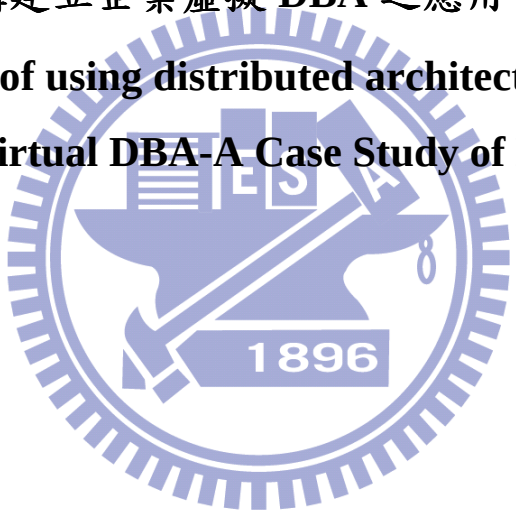
國 立 交 通 大 學

管理學院(資訊管理學程)碩士班

碩士論文

利用分散式架構建立企業虛擬 DBA 之應用-以 M 公司為例

The application of using distributed architecture to establish Enterprise Virtual DBA-A Case Study of M Company



研究生：陳卜瑞

指導教授：蔡銘箴

中華民國九十九年六月

利用分散式架構建立企業虛擬 DBA 之應用-以 M 公司為例

The application of using distributed architecture to establish Enterprise Virtual DBA-A Case Study of M Company

研 究 生： 陳卜瑞　　Student: Pu-jle Chen

指導教授： 蔡銘箴 博士　　Advisor: Min-Jen Tsai

國立交通大學

管理學院(資訊管理學程)碩士班

碩士論文

A Thesis

Submitted to Institute of Information Management

College of Management

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science in Information Management

June 2010

Hsinchu, Taiwan, the Republic of China

中華民國　九十九 年　六　月

摘要

企業因應不同業務流程而建立了各種輔助作業之資訊系統，系統所使用之資料庫規模會隨業務流程擴展而成長，資料庫管理人員的負擔與挑戰也會逐漸加大，不同資料庫之間的關聯性也會因交叉參考而日漸複雜，使系統開發人員找尋正確資料來源需花費的時間變長。

本論文研究利用相關技術文獻之研究，以分散式架構來建立虛擬 DBA 管理系統，期望利用系統管理功能來集中管理資料來源之綱要(Schema)與權限，協助資料庫管理人員及系統開發人員進行日常資料庫之維護及管理工作，透過建立資料來源關聯，以及系統與人員職掌資料的建立，能方便以應用系統為構面之管理角度來管理資料庫，也方便系統開發人員快速取得相關資料庫資訊，進而增加開發之效率，以及提高系統之穩定度，確保公司流程運作順暢。

關鍵字：分散式架構、虛擬 DBA、資料庫管理、資訊安全、物件導向分析設計

Abstract

The enterprise establishes information system to deal with different operation flow. Therefore, the scale of databases expand along with the operation flow grows. The database administrator’s burden and the challenge also enlarge. In addition, the inter connection between the different databases’ become more complex day by day. It also lengthen the system developer to pursue the correct data source.

This study use relation technique to establish the Virtual DBA management system by the distributed architecture. The goal is to expect that the system administration can summarize of Schema and the jurisdiction centralized management data source. It also assists the database administrator and the system developer carries on maintenance and the supervisory work. By establishment of the data source connection, as well as the system and the personnel duties material's establishment, it can facilitate the management of the database with the system architecture. The study also facilitates the system developer to obtain the related database information fast, then increases efficiency of the development, as well as enhances stability of the system, guaranteed that the company flow operation is smooth.

Keyword：Distributed Architecture、Virtual DBA、Database Management、Information Security、Object Oriented Analysis & Design

謝誌

鳳凰花開，離情依依，很快的兩年的碩士學生生涯即將告一段落，回首過去，從職場上重回校園生活，原本工作與課業要兼得就不是件輕鬆的事情，但可以從實務角度去認識理論，算是收獲良多，與老師及幾位不同職場的同學結緣，也算是學問獲得之外的額外珍寶。

本論文得以順利完成，首先要感謝指導教授蔡銘箴老師，於寫作期間不斷從旁悉心的指導。其次感謝口試委員林妙聰教授與巫木誠教授費心的審閱，論文口試中 提供了許多寶貴的意見，使本論文能更加的完善。

求學階段及論文撰寫期間，深受同Lab同學東良、國緯，姿琪，國智給予諸多的協助以及鼓勵，特別是東良同學在暑假期間小女出生當時在課業上的協助，另外公司同事及上司大力的協助與包容，在此表示衷心的感謝。

特別感謝我的老婆瑾瑜，在學校上課期間獨自辛苦照顧寶貝女兒妲圻，也感謝她對我的忍耐與包容，最後要感謝我的雙親，感謝他們在後面默默的支持我，以及不計較這些日子對他們的怠忽，讓我無後顧之憂。僅將本論文獻給我最我最親愛的家人。

陳卜瑞 謹誌

民國九十九年六月

目錄

圖目錄

第一章、緒論

數位化時代的來臨，電腦應用系統早已成為企業的營運命脈，而資料庫也是應用系統後端最重要的資料儲存裝置，企業因應業務流程之需求，需要不斷建立或更新應用系統之功能，而資料庫系統的規模也隨著日漸增加，根據IDC(International Data Corporation，國際數據資訊)的數據顯示，關連性資料庫市場從2006年的產值166億美元，成長到2007年的186億美元，成長率為12.1%[7]。

資料庫管理人員在企業扮演的角色除了資料庫管理系統的管理及資料的保全，與系統開發人員的緊密合作也是重要的職掌之一，市面上的資料庫管理系統，在資料的管理功能上皆可滿足一般企業之需求，但企業實務上對於資料或人員等管理細則，通常會以人工方式管理。

1.1 研究背景與動機

企業建置許多應用系統來輔助業務流程的運行，除了利用電腦系統的大容量的儲存記憶與快速運算的優勢，也為了取代重覆性人工作業並減少錯誤，常見的ERP(Enterprise Resource Planning)系統或Supply Chain或CRM(Customer Relationship Management)系統都是很好的例子，而這些系統的背後也的確存在許多的資料庫管理系統。

企業不論規模大小，通常會由資訊單位來進行應用系統開發及維護，以及相關的資料庫管理，而常見的資訊活動[10]為程式開發與資料使用以及資料綱要(Schema)及權限設定。但屬於資訊單位的管理系統比較少見，相關整合資訊的獲得可能也較不容易，所以專職資料庫管理人員(DataBase Administrator)便必須仰

賴經驗來進行人工方式管理。

資料庫權限通常會依不同的系統以及相關負責人而不同之設定，企業在人員職務異動或人員離職交接之動作時，由於考量系統運作之正常，通常不會輕易異動資料庫之權限或密碼，使得異動後原系統負責人仍知悉原資料庫之權限，造成管理漏洞。

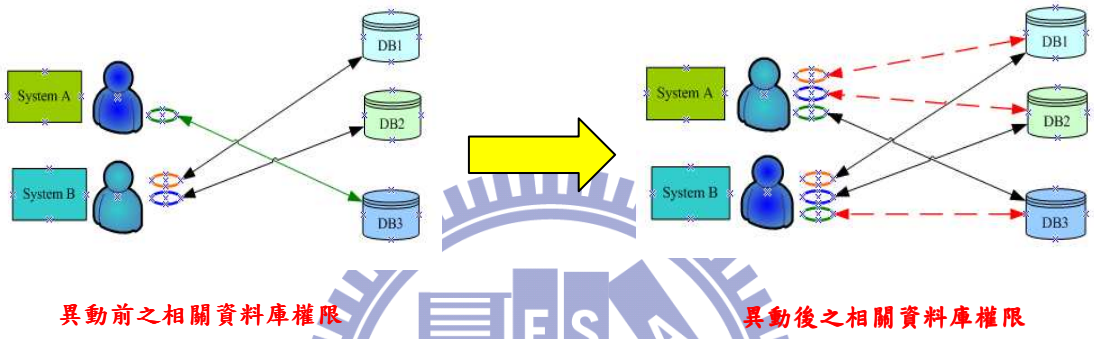


圖 1.1 人員職務異動前後之資料庫權限示意圖

企業內部之應用系統由於業務眾多，所以相關系統資料相互參考的情況頻繁，在最上游之系統或資料庫進行帳號或資料欄位定義異動時，直接連結之系統或資料庫因未得到異動通知同步修改而造成系統錯誤，若要完全通知到所有下游之系統或資料負責人進行同步更動，的確不易完全達成，也是企業的管理挑戰。

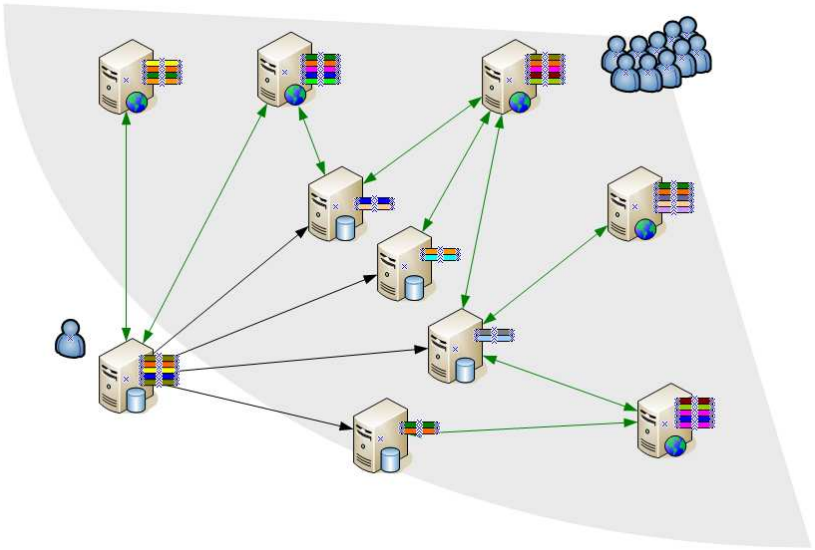


圖 1.2 資料來源之參考關聯圖

新系統建置或是進行系統環境移動時，以人工的方式來建置除了費時且出錯機率很高，而資料庫管理系統提供之備份還原的功能，並無法完全提供以系統（跨Server）為角度來建置系統或進行災難復原，

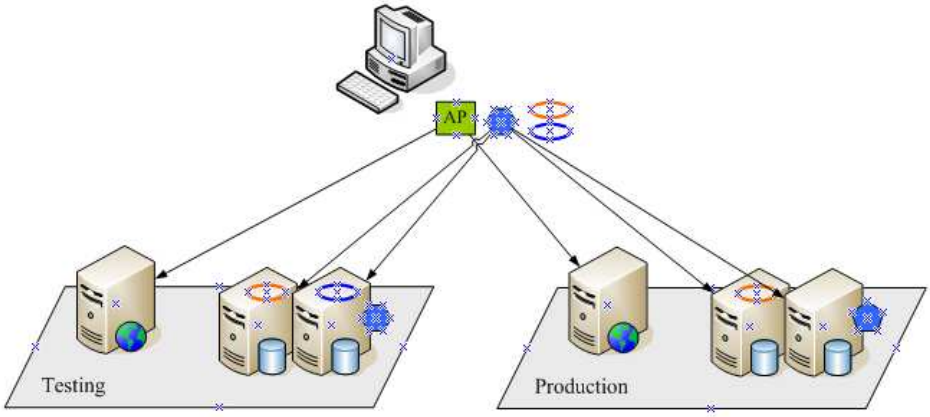


圖1.3 人工進行系統部署之示意圖

1.2 研究目的

目前大部份企業之業務流程幾乎都已朝向電子化發展，而資料庫是企業資訊系統的核心，在本論文研究中，期望透過研究個案相關現況分析以及利用目前較為成熟之資訊技術以及管理方法，建立相關的系統功能，以改善研究個案之內部管理漏洞。

