私立東海大學資訊工程與科學研究所

碩 士 論 文

指導教授：朱正忠 博士

以 SOA 服務導向架構重整電子化金融平台

Service Oriented Architecture Based
Re-engineering for Finance E-Commerce Platform

研 究 生：陳榮信

中 華 民 國 九 十 七 年 七 月 七 日

私立東海大學資訊工程與科學研究所

碩 士 論 文

以 SOA 服務導向架構重整電子化金融平台

Service Oriented Architecture Based Re-engineering for Finance E-Commerce Platform

研 究 生：陳榮信

本論文業經審查及口試合格，特此證明

口試委員簽名　　指導教授簽名

_____ _____

_____ _____

_____ _____

中 華 民 國 九 十 七 年 七 月 七 日

摘　要

隨著資訊科技的日新月異，銀行的金融服務也漸漸的從客戶的臨櫃交易，轉化成網路化的資訊交易服務，進而改變了客戶的使用習慣，電子商務逐漸取代了原有的交易模式。網路化的興起更加速了客戶對電子商務的需求，而在這快速成長的需求中，銀行業這個龐大的金融體系，如何能跟上時代的腳步？如何能即時的符合客戶千變萬化的需求？然而在上述的條件下，又如何能節省資訊成本，便是一門重要的課題。

過去的銀行均著重於臨櫃的交易，隨著客戶使用的習慣的不同，銀行業也漸漸發展出各式各樣的電子化的金融交易介面，但也因每一時期的業務需求的不同，產生了許多功能相近的交易系統，而要如何維護這些系統且又能快速的滿足使用者新的需求，不至於重覆開發消耗人力，實是資訊部門主管(CIO)應相當關切的問題。

本篇論文將以銀行業中之電子化金融平台作為範例，設計出以服務導向架構(SOA) 為架構方法及以網路服務(Web Service)為服務標準介面之整合式電子商務金融服務平台（Financial E-Commerce Integration Platform，FECIP)，試圖在FECIP 架構下，找出一個可以符會現代金融業的開發架構，並可解決重覆開發，達到重覆利用、快速整合的系統開發模式。

關鍵字：服務導向架構、網路服務、網路銀行、重整工程。

Abstract

Financial banking services have gradually changed from brick-and-mortar services into internet transaction services, following the renovation of information technology. This also has affected how customers dealing with banks. Therefore , emerging E-Commerce has taken over the traditional transaction method gradually. To keep up with the growing demands from customers and satisfy their various needs with cost-saving IT development strategy has become important issues for financial industries.

Banking services used to focus on doing all transactions face-to-face; however , they are developing various electronic financial transaction interfaces to meet the current business needs. Nevertheless, there has been a lot of transaction systems with similar functions which are generated in order to cope with business requirements in every stage. How to maintain those systems and to swiftly meet customers' new demands without wasting manpower would be an important issue for managers in IT departments.

This article will use the electronic financial platform in the banking industry as an example to try to develop a framework of service-oriented architecture(SOA) and a financial E-commerce Integration Platform(FECIP) based on the Web Service as the standard service interface. Under this framework, it attempts to develop a model of fast integration and reusability that fulfill the need of financial industry at the new electronic age.

Key Words： SOA, Web Service, Network Banking, Re-engineering .

誌　謝

塵勞迴脫事非常，緊把繩頭做一場；不經一番寒徹骨，焉得梅花撲鼻香。自接觸碩士學分班開始，進而考進東海的碩士班，這幾年取得碩士的生涯中，心中似乎總是有一塊大石始終在心中壓抑著。從隨性考試後差一名的衝擊，履次的公司變故，直至找到一家 Strong 的公司，卻無法擺脫諸多的障礙，銀行的合併離開了熟悉的台中，責任的加深，再再的考驗著這段研究生的生涯。

首先要感謝的是指導教授朱正忠老師在這一路走來的研究所生涯中的教導，且體諒在職生的辛苦與無奈。同時感謝口試委員王豐堅教授、 楊朝棟教授、黃悅明教授及熊博安教授，有教授你們寶貴的意見，讓學生獲益匪淺，讓論文的內容更加的充實。

