構思的元件(prefabricated components)。這個觀念是由 Douglas Mcllroy 於 1968 年在德國 Garmisch，NATO 的軟體工程研討會其中的一場演說中首先被提出的，當時他演講的標題爲「大量製造的軟體元件」(Mass Produced Software Components)。這場研討會開始反擊所謂的「軟體危機」。他接下來關於使用 pipes、filters 應用在 Unix 作業系統上的想法，成爲了第一個實作了這樣觀念的基礎架構(infrastructure)。

至於現今軟體元件的觀念則大致上是採用 Brad Cox 的定義，他稱呼他們的軟體元件爲軟體 IC(Software ICs)，並且藉由創造了物件化的 C 語言來開始這樣的元件基礎架構與他們的市場(他將這些觀點寫在他 1986 年所出的一本書中－Object-Oriented Programming – An Evolutionary Approach)。然而，Cox 這樣的企圖卻沒有達成。

在另一方面，微軟公司(Microsoft)也努力地爲他們的元件技術鋪路，最廣爲人知的便是 OLE 與 COM，成功地封裝與簡化物件介面(object interfaces)。時至今日，已經有許多不同而且成功的軟體元件模型誕生。

軟體元件與 ERP 系統

鑑於軟體元件化的諸多好處，ERP 系統也多將其程式元件化。例如 SAP 提供了許多 Business Objects 做爲其系統自身運作及與外界系統建構介面時的基礎，外界系統與 SAP 的溝通得透過 BAPI，而每一個 BAPI

一定得定義於某個 Business Object 上，透過 Business Object Repository 程式師可看到 SAP R3 中的所有可用的 Business Object。這樣一來相關的 BAPI 都可透過同一 Business Object 來管理其生命週期 (Life cycle)、商業邏輯 (business logic)、限制(constraints)等。

軟體元件與分散式軟體元件技術

隨著軟體元件和分散式運算技術的發展，目前已經出現了許多不同且成功的分散式物/元件模型。CORBA(Common Object Request Broker Architecture)即是一項被廣爲使用的分散式物/元件技術。CORBA 可以相容於 C++、Java 等多數的程式語言，並提供不同作業系統與程式語言之間遠端物件的相互溝通與運作。

相對於此，微軟公司也發展出自家的一系列物/元件技術。從起初推出的 COM(Component Object Model) 技術，讓一台電腦內的物件之間可以進行資料交換的作業，到後來發展的 DCOM(Distributed COM)、COM+(Component Object Model plus)，使得物件之間資料交換的作業可以跨越網路來運作。2000 年，微軟公司發表了名爲.NET 的新一代分散式處理技術，這項技術讓使用者可以易於設計、開發與運用 Web 服務(Web Services)。Web 服務以 XML(eXtensible Markup Language)作爲物件之間溝通的基礎，以 SOAP(Simple Object Access Protocol)作爲資料傳遞時的通訊協定，讓物件可以跨異質的平台環境、程式語言來作遠端的呼叫與使用。目前微軟公司將這些技術與相關的應用服務(例如應用程式的設計、開發、部署、使用以及管理等功能)，全部整合在名爲.NET Enterprise Servers 的伺服器應用軟體之中。

在另一方面，昇陽公司以 Java 爲基礎，在發展物/元件技術方面也具有領先的地位。1999 年底，昇陽公司正式發表了 J2EE(Java 2 Platform， Enterprise Edition)，結合了網路上的各項應用服務，例如：JMS(Java Message Service)、JTA(Java Transaction APIs)、JNDI(Java Naming and Directory Interface)等等，以及符合企業應用程式需求的 EJB(Enterprise JavaBeans)技術，提供了在企業應用程式上的一個完整架構。EJB 是昇陽運用在企業伺服器上最核心的元件技術，主要是由 Enterprise JavaBeans 和 EJB Container 所組成。EJB 提供了企業應用程式主要的商業邏輯，而 EJB Container 則提供了企業應用程式於執行時期的安全性(security)、生命週期(lifecycle)、執行實例輪調(instances pooling)、同時共用(concurrency)、永續性(persistence)等等各項機制之管理。EJB 簡化了程式開發人員在程式中對於上述管理機制的撰寫，而只要集中精力於開發包含主要商業邏輯之軟體元件。此外，J2EE 對於 Web 服務也有廣泛的支援。透過一些延伸套件的使用，就可以輕易地讓原本的 EJB 支援 Web 服務的架構。比較普遍被使用的例如：JAX-RPC(Java APIs for XML-based RPC)、JAXM(Java APIs for XML Messaging)、JAXR(Java APIs for XML Registries)等等，這些由 JSR(Java Specification Requests)規範實作出來的套件目前皆是由 JCP(Java Community Process)來製定與維護。

分散式軟體元件技術與下一代的 ERP 系統

基於上述分散式軟體元件技術標準的高度成熟，目前 ERP 系統多已開始用分散式軟體元件技術標準來建構與結合其下一代的產品。例如 SAP 目前的主打系統 NetWeaver 即爲同時支援 Web Services、.NET、J2EE 的產品。以後的 ERP 系統將會是以各個相關的 ERP 及非 ERP 軟體元件透過 Web Services、.NET、J2EE 來共同達成企業執行任務的綜合性系統。