本研究希望藉由 Virtual DBA 管理系統的建立，輔助資料庫人員於資料綱要(Schema)的管理與授權，並進一步使系統開發人員更快速及方便的使用資料庫，並以下列四大研究目的為重心。

- **提供單一存取異動介面，集中控管權責**

一般資料庫會提供服務給兩種角色，一種為系統開發人員以及其開發維護的資訊系統，另一種則是未透過系統之資料庫管理人員，由於市面上資料庫軟體，不論是否需要付費使用，種類相當多，也具備各種不同功能，但簡單之資料庫存取修改功能都有具備，但因此也增加了資料安全管控上的困難，若能透過單一系統介面來集中控管以上兩種角色之資料庫存取權限，使用者在沒有足夠權限下，便只能利用系統介面來進行資料庫操作，因而可防止許多內部未經授權的資料庫操作行為。

- **透過人員維護職掌關係，即時同步人員權限**

通常企業會以書面之方式（如系統維護文件）來記錄資訊系統維護人員以及資訊系統相關資訊，而通常人員異動前之維護交接動作，也是以此文件為主，但隨著交接程度的細膩與否，容易因人為疏失而造成資訊的斷層，也會增加承接

者日後出錯的機率，若能將資訊系統與維護人員之職掌關係建立在系統中，便能一目了然誰負責那些系統，這些系統又使用那些資料庫，在人員職掌異動時，便只要交接細部程式內容即可，如此除了不會有資訊斷層，也可節省交接之時間。

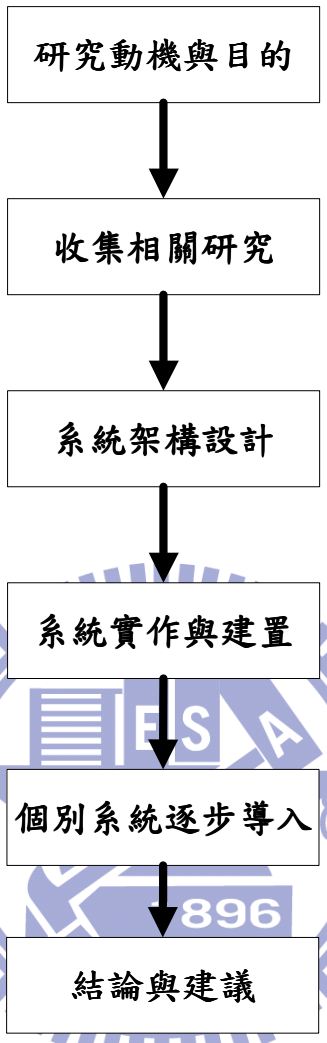
- **提供存取記錄與異常偵測機制**

一般資料庫管理軟體皆有具備存取及異動記錄之功能，但由於連結資料庫之使用帳號，通常不為員工帳號，而為資料庫之系統帳號，故在異常發生之時，也難以就帳號去判別是人工或資訊系統所為，也難歸咎是那些個員工之行為，若能透過相同介面之系統來提供存取，在異常情況發生時，責任的歸屬以及發生當時立即要通知的對象，都可直接透過Alarm的機制來讓資料或系統負責人第一時間得知，並立即處理，以免影響層面擴大，事後補救之成本更大。

- **建立資料來源關聯，減少系統部署時間及異動錯誤**

企業裡最難管理的就是內部資料來源的關聯性，由於資訊系統及資料庫分散，資料欄位的定義有時會出現相同意義，但卻在不同系統中有不同的屬性定義，另外也會出現跨系統的資料參考之現象，如人事資料或會計財務資料，而資料上下游所形成的資料流，也是日後異動資料定義時，最容易造成下游系統未同步更新而發生錯誤的原因，若能將企業之欄位定義集中管理，並明確記錄跨系統之資料流關係，除了在建立新系統時不會出現屬性不對之欄位，另外在系統移轉或者災難復原時，可快速得知參考資料之上下游，便可輕易進行系統的設定，節省轉移及復原之時間，並可減少人工操作之錯誤。

1.3 研究步驟



1.4 論文章節說明

本研究主要包含「緒論」、「相關研究」、「架構設計」、「系統實作與建置」和最終的「結論與未來發展」等五章。各章的主要內容說明如下：

第一章 緒論

說明設計本論文之應用系統的動機、目的。

第二章 相關研究

針對本論文所設計之應用系統其實務應用與理論之相關技術架構參考文獻進行分析。

第三章 架構設計

說明本論文所設計之應用系統其架構及進行的方式。

第四章 系統實作與建置

說明本論文所設計之應用系統其實作之模組與功能。

第五章 結論與未來發展

說明本論文所設計之應用系統已產生之效益以及未來可再延伸應用之範圍。

第二章、文獻探討

本章首先將探討有關資料庫管理系統之運作，以及資料庫管理人員常用之工作內容，在技術領域則將探討以Microsoft所發展之分散式架構可為應用系統帶來什麼效益，以及分散式架構常用之Web Service技術，接著探討以多層式架構為基底的程式開發標準相關技術架構，如Microsoft Enterprise Library與O/R Mapping(Object Relation Mapping)，最後會探討操作SQL Server底層介面功能所使用的SMO物件(Server Management Object)，以作為本論文所提出架構方法之參考依據。

2.1 資料庫系統的使用成員

一個資料庫系統使用成員有四類，分別為使用者、直接使用者、應用程式與資料庫管理員[11]。

- 使用者：經由開發的程式取得資料庫中資料，並不會直接與資料庫系統接觸。
- 直接使用者：使用資料庫系統所提供指令，直接對資料庫下達查詢指令，此類人員多為企業內資訊部門中程式開發人員或應用程式負責人員。
- 應用程式：開發應用程式向資料庫管理系統下達命令，例如企業開發轉檔程式。
- 資料庫管理員：負責資料庫管理工作的人員，負責定義資料儲存結構、

管理資料庫綱要「Database Schema」、資料庫安全控制、資料備份與回存、監督、調整資料庫效能並排除資料庫使用上等問題。

2.2 資料庫管理系統

DBMS (Database Management System)是一種應用軟體。此一軟體能有系統地建立、更新、儲存及取得資料庫中資料。使用者與開發的程式人員能夠共享資料，同時程式與程式間亦可共享資料，無須為每個程式產生與儲存新的檔案。DBMS亦能提供資料存取控制、管理好資料的完整性與一致性的控制，以及資料庫的復原機制[9]。

2.3 資料庫管理系統架構

資料庫管理系統為方便使用者使用，系統中複雜的資料結構與相關存取控制特地隱藏起來，使用者只須留意資料本身的處理。其目的將使用者應用程式與資料儲存體分開，達到「資料獨立」及「資料分享」[10]。

現今資料庫管理系統大多採用美國國家標準協會系統綜合規劃委員會ANSI/SPARC(American National Standards Institute / Standards Planning And Requirements Committee)中的資料庫管理小組所制定的資料庫系統架構，其設計準則採ANSI/SPARC三階層網目架構如圖2.1所示。

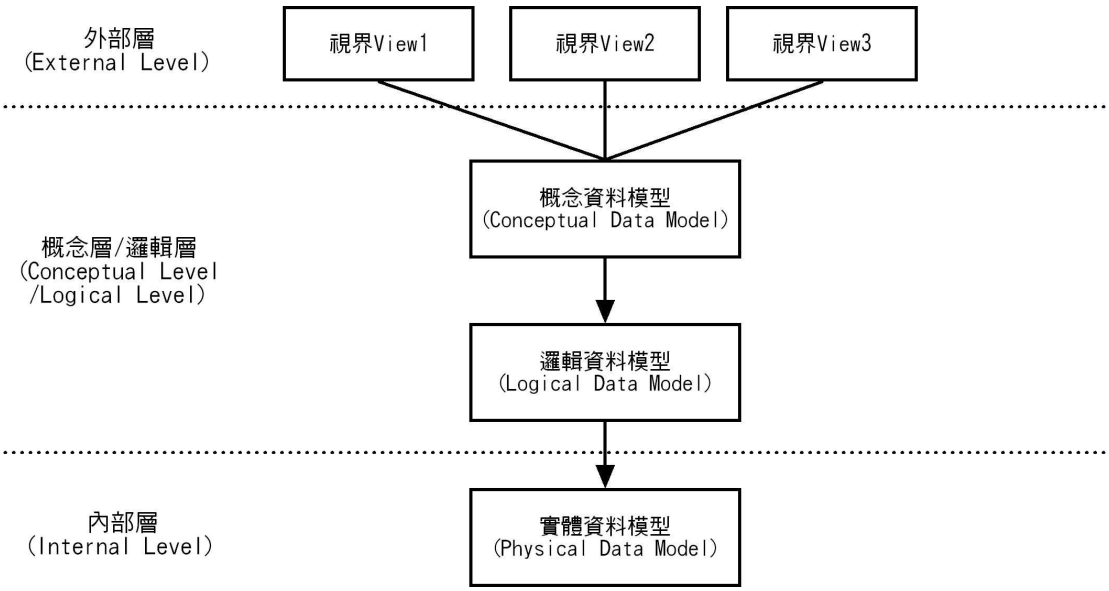


圖2.1 資料庫系統三階層網目架構[10]

1. 外部層次(External Level)：為使用者層次，每一個概略表可看成不同的使用者。

2. 概念層次(Conceptual Level)：主要在描述資料庫管理系統中全部資料邏輯結構和特徵，其中概念綱要(Conceptual Schema)在定義資料庫實體與關聯邏輯含意。

3. 內部層次(Internal Level)：為實體層次(Physical Level)或儲存網目(Storage Schemas)，其中內部綱要(Internal Schema)代表資料儲存的描述，定義檔案儲存體的資料集合，檔案可以是儲存概念資料庫中一個或多個實體描述。

2.4 DBA(資料庫管理員)工作內容研究

本節主要研究SQL Server 2000/2005資料庫管理人員日常作業[11]以及任

務，來作為本系統開發之參考。

- **安裝和設定**

DBA 常被要求支援軟硬體的安裝及協助設定。在一般的維護工作上，由於 DBA 必須確定所安裝或設定的系統和資料庫是可執行及和穩定的，因此整個系統的設定亦皆屬責任範圍內。

- **維持安全性**

監視系統的安全性及隨時報告所發現的問題是 DBA 的另一項責任。一個系統需要什麼種類或多少份量的安全系統，取決於使用系統的權限範圍。如一個沒有與網路相連接的系統僅供給幾位可信賴的員工使用，這樣的系統所需的安全裝置就不會比一個與網路有連接的系統所需要的安全裝置來得複雜。除了管理使用者之外，DBA 也需要設計並實施網路保全計劃，這方面的工作通常會指派給具有網路保全相關經驗的資深人員執行。

- **系統稽核**

系統稽核包括監控 Sever 錯誤記錄檔和作業系統事件記錄檔。這些記錄檔和事件記錄檔記載 DataBase Server 以及作業系統保全有關的重要資訊。必須定時監控這些資料以確保系統內部的安全。如 data definition language (DDL) statement 和插入、更新及刪除的功能。還可用來監控特定的事件、登入的時間、使用者名稱及相關活動。

- **備份和回復**

執行規則複製和測試備份回復 DBA 工作的一部分。如果系統出現了故障，在很多情況下，使用備份是回復資料庫的唯一途徑。但是如果沒有正確的執行備份，就不可能回復資料庫，那會導致資料的遺失和可能好多天的停機，甚至可能造成損失。

- **使用者權限管理**

使用者管理也是 DBA 一項例行工作，包括管理 Server 的登入和使用者權限。任何一個需要使用資料庫的員工都必須先透過 DBA 設定其權限才得以進入系統。通常在企業間開放權限時，是很少完全開放一個人的權限，通常會依照該部門業務所需要的資料來開放相關的權限。

- **調整及監控**

如果突然遇到反應時間變長、所使用的 CPU Time 變大等狀況，可能都是系統出現問題的預兆。當您遇到不同的系統，監控的方式及解釋監控的結果也會隨之不同。因此必須根據所遇到的系統作判斷並解決問題。您也必須定期監控系統的資源使用情形。如此，就可以在系統效能降低時即時調整系統。