感謝婉婷刺激我不服輸的衝勁，雖造就了壓抑但如今終有所成，感謝淑姿引領我結識東海；感謝Dr. Ryan 賦予的知識前瞻性；感謝同窗好友兼現在的長官明富；班上最美麗的牙醫姿秀，在大家用功之餘為我們準備美食；介棟、志煌、海燕、淑惠大姐等既是同學又是學長也是學弟的朋友們，回首學分班時的考驗心中有苦卻也甜在心裡，感謝這段期間你們所給予的協助與叮嚀。

在台中台北的二地奔波中感謝玲雪默默的陪伴，感謝姐姐在我忙碌的這幾年中一直扮演著家裡的管家，幫我分憂，感謝父母親讓我生活在這無憂的家庭，還有我們家的關公更是我的心靈導師，做我強有力的後盾，讓我能堅持的走過來。謝謝資訊技術實驗室的學弟妹們在口試期間的連絡協調，謝謝所有的有緣人。

最後，謹以此篇論文，感謝所有幫助過我、愛我的家人朋友們。

東海大學資訊技術實驗室　陳榮信

目　次

圖目錄

表目錄

第 1 章. 緒 論

1.1. 研究動機

自網際網路流行以後，銀行採用 Web 模式開發應用程式的情形日益普遍，大型主機(Mainframe)式的服務模式已不能符合現今使用者的需求。因銀行內部開發及需求時間委外廠商的不同，以及同價格考量和各系統整合的技術門檻，使得許多金融相關的網站仍只採用多個 web 應用程式來建構其電子金融服務。放眼現今各家銀行在網路上的資訊服務系統，大致上不外乎是網路銀行、網路ATM、企業金融網…等系統。而上述這三個系統由於業務需求的時機和委外建置的廠商及方法架構不同，使得銀行經常面臨到擁有三套以上功能相近卻又不能相容的系統，如此不僅對系統的維護及人力成本造成了負擔，且在面對新的需求時，更是需耗費三倍的人力、三倍的時間、以及三倍的成本才能完成。

以上述三套資訊服務系統為例，在此三套系統中，為方便銀行客戶能夠方便查詢其帳戶的交易明細資料，均包含了【交易明細查詢】的功能，然而因不同時期、不同廠商、不同標準之下建立出來的明細查詢，在三套系統中，分別重覆開發了三次。若有一天在需求的改變之下，需要變更明細查詢的技巧內容時，則勢必需要消耗三倍的人力與工時。

然而，電子商務日益蓬勃，網站內容與服務也日益多元，銀行在電子化銀行的需求及觀念，也不再只是單單局限於被定位為銀行提供客戶轉帳工具或服務的其中一個選項，為了別家有網路銀行，我們銀行也要有置；別家有 WEB ATM ，所以我們亦不能輸人的觀念而已。然而，隨著網路商機在 2000 年泡沫化後的甦醒、安全技術的提昇及實體通路市場的飽和，使得金融競爭的戰場逐漸延燒到虛

擬通路上。各大金控正投入龐大的資金與人力改造、重整現有的網路銀行系統，銀行漸漸朝向的是建構一個整合式的金融平台，它將不再只是以往銀行櫃台電子化、網路化，而已，而是一個能夠整合金控內、證?、投信、人壽與異業結盟、子公司橫向交差行銷、數個業務垂直結合行銷及合作企業之電子商務金融平台，以在虛擬通路佔有一席之地，增加競爭力。

然而頂著多變化的業務行銷企業整合的電子商務金融平台的光環，銀行資訊系統如何才能在系統擴充不易的現況又能快速迎合市場及業務拓展的需求量及其效能，更是資訊部門的一大課題。