- **資源分享**

DBA 有時也會被要求成為開發者、設計者和使用者的顧問角色。幫助一般使用者有關特定問題的協助，或者開發訓練課程，甚至親自教授課程。幫助開發人員，提供關於過去系統使用的情況以及新的進展對使用者的好處等資訊。這可能包括通知使用者可用的新資料表和索引，以及所有可能有用的新特性。幫助設計人員，提供不同的設計特性對使用者有何益處等。有時設計人員開發的應用程

式可能缺少一些使用者希望或需要的特性，而將這些資訊傳遞給他們可能有助於將來的設計。如果使用者對於使用某些功能有問題，此時他們更可能向您提出問題，因此您是設計人員很好的資訊提供者。

2.5 多層式架構(Multi-Tier Architecture)

分散式架構(Distributed Architecture)在近年來不斷的被討論[4][5][14]，它是一種全面性的IT架構變革，與傳統Client／Server或單機介面不同之程式架構，在企業業務流程企業在考量確保過去IT的投資，又可持續保有IT架構的發展與發揮IT系統所應帶來的效益，分散式架構的導入，已在多數企業的心中醞釀。

本研究之企業樣板相關概念的基礎和架構，是衍生自 Microsoft 近年來為降低企業環境應用程式之複雜性所開發的分散式應用程式架構[14]。這個模型有時也被稱為「多層式架構」，因為以邏輯組合的概念最容易瞭解它的意義。因為使用的是樣板，樣板在這專案開發之前就已經為您完成了一些架構。這項優點可以減少您的工作量和規劃的時間。各種分散式應用程式樣板提供一種開發應用程式(使用分散式架構) 的標準方法。只要使用樣板，就是選擇在標準的位置上實作服務。您並不需要實作所有的層。樣板會取得實作的部分，它們會在與分散式應用程式架構一致的層內實作。舉例來講，如果您需要 XML Web Service ，您只需在 Web 服務專案層中實作它，不用將它加入到用戶端或商務邏輯層。 一般多層式架構分為顯示層(Presentation Layer)、邏輯層(Business Logic Layer)以及資料存取層(Data Access Layer)，如圖2.2所示。

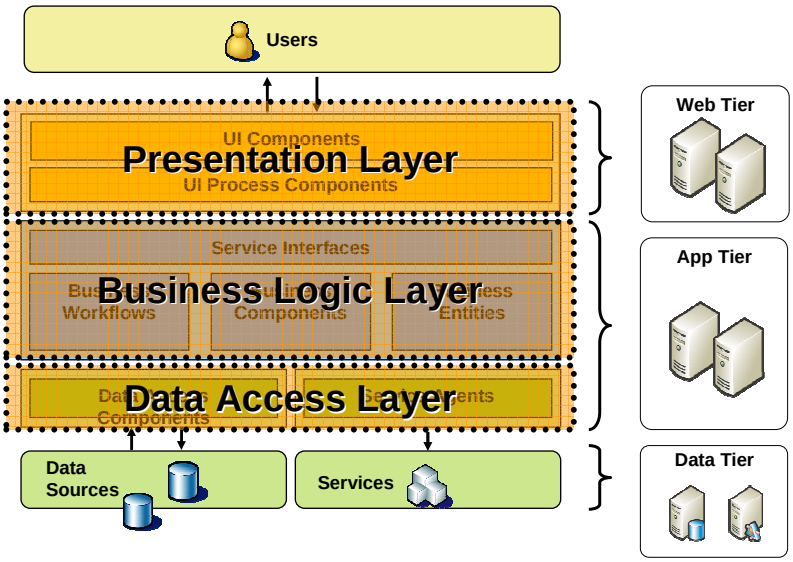


圖 2.2 多層式架構與應用程式用階層對應圖

2.6 程式產生器(Code Generator)

產生器應用於系統的產生者，稱之為應用系統程式產生器，目前已有許多關於應用系統產生器系統的研究[2][3][4][13][15]，製作應用系統產生器能夠有效提升軟體設計的產出，具有降低程式碼的錯誤，能夠使得開發者專注於需求規格的錯誤以及易於建構與測試替代方案的雛型，最重要的優點是藉由產生器自動產生標準介面或輸出規格，易於將建構的過程標準化。

程式產生器分為三種[16]：(1)商業資料處理的應用系統產生器。(2)程式語言的文法與詞彙分析產生器。(3)CASE 工具中的程式產生器。產生器主要是將特定問題域的需求規格(Specification)轉換成程式片段、子程式(Subroutine) 或軟

體系統，本論文研究的範圍如下:

- 利用所使用的System Architecture 及Framework，制定程式範本，用程式產生器產生程式
- Data Access Layer 程式相依於 table schema，最適合用程式產生器來產生程式

2.7 O/R Mapping(Object Relation Mapping)

O/R Mapping 主要用於實作從物件資料(物件導向)到關聯資料(關聯式資料庫)的存儲對映的技術[12][18](參見圖 2.3)。在本論文研究中，主要用來產生與資料庫資料綱要(Data Schema) 對應之類別程式，也是本論文程式產生器使用之主要規則。

O/R Mapping 之運作步驟：從 Data Sources 出發，定義 Pattern Rule (Template)，利用 Code Generator 來產生 Concrete/Base Class，最後進行 Customize Class (參見圖 2.4)。

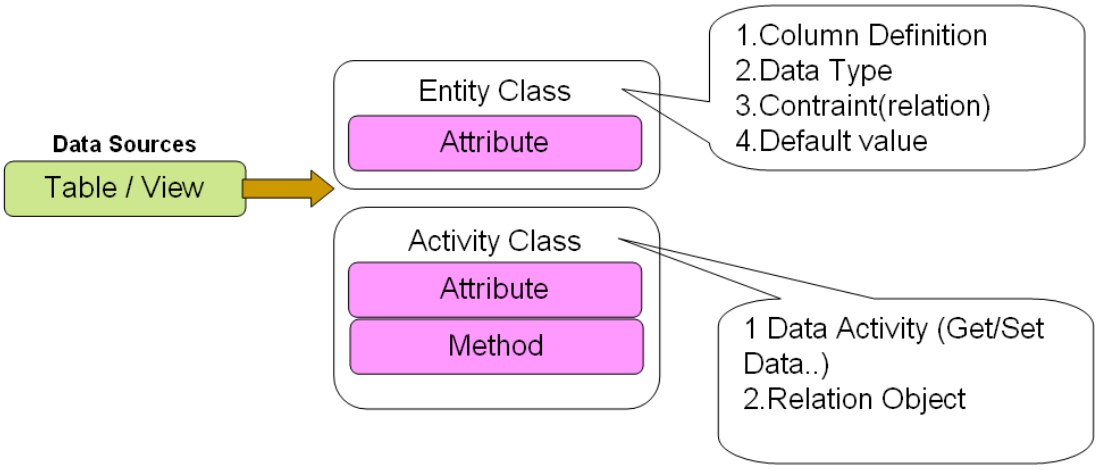


圖 2.3 從物件資料到關聯資料的存儲對映的技術

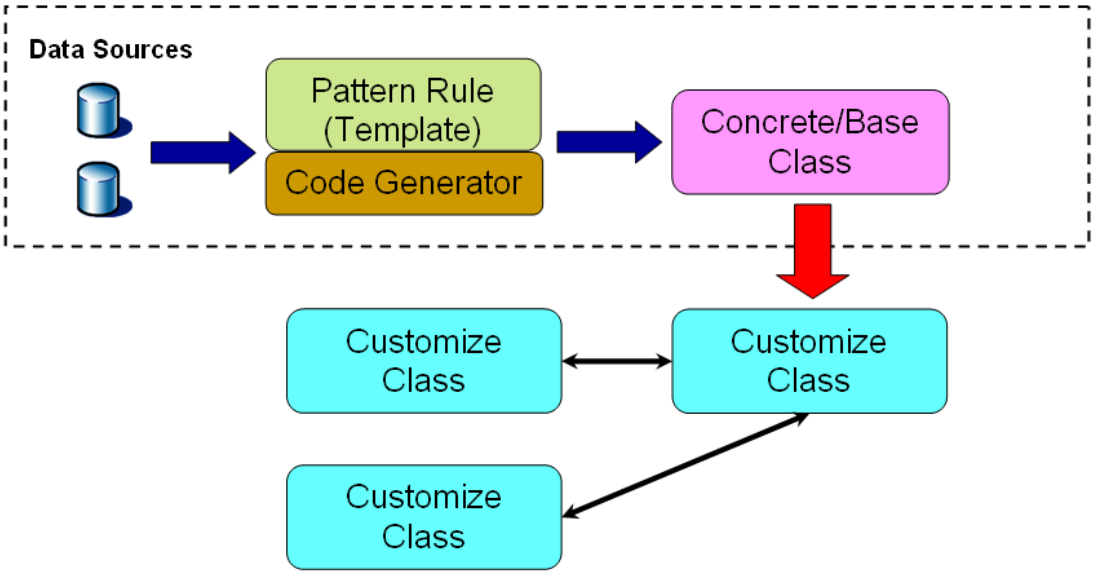


圖 2.4 O/R Mapping 之運作步驟

2.8 Microsoft Enterprise Library Framework

我們在製做大型專案都常會用到一些機制，如異常狀態處理(Exception Handling)、工作日誌(Log)等。在設計與實作上，我們可能會想要有一個架構可以涵蓋我們的需求，並且可重覆使用。

通常自己去設計完全符合自己風格的Framework來解決我們的問題，可是微軟所推出的Enterprise Library算是比完整解決方案了。本論文多層式架構採用之樣版是使用Micorost Enterprise Library[19]來作為Framework 參考。

Enterprise Library擁有幾個優點：

(1) 利用已包裝過的Library來協助我們快速解決問題

(2) 透過設定檔的方式，讓系統在需要改變運作方式時可以快速切換

(3) 所有的Application Block除了內建的功能外，都允許我們自己自訂處理方法

(4) 處理機制都已配合.NET特性，進行過效能的最佳化

Enterprise Library不是用來取代.NET Framework裡任何的功能，它只是更加強化.NET Framework裡原有的不足。Enterprise Library裡內建了六個方面的功能（稱之為Application Block），和一個部份將會和其它Library共用的核心（Core）。

下圖是微軟官方的架構圖，簡單介紹Enterprise Library 2.0的整體架構：

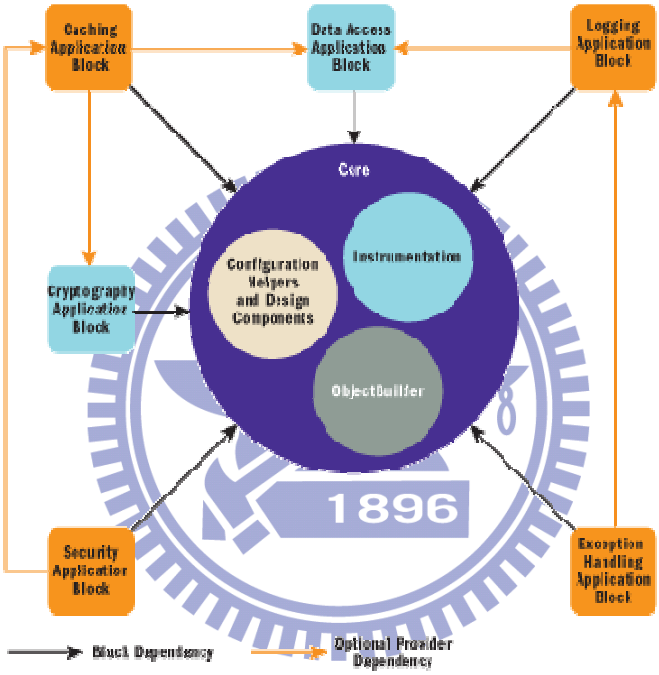


圖 2.5 Microsoft Enterprise Library 架構及運作關係[18]

我們可以把這個架構圖分成幾個部份來看：

(一) 核心(Core)：

- Configuration：Enterprise Library的設定檔就預設是我們平常在使用的app.config或web.config，不過我們也可以改成設定在資料庫中。特殊的是，在設定檔中也允許讀寫除了像string、int之外的自訂複雜型別。在設計的過程中，我們則可以透過System.Configuration來操作設定值。

- Configuration Design & Tools：微軟的團隊在設計這個工具的時候，也考慮到在撰寫app.cinfig或web.config已經愈趨複雜，若要把Application Block的設定也加到裡面，撰寫的困難度一定也會大大提高。所以就設計了Enterprise Library Configuration工具來協助我們撰寫設定檔。

- Instrumentation：EL中所有的Application Block都內建了以下功能來協助我們開發、測試：a. Event Log events b. Performance c.WMI events。

- Object Builder：這是使用設定檔就能夠造成系統運作改變的重要原因，可以發現Application Block都是使用如XXXFactory的類別來產生物件。Object Builder的角色便是當XXXFactory發出需要創建物件的需求時，便讀取設定檔並創建對應的物件以供使用，保證系統運作的正確。

(二) Application Block

- 快取(Caching Application Block)
- 加解密(Cryptography Application Block)
- 資料存取(Data Access Application Block)
- 異常狀態處理(Exception Handling Application Block)
- 工作日誌(Logging Application Block)
- 安全控管(Security Application Block)

本論文設計之系統，乃參考Microsoft Enterprise Library 分散式架構之Design Pattern。架構當中引用Data Access Application Block，Configuration Management

Application Block以及Exception Management Application Block等三大模組來進行系統之Log 記錄、Exception Alert以及Data Access等功能。

2.9 Web Services

由於Web Services的彈性架構與具有跨平台之共通協定，所以成為分散式架構的最佳實作方式[20]，利用WebServices 來建構分散式架構最主要的優點在於Web Services 是公開平台且普遍、架構簡單。以下將說明Web Services的核心技術。

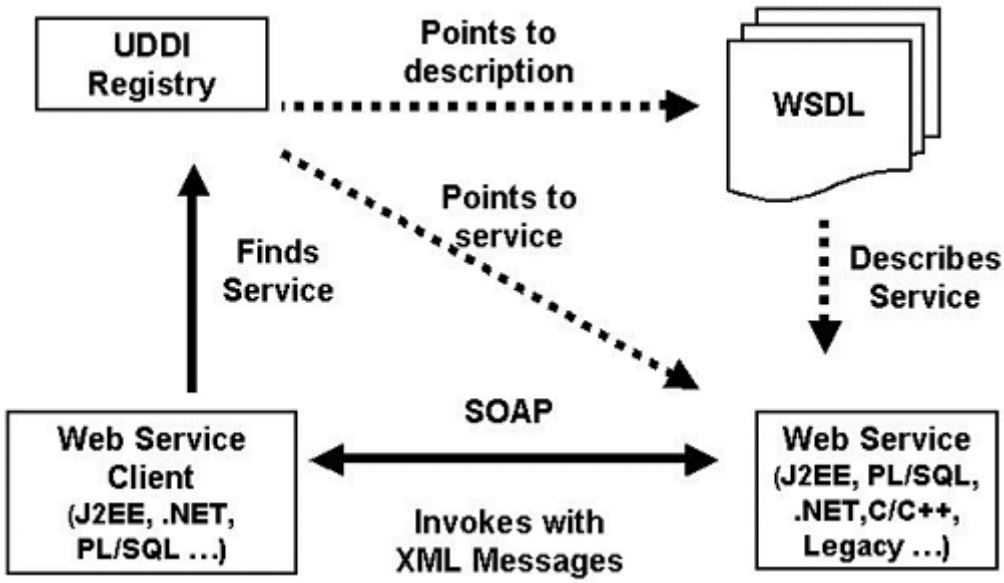


圖 2.6 WebService 架構及運作關係圖[19]

- **XML**

XML（eXtensible Markup Language, XML）是一個開放且標準的可延伸性標籤語言，XML 文件由純文字所表示，一份XML 文件不但具有自我描述的能力，也可用來描述其它語言。因此，XML相當適合在電子商業異質系統整合的應用。

- **SOAP**

SOAP（Sample Object Access Protocol, SOAP）為一個分散式環境中訊息傳遞與呼叫的標準，其訊息的格式是XML 語言，其兼具了平台獨立性與語言中立性。SOAP 的訊息傳輸方式可用HTTP 協定來做同步（synchronous）傳輸，或是利用SMTP 來做非同步（asynchronous）呼叫。因此跨組織間的Web Service物件呼叫、訊息交換或資訊整合皆可利用SOAP 來做為訊息傳遞的標準。

- **WSDL**

WSDL（ Web Service Description Language, WSDL） 是由Microsoft 與IBM 所聯合提出的一個以XML 為基礎的規範語言。當Web Service Provider 欲對外公佈其所提供的Web Service，就必撰寫WSDL 服務描述檔案，並將之註冊到UDDI Service Registry。WSDL 描述內容包括：

1.描述Service Provider 所提供的Web Service operations。

2.描述Service Requester 如何和Web Service 的operations 溝通，內容包括傳輸協定、格式及參數。

WSDL 規範Web Service 輸入及輸出的標準，讓使用者經由此規格即可得知服務提供者所提供的服務、文件資料型態和所需的基本軟硬體需求。所以利用WSDL可以得知Web Service 所提供的服務格式。

- **UDDI**

UDDI（Universal Description Discovery and Integration, UDDI）是由IBM、Microsoft 及Ariba 所共同制定的開放性平台架構，提供不同地理位置的Web Service 在網際網路上互動之機制。UDDI 定義Web Service 的註冊、搜尋和發

現，以提供網路上Web Service的取用及分享。

目前Microsoft及IBM分別設置了UDDI Service Registry 供應用服務提供者註冊其已開發的服務。企業可經由此規格來註冊其所提供的服務，而服務要求者則可藉由服務查詢來取得所需的服務資訊。UDDI 的目標是要讓任何網路上的服務皆可即時的傳遞服務訊息和內容給所需要的使用者，讓服務提供者和服務使用者能隨時隨地的相互溝通。UDDI 目前規格的缺失之一是其發現機制是"被動式的發現"，亦即使用者需向服務註冊單元進行查詢方能得知是否有其所需要的服務出現。

2.10 Microsoft SMO(Server Management Objects)

- **SMO 簡介**

SQL Server Management Objects (SMO)[5]是Micorost針對應用程式開發人員用來與 SQL Server 底層 API 直接溝通所設計的物件。在 SQL Server 2005 之後 SMO 物件模型會擴充並取代 SQL Server 2000 Distributed Management Objects (SQL-DMO) 物件模型。與 SQL-DMO 相較之下，SMO 在效能、控制能力和便於使用的程度上都有提升。且因為 SMO 與 SQL Server 2000、SQL Server 2005、SQL Server 2008 和 SQL Server 2008 R2 相容，所以可以管理多個版本的環境，以下說明本論文所使用之 SMO Object。

- **資料庫連線物件(Server Connection)**

Server Connection 是用來與 SQL Server Instance 建立連線之物件，使用 Server Connection 物件變數的優點是，連接資訊可以重複使用，並利用連接資訊 (例

如，SQL Server 執行個體的名稱與驗證模式)設定屬性。然後傳遞 Server Connection 物件變數，做為 Server 物件建構函式的參數。可以使用 Copy 方法來取得現有連接設定的複本。若要重複使用 Server Connection 通常不需要呼叫 Server Connection 物件的 Connect 方法。SMO 將會在需要時，自動建立連接，並在執行作業完畢之後，將連接釋放到連接集區。呼叫 Connect 方法時，並不會將連接釋放到集區。若要將連接釋放到集區，必須明確地呼叫 Disconnect 方法。此外，您可以設定 Server Connection 物件的 NonPooledConnection 屬性來要求非集區的連接。

- **資料庫物件(DataBase)**

在 SQL Server 管理物件(SMO)中，資料庫是由 DataBase 物件表示。在 SMO 物件階層中，DataBase 物件位於 Server 物件之下。可利用 Create(建立) 、改變(Alter)、移除(Drop)等 Method 來操作。

- **資料表物件(Table)**

在 SQL Server 管理物件(SMO)中，資料表是由 Table 物件表示。在 SMO 物件階層中，Table 物件位於 DataBase 物件之下。可利用 Create(建立)、改變(Alter)、移除(Drop)等 Method 來操作。

- **索引物件(Index)**

在 SQL Server 管理物件(SMO)階層中，索引是由 Index 物件表示。索引資料行是由 IndexedColumns 屬性表示的 IndexedColumn 物件集合來表示。

- **檢視物件(View)**

在 SQL Server 管理物件(SMO)中，SQL Server 檢視是由 View 物件表示。View 物件的 TextBody 屬性會定義檢視。該屬性等於用於建立檢視的 Transact-SQL SELECT 陳述式。

- **觸發程式物件(Trigger)**

在 SMO 中，觸發程序是利用 Trigger 物件表示。在觸發程序引發時所執行的 Transact-SQL 程式碼是由 Trigger 物件的 TextBody 屬性所設定。觸發程序的類型是利用 Trigger 物件的其他屬性所設定，例如 Update 屬性。這是布林值屬性，指定觸發程序是否由記錄的 UPDATE 在父資料表上引發。

- **預存程序物件(StoredProcedure)**

在 SQL Server 管理物件 (SMO)中，預存程序會由 StoredProcedure 物件表示。若要在 SMO 中建立 StoredProcedure 物件，需要將 TextBody 屬性設定為定義預存程序的 Transact-SQL 指令碼。參數需要@前置詞，且必須藉由使用 StoredProcedure Parameter 物件以及加入至 StoredProcedure 物件的 StoredProcedure Parameter 集合來個別地建立。

- **管理使用者、角色和登入**

在 SMO 中，登入是由 Login 物件表示。當登入存在於 SQL Server 時，可以加入至伺服器角色。伺服器角色是由 ServerRole 物件表示，資料庫角色是由 DatabaseRole 物件表示，應用程式角色則是由 ApplicationRole 物件表示。與伺服器層級相關聯的權限會列為 ServerPermission 物件的屬性。伺服器層級權限可授與個別的登入帳戶，也可從這些帳戶拒絕或撤銷。每個 Database 物件都具有

UserCollection 物件，可指定資料庫中的所有使用者。每個使用者都與一個登入相關聯。單一的登入可以與一個以上資料庫中的使用者產生關聯。Login 物件的 EnumDatabaseMappings 方法可用於列出每個資料庫中與該登入相關聯的所有使用者；或者，User 物件的 Login 屬性也可指定與該使用者相關的登入。SQL Server 資料庫也具有指定資料庫層級權限集的角色，這些權限可以讓使用者執行特定的工作。與伺服器角色不同的是，資料庫角色不是固定的。這些角色可以建立、修改和移除。若要進行大量管理，可以為資料庫角色指派權限和使用者。

- **授與、撤銷和拒絕權限**

ServerPermission 物件可用於將一組權限或個別的伺服器權限指派給 ServerPermissionSet 物件。如果是伺服器層級權限，被授與者是指登入。Windows 驗證過的登入會列為 Windows 使用者名稱。當這個程式碼範例執行時，會從被授與者撤銷權限，並使用 EnumServerPermissions 方法確認該權限已經移除。

資料庫權限和資料庫物件權限也可藉由 DatabasePermissionSet 物件和 ObjectPermissionSet 物件，以類似的方式進行指派。

- **備份及還原資料庫和交易記錄**

在 SMO 中，Backup 類別和 Restore 類別都是公用程式類別，可提供工具來完成備份及還原的特定工作。Backup 物件表示所需的特定備份工作，而不是伺服器執行個體上的 Microsoft SQL Server 物件。如果發生資料遺失或損毀，則必須完整或部分地還原備份。部分還原會使用 FileGroupCollection 集合來分要還原的資料。如果是進行交易記錄的備份，則可以使用 Restore 物件的 ToPointInTime 屬性還原至特定的時間點。也可以使用 SqlVerify 方法來驗證資

料。建議的備份程序是定期執行還原作業並檢查資料庫中的資料，以檢查備份的完整性。與 Backup 物件類似，Restore 物件不需要藉由使用 Create 方法來建立，因為它不代表 SQL Server 執行個體上的任何物件。Restore 物件是一組用於還原資料庫的屬性和方法。