然而搭著科技進步的順風車，隨著網路的 SOA（Service Oriented Architecture）技術又日益成熟的情況之下，以 SOA 架構來做為電子商務金融平台，已成銀行提供一個高效率又不易中斷服務的絕佳方案。因此本論文的研究動機在了解以 SOA 架構在電子金融服務網站開發上的一系列技術，並透過軟體工程的技術重整系統而得到一個高維護性的服導向軟體系統，以提昇效能、降低升級成本。

1.2. 研究目的

鑑於上述的動機，本文論文將針對此一課題進行探討，其研究目的如下：

1. 提出一套以服務導向架構（SOA）為標準程序之方法以重整、轉換舊有的網站應用程式系統，至以服務為基礎的多層式服務導向架構系統，以提昇系統功能、彈性，減低維護及新系統開發之成本。

2. 研究 SOA 服務元件的相關技術，以建立一個易於維護、擴充、再使用的可延展性系統。

3. 嘗試以服務導向架構（SOA）為基礎，建立一套排除上述銀行面臨的問題之系統架構與開發模式建立銀行中 WEB 資訊系統的標準開發流程（SDLC）。

1.3. 論文章節概要

本論文之架構與各章節概述如下：

第一章　緒論：介紹本研究背景、動機與目的。

第二章　背景知識與理論探討：

探討與彙整國內之電子銀行種類及現行網路技術：服務導向架構、網路服務、企業流程管理 MVC、RIA 等分析技術作概要的簡述。

第三章　個案探討：

針對本研究所要探討的 S 銀行網路金融服務平台的現狀，S 銀行面臨的挑戰進行分析與探討。

第四章　實作規劃與建議：

(1). 對於網路金融服務平台的建構模式提出彈性及易於整合開發的架構模式提出規劃與建議。

(2). 提供與建議銀行使用的網路金融服務平台的安全模式。

第五章　結論與建議：

本研究結論與未來研究方向。

第 2 章.　背景知識與理論探討

2.1.　網路金融服務平台的種類

提及網路金融服務平台直覺的我們便會連接到幾個名詞「電子銀行」「網路銀行」..等等，然究竟「電子銀行」、「網路銀行」、「網路金融服務平台」、「金融電子商務平台」…等，是否代表同一事物，而其關係又為何呢！

銀行由櫃檯櫃員的服務機制，漸漸朝向電子化發展，也產生了許多不同方式的服務管道，電子銀行(Electronic Banking)就此行形成，其中包括自動櫃員機系統 (ATM) 、電話銀行(Phone Banking)、無人銀行，而常見的網路銀行亦是屬於電子銀行銀行的一支，然而什麼是「網路銀行」？依據巴賽爾銀行監理委員會(Basle Committee on Banking Supervision 1998)的定義：「網路銀行業務是指經由電子通路(Electronic channels)提供金融商品及服務」。另外在銀行公會公佈的「個人電腦銀行業務及網路銀行業務服務契約範本」第二條名詞定義第二項中，則指出「網路銀行業務」（Network Banking）：指客戶端電腦經由網際網路與銀行電腦連線，無須親赴銀行櫃台，即可直接取得銀行所提供之各項金融服務。而現今的網路銀行，依財金公司業務別[02]，安全機制及各家銀行的拓展方式不同而發展出許多延伸性的網路銀行。

2.1.1.　金融電子資料交換系統(FEDI)

想要瞭解 FEDI，首先就會觸及到什麼是 EDI，EDI (Electronic Data Interchange) 是一種訊息的格式，仍係指貿易伙伴間業務往來之資料，以標準格式(採用聯合國 UN/EDIFACT 所制定的標準訊息)經由電腦應用系統對電腦應用

系統(AP to AP)模式直接做訊息的交換。

而什麼又是 FEDI 呢!財金公司對 FEDI 的簡述是：金融電子資料交換(Financial EDI)，簡稱金融 EDI、FEDI，係指企業或個人利用電腦作業，以特定的標準格式，經由通訊網路與金融機構連線，進行企業或個人之付款、資金調撥及轉帳等金融服務。