第三章、架構設計

本章根據前章節相關技術文獻之探討，進行相關系統架構、功能模組之設計，以期望滿足研究目的之初衷。

3.1 單一存取介面之 Virtual DBA 機制

一般傳統之資料庫授權方式如圖 3.1 所示，系統帳號權限是由 DBA 控管，再分別授權給系統開發人員以及指定系統執行時所使用之帳號，雖可控管但無法防止惡意之帳用盜用或內部管理人員非法之監看機密資料。

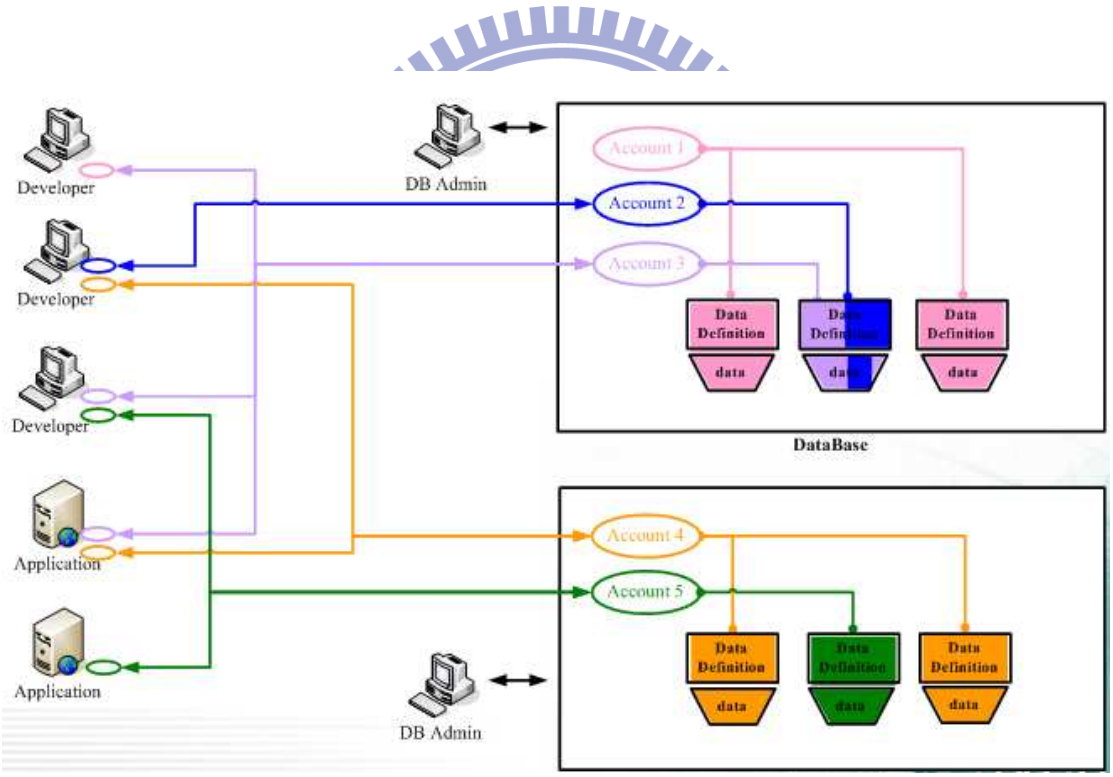


圖 3.1 一般傳統之資料庫授權方式

透過單一存取介面之管理方式如圖 3.2 所示，資料庫之帳號權限統一由 Virtual DBA 集中控管，系統開發人員或資料庫管理人員不需要使用資料庫帳號便可登入 Virtual DBA 系統進行資料庫物件屬性之修改，而實際可存取 DataBase

Server 之帳號，是由 Virtual DBA 所掌控之系統管理權限帳號，再依管理人員或應用系統之職掌資訊，在系統介面上給予適當瀏覽或異動之權限。

所有系統管理權限之密碼則只有部門之權責主管才有檢視或更動之權利，一般開發者甚至是資料庫管員並無法得知。

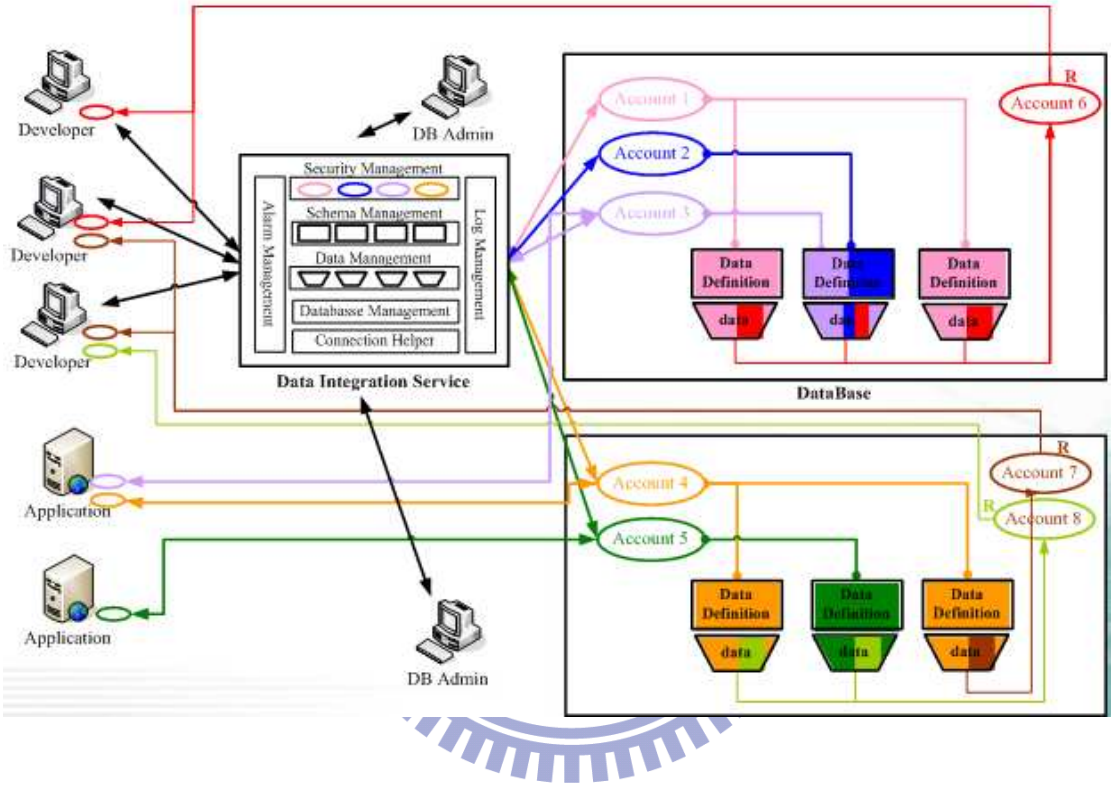


圖 3.2 單一存取介面之管理方式

3.2 依職務授予適當權限

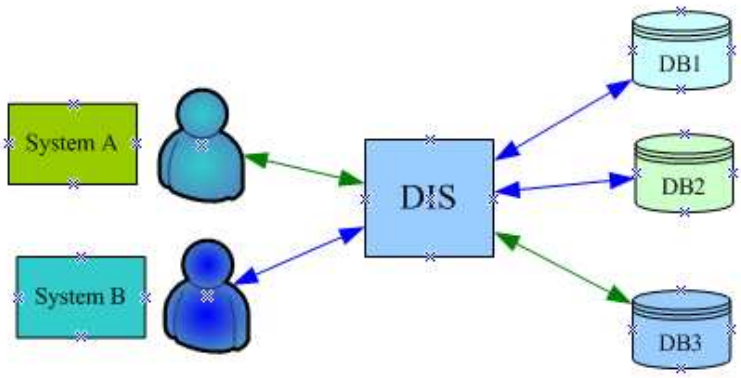


圖 3.3 由管理系統控管人員職掌及資料庫權限對應

系統職掌關係主要在記錄維護人員與應用系統之關聯資訊，而透過這些資訊便可明確定義維護人員具有那些資料庫之異動權限，以圖 3.3 為例，System A 之維護人員與 System B 之維護人員各自有其維護之資料庫。其概念資料流程圖如圖 3.4 所示，首先必須建立 System 以及 Server 相關之資訊，然後再透過員工與 System 之關係，便可明確定義員工職掌，也間接定義那些資料庫之該員工有權利進行異動的範圍。

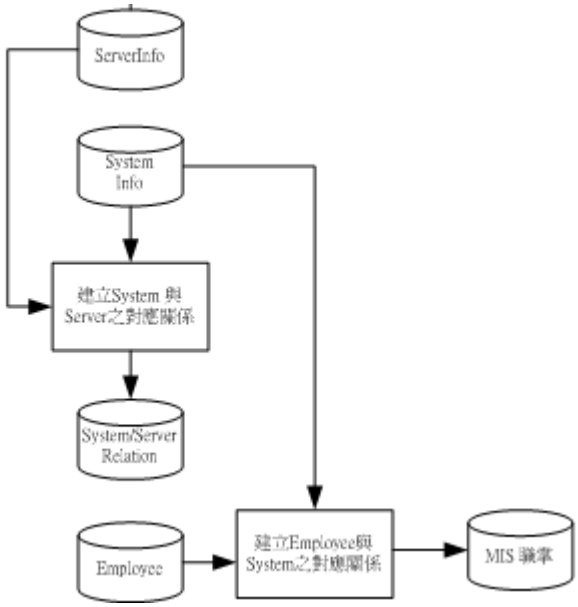


圖 3.4 人員職掌及資料庫權限對應流程圖

3.3 經由資料關聯定義，即時獲得其它系統資料庫異動內容

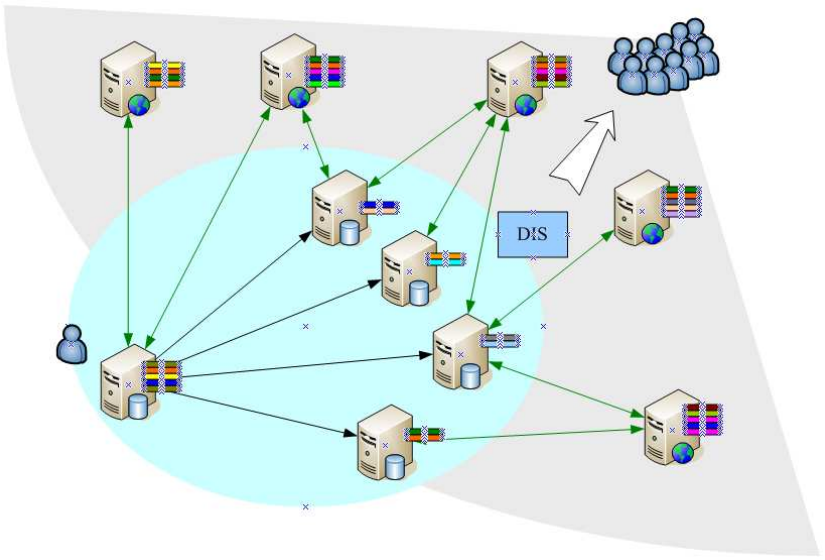


圖 3.5 資料庫來源關聯控管及異動機制

如圖 3.5 所示，企業內部之應用系統其資料來源的參考是很頻繁且複雜，也是較難控管的部份，若能完整記錄資料關聯之上下游關係，任何上游系統的資料屬性更動都可追溯至下游之資料源，且透過資料與系統職掌之資訊，可在第一時間通知相關負責人員，減少異動後所產生的錯誤。

3.4 利用比對功能確保部署完整性

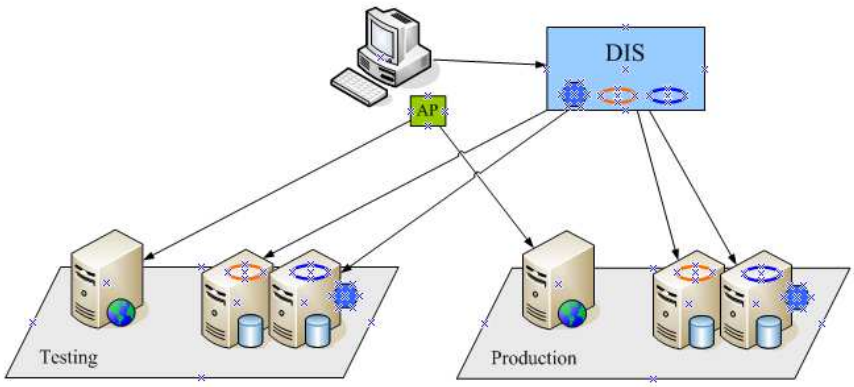


圖 3.6 資料庫欄位比對機制

資料庫的部署常會出現在新系統上線時，以上圖為例，要把 Testing 環境所使用到的資料庫複製到 Production，才能使系統正常運行，目前資料庫軟體工具只能以 Server 為單位來進行 Backup/Restore 或 Import/Export，所以若有人工操作遺漏，便會使 Production 系統無法正常執行。

透過以系統為角度之觀點，便能輕易得知系統所使用的資料庫有那些，而且也不受跨 Server 之影響，在部署的完整性上比單一 Server 更強，而部署系統之前置動作就是比對，它是系統決定要部署那些單元的依據。

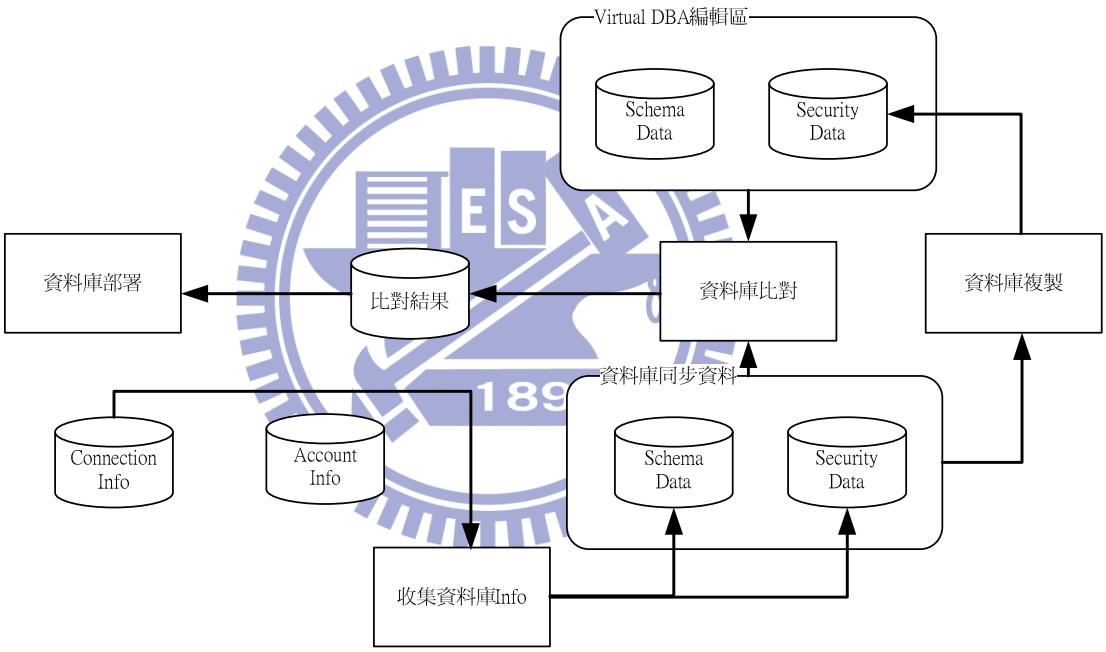


圖 3.7 資料庫物件比對機制流程圖

如圖 3.7 所示，在本系統的設計上，採用編輯區及資料庫同步區來區別控系統管值與實際值，也是資料庫比對的兩個重要來源，透過這樣的機制可以完全掌握其差異性，使部署的錯誤減到最低。

3.5 系統作業流程全覽圖

本小節主要說明 Virtual DBA 之所有功能流程說明，如圖 3.8 所示。

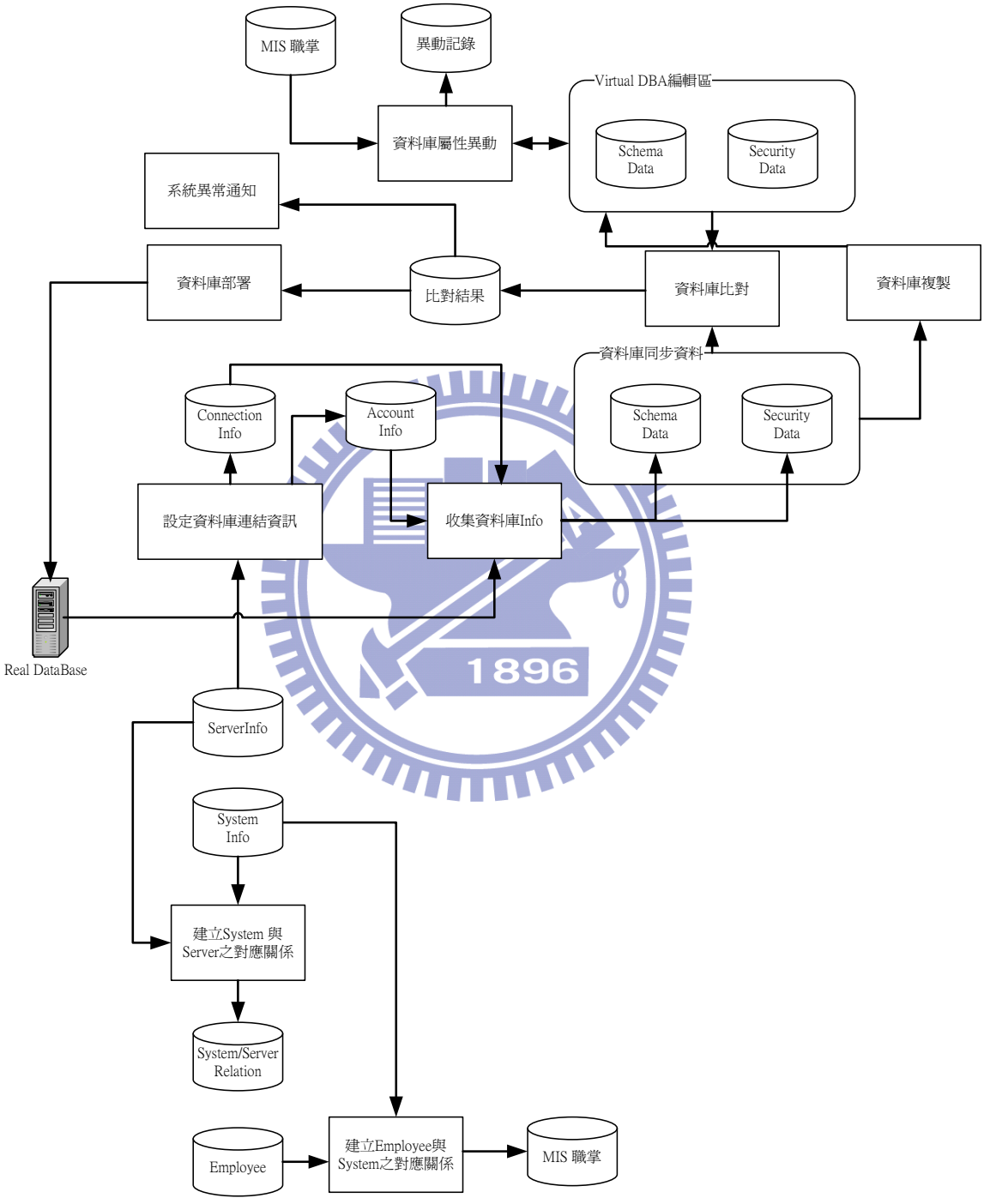


圖 3.8 系統功能流程圖全覽圖

第四章、系統實作與建置

本章節根據前章節所提出之系統架構設計方法，進行系統實作來驗證系統之可行性，並詳細介紹系統實作所使用的開發平台、系統架構、程式架構、功能模組以及實際導入使用的步驟與本系統的功能限制。

4.1 開發平台及語言

- **Microsoft Visual Studio 2005 & C#.Net**

本系統之程式開發工具為Microsoft Visual Studio 2005，此為Microsoft支援至.Net Framework 2.0以上之開發平台，而本系統採用之程式語言為C# .Net，此語言在.Net Framework支援的多種語言中，對於物件導向設計的支援最多，而本系統是利用C#在.Net Framework 2.0平台上進行程式開發。

- **Microsoft SQL Server 2005**

本論文現階段研究範圍為Microsoft SQL Server，而在資料庫維護上，則是採用SQL Server 2005之管理介面來同時控管SQL Server 2000以SQL Server 2005。

4.2 系統架構

本系統架構主要包含 Task Management System 以及 Virtual DBA System，Task Management System 主要功能是控管人員職掌資料以及維護應用系統、Server 等相關資料，Virtual DBA System 則為企業 Schema 與人員權限的集中化管理。

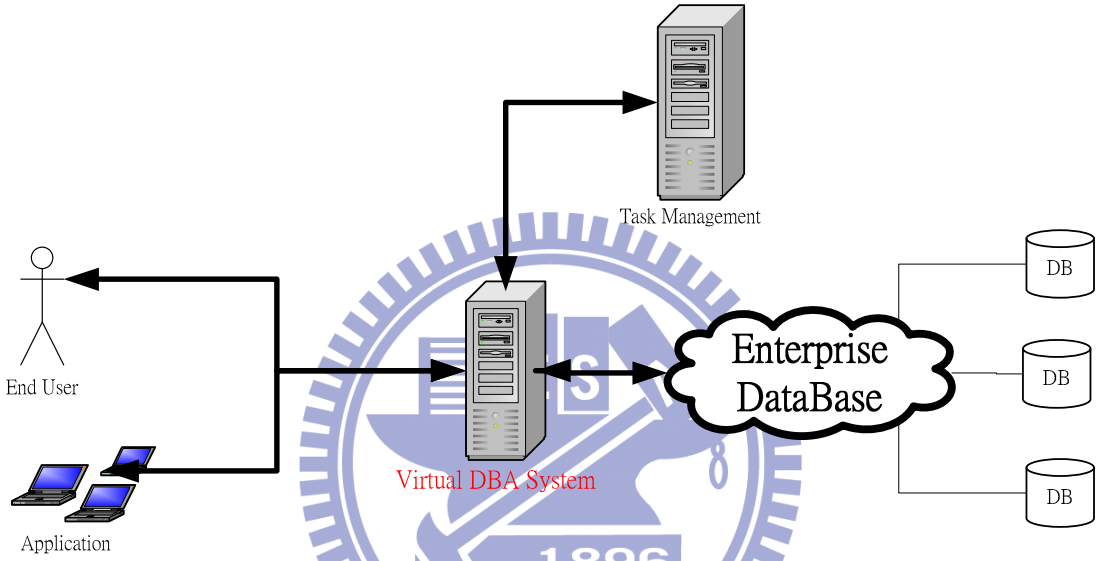


圖 4.1 Virtual DBA 系統架構圖

4.3 程式架構

本程式架構主要分為兩大部份，一為參考外部之 API Library，如 Microsoft Enterprise 以及 Microsoft Management Objects Library，另一部份則為自行開發之模組功能。