FEDI 通常運用在提供企業財務作業與銀行之間的溝通橋樑，透過金融 EDI，企業財務系統可連結銀行進行各項如支付、資金調撥等財務作業。並補足企業在使用網路銀行上的不足。

2.1.2. 企業金融網(FXML)

「金融 XML 業務」(FMXL 業務)仍財政部在 2000 年基於網際網路開放性與電子資料交換作業,督促銀行公會成立 XML 訊息訂定小組[08],負責金融 XML（FXML）訂定的訊息標準，主要是提供用戶利用金融機構提供之 FXML 金融服務平台，在其個人電腦或相關設備上發動交易，透過網際網路以 FXML 之訊息格式傳送給金融機構，辦理資金調撥、傳送和付款相關的各種指示及資訊查詢、線上融資等各項金融服務，凡參加 FXML 金融服務之金融機構皆可彼此執行廿四小時，全年無休金融服務。

2.1.3. Web ATM

什麼是「Web ATM」？簡單來說就是網路上的 ATM 系統，只要在個人電腦上搭配晶片讀卡機,便能將 PC 模擬成擁有銀行的實體 ATM 九成的 PC 版 ATM ，並能在網際網路上執行包括餘額查詢、自行或跨行轉帳、約定或非約定帳戶轉帳等各項金融服務。除了無法吐出現鈔外，功用上與實體 ATM 相差無機。

2.1.4. 多重服務介面

由圖(1) 我們可以明顯看的出上一章節所提到的金融服務平台中，其交易的介面是以網際網路作為其交易的通路，而當一個以網際網路作為通道的系統，其終端所銜接的設備，更是無限的多樣性。而傳統電子金融服務系統，均會針對每一個 End User 所使用的介面不同而建置不同介面但功能相仿的系統。也因如此的循環建置下，造成與後端中心主機連接的 Gateway 愈接愈多，也愈來愈錯綜複雜，造就了多重服務介面與錯綜複雜的後端連接架構的窘境。

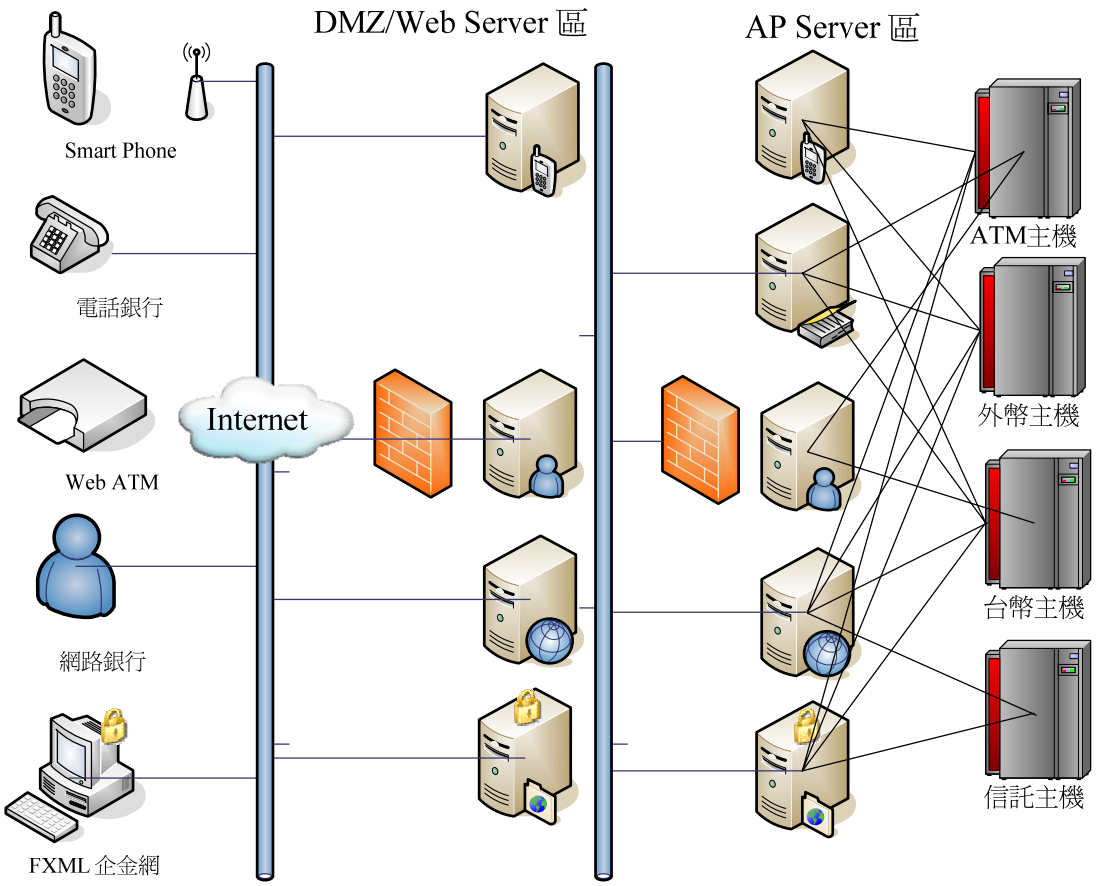


圖 1 傳統電子金融服務架構

2.2. 服務導向架構(SOA)

Service-Oriented 的 Service 指的就是 Web Service .W3C 對於 SOA 的定義是[10]：[一組可以被呼叫的元件，其介面描述資訊可以被發佈與探索](A set of components which can be invoked ， and whose interface description can be published and discovered)。SOA 的特色與精髓在於(完美的協調各種網路服務(Web Service)快速的定義與規範出企業所需的各種資料處理流程)

服務導向架構（Service Oriented Architecture； SOA）[04] [05]是一種建立應用程式的方法，這種方法將所有的元件都當作可重覆使用的服務。SOA 會指定一個共用介面，程式開發人員便可透過此介面來存取這些服務，將多個既有應用程式的各部分結合起來，以快速建置新的應用程式。而一個好的 SOA 方法中，須讓所有的元件都會顯示為可重覆使用的服務，而非只有 Web 服務而已，就原有的系統資源無需先將其轉換為 Web 服務。

2.3. 網路服務 Web Services

2.3.1. Web service 概述

資訊系統的演進，由大型主機(Main Frame)的時代開始逐漸走向分散式的系統架構，由一台 Server 管理一群 Client，進階至多 server 間相互溝通，讓原本性質不同的資訊系統及異質系統間彼此整合，因此一個系統架構如何部署溝通分配，便成了一個重要的課題。

在分散式系統的架構上許多的學者大力提倡 Web Services 的概念，而每個人也都詮釋了不同的 Web Services 新解。而回歸到最原點，到底什麼是 Web

Services？Web Services 最簡單的定義是什麼？在此引述 W3C 對 Web Services 的定義[03]：(A Web service is a software system identified by a URI， whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition， using XML based messages conveyed by internet protocols.)由此可知 Web Services 簡單的來說就是一種透過網路協定以 XML 為資料格式作為資訊傳遞交換的一種方法，而 Web Services 又是如何運作的呢！

Web Services 的組成是以 XML、WSDL、SOAP、UDDI 基礎，以 XML 格式為基準將資料轉變為 Web Services 的資料，利用 WSDL 描述將服務的對象做一個描述，使另一端可以透過這一個描述，解譯所得的資料。以 SOAP 通訊底層，進行傳送的動作，向 UDDI 進行搜尋或是註冊動作。WSDL、SOAP 與 UDDI 皆是用 XML 方法來描述，圖(2)是 Web Service 的運作架構。