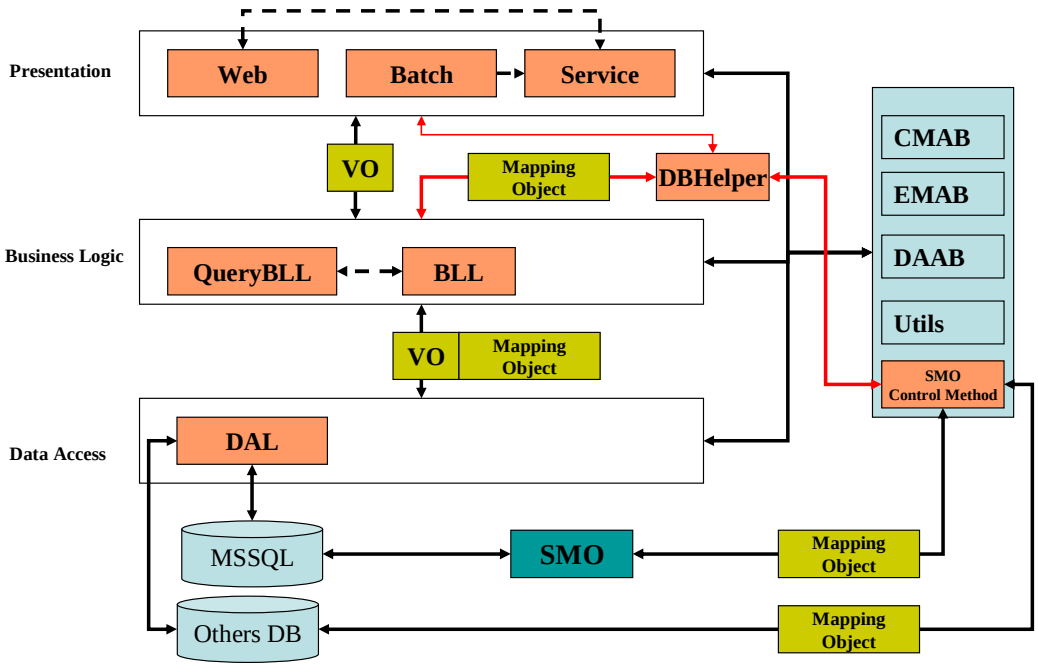


圖 4.2 Virtual DBA 程式模組架構圖

- **引用外部之 API Library**

本系統之程式架構三層式架構為主體如，其中引用 Microsoft Enterprise Library 之 Confiuration Management Application Block(CMAB)、Exception Management Application Block(EMAB)以及 Data Access Application Block(DAAB)等三個 Library，主要用來控制參數設定、程式錯誤控制以及資料連結的共用程

式，與 SQL Server 底層 API 聯結則是引用 SQL Server Management Objects(SMO)之物件模組。

- **自行開發模組**

1. Web/Batch/Service

本架構在 Presentation Layer 中主要有三種介面，Web 是操作管理網頁，Batch 為後端資料收集以及其它操作程式，Service 則提供對外介面供本系統或其它系統使用

2. QueryBLL/BLL

本架構之 Business Logic Layer 中主要有兩大模組，QueryBLL 主要負責與 DataBase Transaction(COM+)無關之資料的取得，BLL 則負責需要用到 Transaction(COM+)之 SQL Statement 操作(如 Insert/Update/Delete)。

3. DAO/VO

本架構之 Data Access Layer 中主要有兩大模組，DAO 是負責直接與資料庫連得並執行 SQL Statement 之模組，而 VO 則是利用 O/R Mapping 產生之資料基礎元件(通常相對於 DataBase 之 Table 結構)，主要用來當做跨 Layer 之資料傳輸。

4. DBHelper

本架構之另一個與資料庫連結之模組，上述提及之 DAO 是透過 Microsoft COM+之 Transaction 控管，而另一個直接操作 SMO API 之模組為 DBHelper，主要用來進行 DDL(Data Definition Language)之欄位定義或資料物件之建立修改。

5. Mapping Object/Control Method

Mapping Object 主要用來整合 SMO 與 DAO 之溝通物件類別，而 Control Method 則為實施操作之定義(並包含對於異質資料庫之存取) 。

4.4 系統功能模組

本小節主要說明本系統各功能模組之簡單說明。

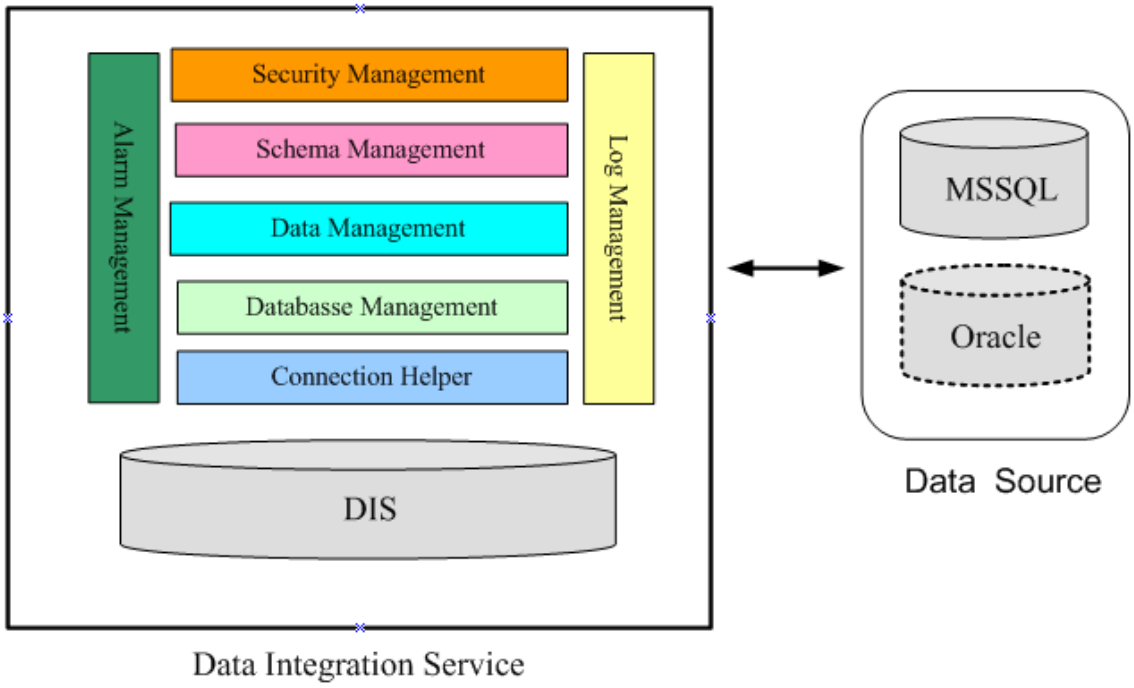


圖 4.3 Virtual DBA 系統功能模組圖

- **Security Management**

本功能主要負責控管資料庫使用人員的帳號增刪、變更帳號密碼、變更對於 DataBase Schema 的存取權限，以及控制，系統畫面如圖 4.4、圖 4.5 以及圖 4.6 所示。

1. 設定帳號

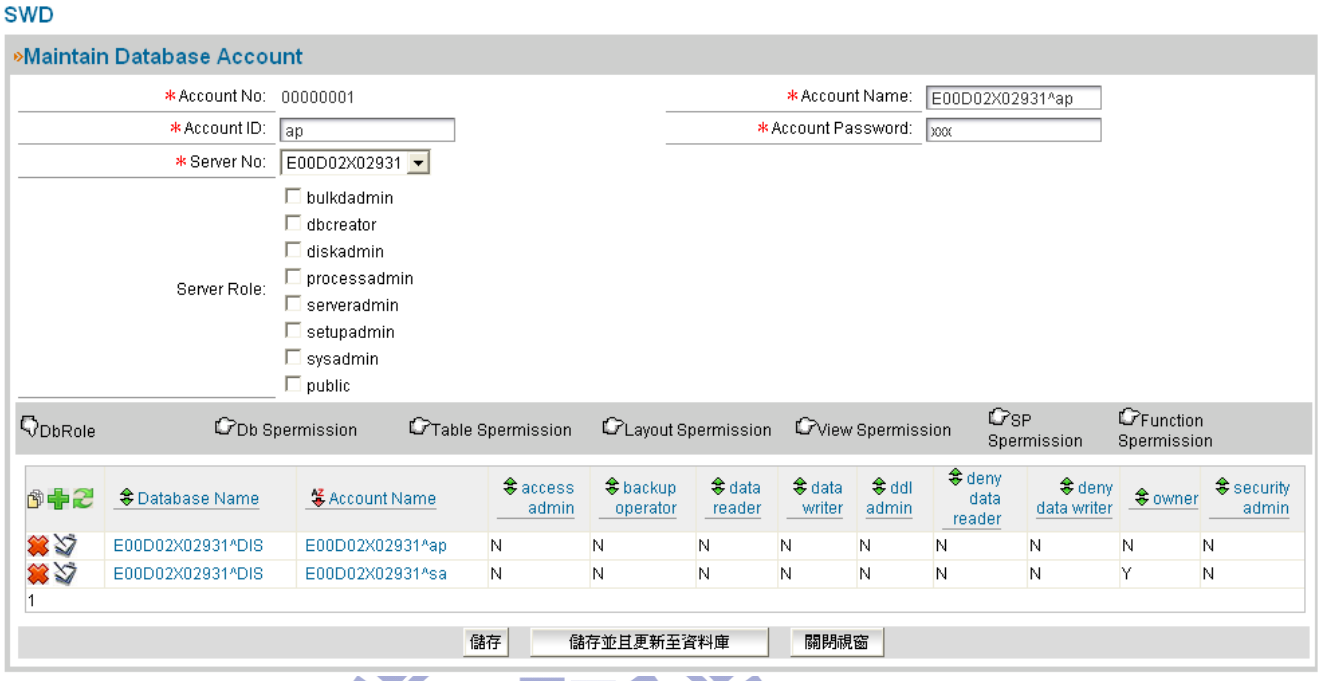


圖 4.4 帳號維護畫面

2. 設定權限

圖 4.5 權限設定畫面

3. 因應職務異動調整資料庫存取權限

在Task Management系統中調整系統的Sustain及Agent人員。DIS系統依據系統sustain人員資料決定該員是否對Table的有修改權限。可修改範圍：該員所sustain系統所own的Table的Table Schema。

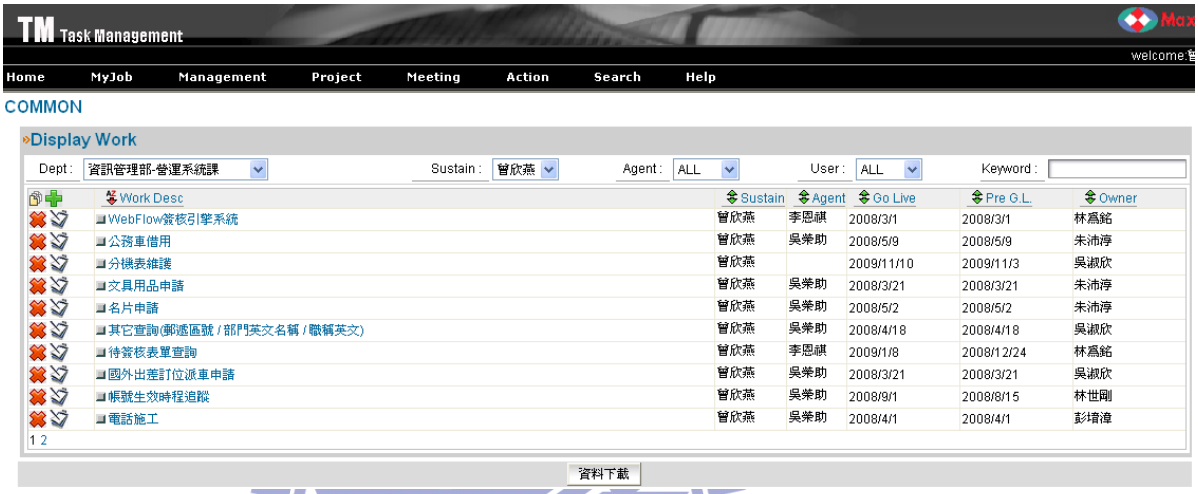


圖 4.6 職掌異動設定畫面

- **Schema Management**

本功能主要負責為同步收集不同資料庫之 Schema 功能，將其集中化管理，方便資料庫管理人員或系統開發人員快速得知企業內資料來源之相關資訊。

1. 同步收集 Schema

本功能同步收集功能區分為以 DataBase Server 為角度或是以 System 為角度來收集資料庫權限及 Schema，如圖 4.7 所示操作，根據設定的收集條件，會產生自動化同步 Job，後端程式會自動將相關資訊收集至 Virtual DBA 系統。