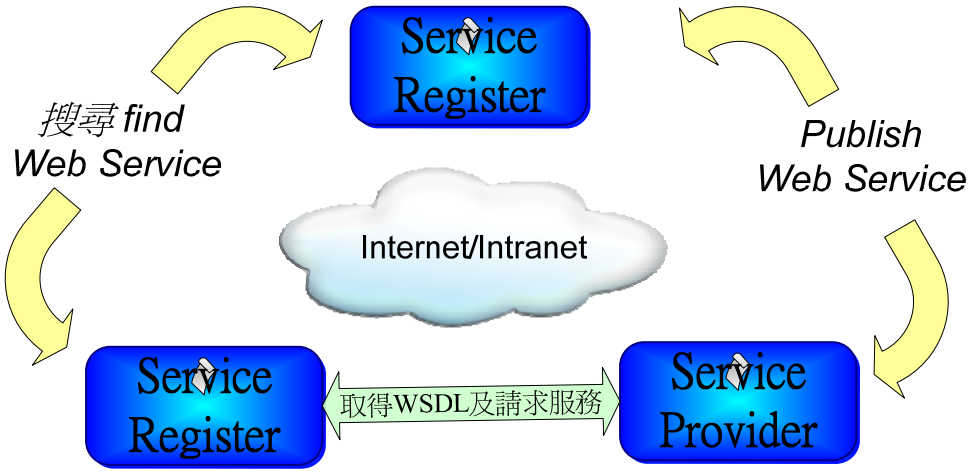


圖 2 Web Service 的訊息架構

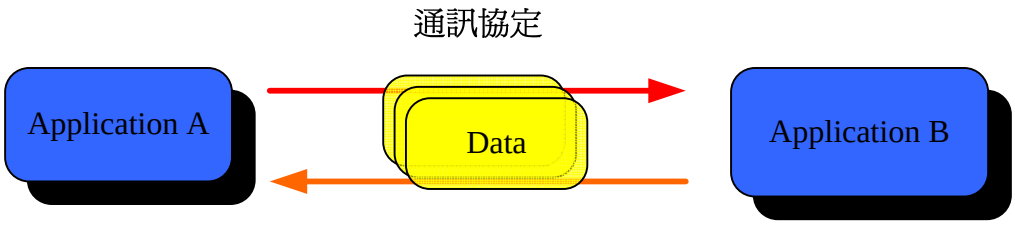
主要由三大區塊組成，分別是

・Service Provider ： 提供服務及服務本身的執行環境。

・Service Register ： 是一種儲存 Web Service 資訊的環境，讓 Service Prov 註冊公佈 Service 的資訊，讓 Service Requester 搜尋服務，並取得和 Web Service 溝通的相關資訊。

・Service Requester： 某種 Client 或應用程式，在 Internet 上搜尋、使用 Web Service。

Web Service 網路服務功能相似於以往的分散式技術，是一種更寬鬆的（Loosely Coupled）分散式架構，以最簡潔的模式說明，Web Server 的運作原理概觀如圖 3 所示，就是二個各自獨立的應用程式系統（Application System）間傳遞訊息的方式。



Web Services 運作原理概觀

圖 3 Web Service

2.3.2. SOAP (Simple Object Access Protocol)

簡易物件存取協定(Simple Object Access Protocol；SOAP) 是 Web Services 用來傳遞資料的編碼格式，目前 W3C 已經制訂了 SOAP 1.2 的規格[10]，相關

標準可參考 http://www.w3.org/TR/soap/，SOAP 的目的在提供 RPC(遠端程序呼叫) 的功能，在 SOAP 中主要是以標準 XML 的方式來制訂一個訊息的格式，一個完整的 SOAP 封包中會使用 Header + Body 來標示訊息的各個部分內容，此外會使用 HTTP、SMTP 等網路通訊協定來進行資料的連結交換。請特別注意 SOAP 的標準和底層的網路通訊協定無關，因此只要根據 SOAP 規格產生出 SOAP 文件之後，接下來不論使用網路上任何通訊協定都可以進行 SOAP 文件的交換，只要雙方彼此都能根據所選擇的通訊協定交換資料，就可以彼此交談，這也正是 Web Service 突破平台、語言疆界的最佳利器。

2.3.3. WSDL (Web Services Description Language)

是用來描述 Web Service 提供者所提供的服務運作方式[11]，其中包含了服務要求者如何與服務提供者溝通的傳輸協定、格式及相關參數...等互動方式之抽述。而 WSDL 本身也是以 XML 語法所定義的一種語言。透過 WSDL、XML Web Service 用戶端就可以了解如何建構 SOAP 訊息來呼叫 XML Web Service 所提供的方法，或是藉由工具，根據 WSDL 來產生 Proxy 物件，利用 Remote Procedure Call 的方式建構 SOAP 訊息。

2.3.4. UDDI (Universal Description ,Discovery and Integration)

UDDI 是由 IBM、Microsoft 及 Ariba 等三家公司聯合制定的開放性平台架構為一個註冊以及儲存資料庫的標準[15]。其主要負責的工作為服務註冊與服務查詢，提供註冊與搜尋 Web Service 資訊的一個標準。其目的是讓網路上 Web Service 提供者可即時的傳遞服務訊息和內容給所需要的程式開發者以及企業使用者，即時取得在網際網路上分享的 Web Service。

2.4. BPM 企業流程管理

依據 Gartner Group 對【企業流程管理】(Business Process Management ; BPM)的定義: BPM 是指一套完整的工具與服務，除提供企業內部工作流程分析與資訊系統的整合之外，也包括企業外部交易夥伴的應用整合，以因應未來企業協同運作的需求。

BPM 提倡之初，卻進行的並不是那麼順利，然而隨著 Web Services 的興起，開始造就了 BPM 的風潮更是發揚服務導向架構(Service Oriented Architecture ; SOA)的一個重要驅動力，BPM 這個名詞開始伴隨著服務導向架構。企業應用整合(Enterprise Application Integration ; EAI)出現，成功的使企業藉由 BPM、SOA 、EAI 整合起多個異質的應用系統平台，建構出可以「隨需應變」的企業流程，使的系統整合更具彈性化。

2.5. BPEL 商業流程執行語言

商業流程執行語言(Business Process Execution Language；BPEL) [13][14]，是一種藉由 XML 格式來描述工作流程或商業流程的語言。透過 BPEL 來描述服務的組合及流程，無需考量各個服務的相依性，更可利用圖形化的介面，輕鬆的使非資訊技術人員也可以籍此圖形化介面訂定新的執行流程，快速的符合業務單位的需求，使系統更具彈性。

2.6. MVC 架構

Web 應用程式的建置隨著科技的發展已逐漸到了一個成熟的階段，現行最常見的 Web Model 即三層式架構(3-tier)。經由此種架構模式，使得使用者端的介面(User Interface)，與中間的應用程式層(Application Layer)以及最底層的資料庫(Database)能夠釐清彼此的功能領域，並且分層管理，不論在資源的重複使用或是管理便利上都有很好的作用。

上述模式雖好，但卻不能符合現今多變的商務需求，不管是後端的資料來源變動了，或是前端的使用者介面變動(如：修改成多國語言或 RIA 模式)，整個 Web 應用程式勢必要做一大幅度的更動，這樣所花費的人力資源及後續的維護成本對一個資訊單位，必定是一個沈重的負荷。

為了實踐一個高效能及高整合性的架構，在此將原本的三層式架構中使用者端的介面與應用程式層再加以細分成，Model-View-Controller (MVC) 架構。以下是對 MVC 三者的定義：

所謂的 MVC 模型是在建置 Web Application 時經常使用的一種模型，這個模型主要將一個 Web Application 區分為二大部份。分別為應用程式層(Application Layer)及使用者介面(User Interface)二大區塊，由這二大部份再依據 MVC 的架構細分成屬於應用程式層 (Application Layer)的 Controller 與 Model，屬於使用者介面(User Interface)的 View 部份，並各取其第一字母 簡稱為 MVC 架構。圖(4) 是 MVC 的架構圖及 MVC 三者的概述：