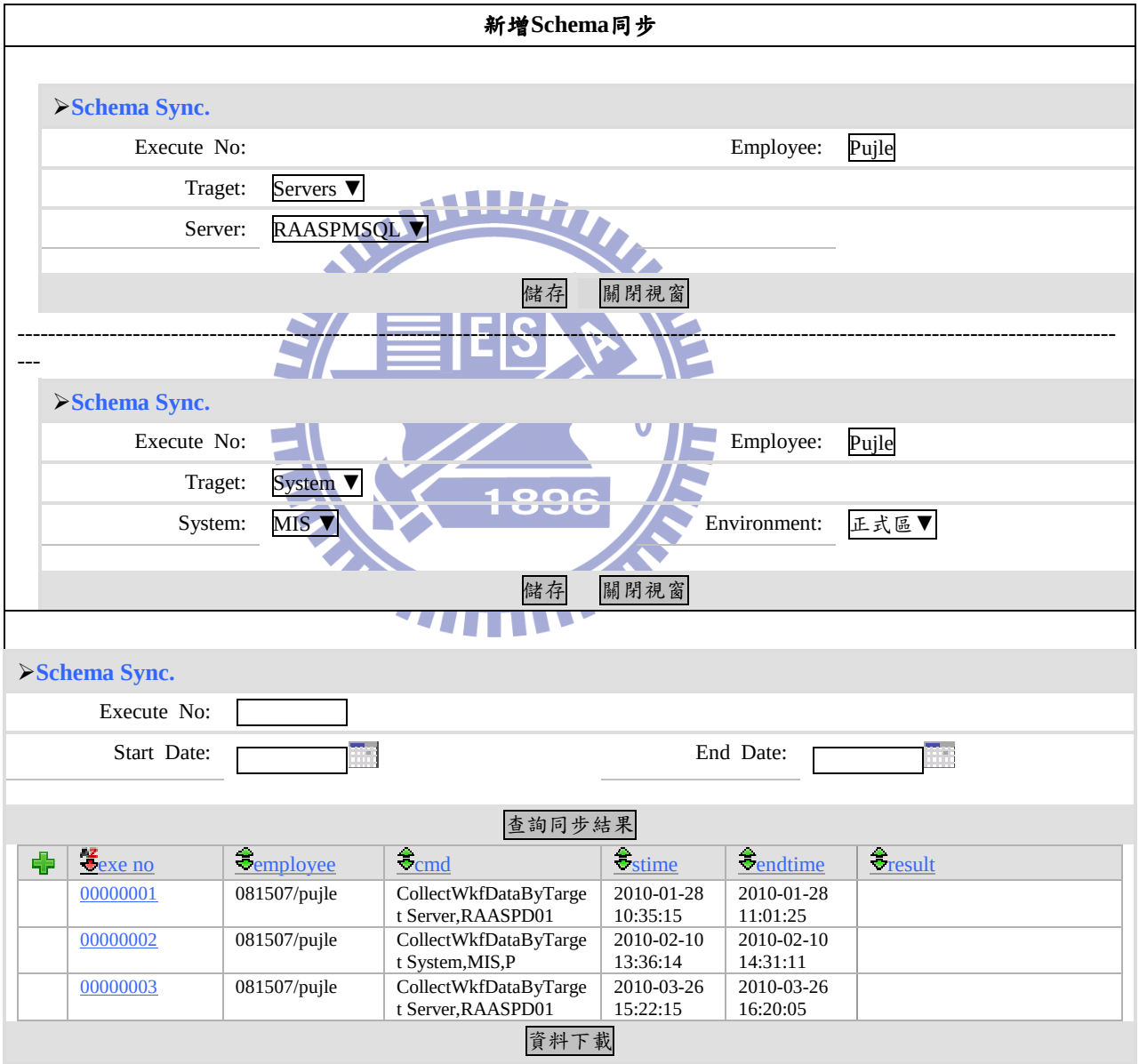


圖 4.7 Schema 同步收集功能畫面

- **DataBase Management**

主要功能為資料庫維護操作之機置，包含收集資料庫物件及帳號權限、比對、部署(包含建立及刪除)以及複製資料庫物件。

- **Alarm Management**

主要功能為主動偵測異常以及傳送 Email 通知給相關系統維護人員之機制。

- **Log Management**

本功能主要記錄所有資料庫操作行為之機制，以方便追蹤所有對於資料庫操作之正確或錯誤之動作，如圖 4.8 所示為 Log 之檢視畫面。

圖 4.8 系統操作 Log 資訊

4.5 系統限制

本小節主要說明在研究設計本系統架構時，針對實際企業之情況以及需求，對於本論文研究設計方向所規範的範圍以及目前無法實作之功能設計。

- 本系統目前可控管之平台為Microsot SQL Server 2000/2005，其它平台則無監

控管理功能。

- 本系統目前只針對資料庫物件定義(Table/View….)以及相關資料庫權限進行管理，對於資料的異動與檢視等安全性，尚未支援。

- 本系統只針對資料庫軟體進行管理，其它硬體或作業系統之管理並不包含。

4.6 系統導入步驟

本小節主要說明要將單一系統之資料庫納入本系統控管，相關系統以及人員管理上所必須實作的步驟說明。

- **建立應用系統與資料庫關聯**

本階段之工作為建立應用系統與資料庫 Server 之相關資訊，會由資料庫的管理人員去設定連線相關的屬性以及資料庫中之最大權限管理者帳號，使 Virtual DBA 系統具有管理者身份去控管該資料庫。

- **建立應用程式與員工職掌資料**

本階段之工作為建立系統維護人員所負責的系統，如此才能使人員具有操作權限。

- **利用系統之管理帳號收集資料庫相關資訊**

本階段之工作是將該資料庫之所有物件資料同步至 Virtual DBA 系統，也等同開始監控該資料庫。

- **建立跨系統資料關聯資訊**

本階段之工作是設定已經收集至 Virtual DBA 的資訊中，那些資料物件是被該系統或資料庫所參考。

- **回收相關開發者系統管理權限，並更改原先使用帳號之密碼**

本階段之工作是若更改原先設定之最大權限帳號之密碼，以使得原先之資料庫管理人員無法再利用此管理帳號來更動資料庫之物件屬性，而帳號之密碼將交由權責主管保管並定期更動。

第五章、結論與未來展望

本章將說明Virtual DBA系統對於研究對象產生之效益，以及本論文研究後續可再加強或精進的部份。

5.1 結論

- 將公司之資料庫來源資訊集中化，相關的資料定義與權限也受到控制，使資料庫管理人員可以更方便及完全掌握公司內部資料來源，而系統開發人員則可快速找至系統相關資料來源之所在，增加應用系統開發之速度，並可減少開發時間及錯誤發生。

- 傳統方式進行複製或災難復原，大部份會從維護記錄來進行操作，公司內部雖有訂定災難復原SLA(Service Level Agreement)為四至六小時，但若計算前置動作，可能需八至十小時，且不包含人工手動之操作錯誤所增加之時間，若以自動化方式進行複製或災難復原，由於所有資訊已集中控管，故操作含驗證時間，單一系統之復原時間，已可降低到一個小時之內完成。

- 人工方式進行資料庫之異動，比較無法清楚記錄相關資訊，所以一但有異動的錯誤發生，首先在解決錯誤之問題尋找上，便要花費相當多的時間，且容易被人為誤判，造成錯誤的處理以致系統發生更大的錯誤。以自動化方式進行資料欄位或權限異動，所有異動之操作行為皆會被完全記錄，且任何資料庫的異動，也會立即告知相關資料庫來源之負責人進行同步動作，除了提升了系統的穩定度，對於系統功能修改的限制，其阻礙也會降到最低。

- 由於明確記錄維護人員與應用系統之職掌關係，一但有人員異動，只要更新

應用系統負責人資訊，在系統權限上之交接便已完成，交接之事宜便可著重在程式或domain know how的之上，除了縮短交接時間，也確保系統權限的交接完整性，阻斷交接不完整之資安問題。

5.2 未來發展

本小節主要說明針對本論文研究設計的限制中，仍可再加強或延伸研究範圍以及系統之功能討論。

- **增加其它平台資料庫軟體(如Oracle，Informix，DB2)之管理支援**

本論文研究因考量不同資料庫平台其管理方法及軟體架構差異太大，目前僅針對 Microsoft SQL Server 進行管理，但企業內部仍有部份系統使用非SQL Server之平台，後續之發展應再將其它平台(如Oracle，DB2)納入管理，才能完全達到管理之最大綜效，且若能整合不同平台之資料庫在統一管理介面上，也能增加企業內部資訊系統的彈性以及延展性。

- **增加對於資料以及資料安全之控管**

資料安全是任何企業，特別是高科技產業相當重視的問題，本論文研究目前尚未考量資料的控管，若僅就資料庫物件及權限之控管，其效益只能侷限在企業之資訊單位，若能對於資料的修改與閱讀進行規範，對於企業智慧財產權的保護將有很大的功效，且是企業穩定維持競爭優勢的一大保障。

- **增加Server上非資料庫之軟硬體環境之控管**

DBA的維護工作是包含資料庫以及週邊環境的管理，資料庫的效能以及穩定度有時也會受到週邊軟硬體的情況而影響，所以後續若能監控 Server上的健康情況，如CPU 的使用程度或是硬碟空間的使用，且能在遠端透過系統進行操作或調校，如此更能滿足資料庫管理人員日常的維護工作。

第六章、參考文獻

[1] 程鼎元，以MVC設計樣式為基礎的應用程式產生器，碩士論文，2005

[2] 王證宜，元件式應用系統產生器：應用系統建構方法論，碩士論文，2003

[3] 張益嘉，應用系統產生器：之架構與資訊系統塑模方法論，碩士論文，2001

[4] 劉明修，以夥伴關係管理概念建構一SOA之電子化採購系統，碩士論文，2008。

[5] 吳國樑，資料庫線上核發系統規劃研究-以C公司為例，碩士論文，2008。

[6] 陳建樺，整合物件導向設計與關聯式資料庫之輔助環境，碩士論文，2005

[7] 網路資訊雜誌,2008,“資料庫市場競爭激烈 Oracle市佔持續領先”
http://news.networkmagazine.com.tw/software-application/sftwnews/2008/04/30

[8] Microsoft Corporation，透過 Microsoft 平台啟用“真實世界的服務導向架構(SOA)”，2006/12。

[9] 林慶德、陳宇芬 譯，「資料庫管理與應用」，台北，台灣培生教育出版股份 有限公司，2006。

[10] 劉金順 譯，「資料庫管理:設計及其應用」，台北，美商麥格羅希爾 台灣分公司，2004。

[11] Microsoft SQL Server 資料庫管理員的角色和任務 Microsoft Windows 2000 平台http://www.gzu521.com/campus/Manual/SQLServer2000/3.htm

[12] ADO.Net 技術及 O/R Mapping 演進，

http://blog.sina.com.tw/dotnet/article.php?pbgid=4907&entryid=581724

[13] 程式產生器，

http://blog.blueshop.com.tw/dplayerd/archive/2009/01/16/57989.aspx

[14] 分散式應用程式架構，Microsoft MSDN WebSite，

http://msdn.microsoft.com/zh-tw/library/aa291882(VS.71).aspx

[15] Jen-Her Wu; Tse-Chih Hsia; I-Chia Chang; Sun-Jen Tsai; Application generator: a framework and methodology for IS construction ,System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on , 6-9 Jan. 2003 Pages:263 – 272.

[16] Sommerville, I., Software Engineering, Massachusetts: Addison-Wesley, 2000.

[17] SMO Overview，

http://msdn.microsoft.com/zh-tw/library/ms162557(v=SQL.105).aspx

[18] Object Relation Mapping，Wikipededia.com，

http://en.wikipedia.org/wiki/Object-relational_mapping

[19] Microsoft Enterprise Application Block

http://msdn.microsoft.com/zh-tw/magazine/cc188689.aspx

[20] Web Service Architecture

http://onjava.com/pub/a/onjava/2005/05/25/j2ee-services.html