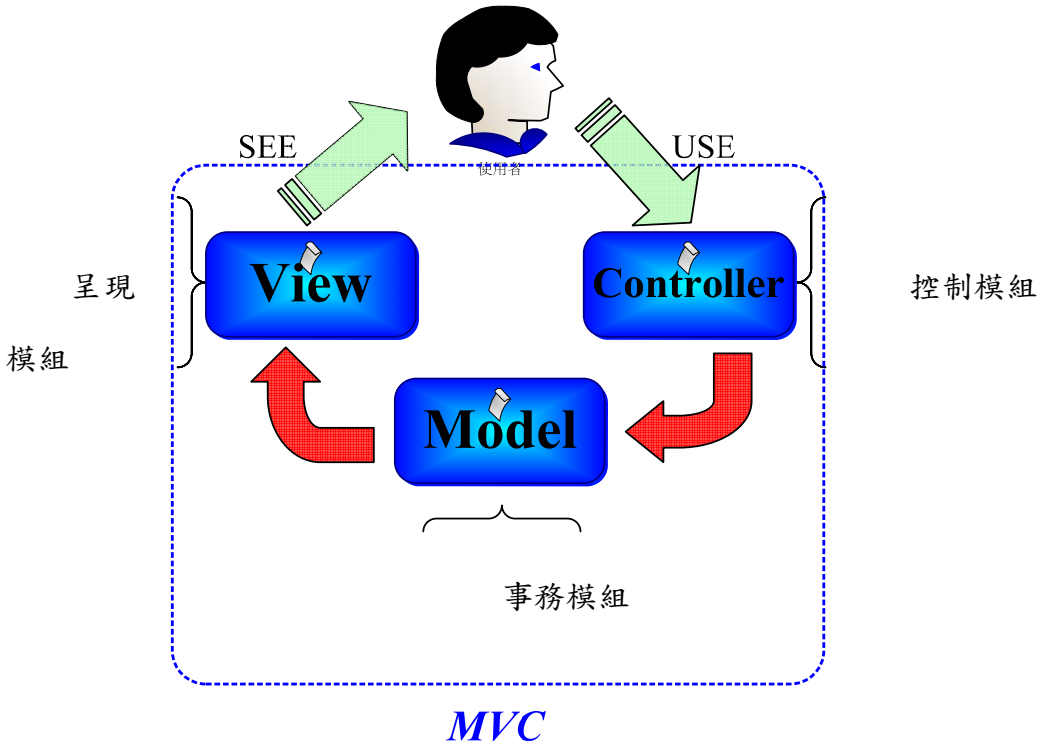


圖 4　Model-View-Controller (MVC)架構模型圖

M-Model(Application Layer)：管理及執行處理、轉換資料的請求(Request)及回覆(Response)。

V- View (User Interface)：負責陳現 Model 中的資料，以不同的視覺化模式展示 Model 模組中的資料。

C- Controller (Application Layer)：負責控制整個系統交易機制的流程模組。

MVC 也可說是設計樣式(Design Patterns)的一種，為了達到使用者介面和所處理的資料分離開來，因此在觀念上是將應用系統的資料本身、資料展示方式以及資料的操作方法等三項功能分開設計以達到不同元件群間相對牽連與關係的程度低，也就是偶合度低，使得系統在維護更為方便。這樣的設計最早是

SmallTalk 語言在 1980 年代所提出的觀念，主要是應用於建構高互動性的應用系統時所使用的良好方法。

呈如以上陳述對 MVC 的介紹，簡單的說 MVC 就是這樣把一個 Web Application 加以分割，使用 MVC 架構開發的優點是能使開發流程更明確，使用 MVC 方式開發系統可以完全切開顯示端 View 及商業邏輯的開發，程序的管控則交由 Control 掌控整個系統之流程，有效的使美工設計人員(UI)與程式開發人員可以專注於本身的工作。達到開發分工，提昇再利用的目的。

2.7. RIA (Rich Internet Application)

資訊系統由原本的大型主機系統與終端機的年代逐漸被「主從式架構」(Client-Server)的應用系統所取代，然而隨著網際網路的盛行，應用系統的開發模式又隨之轉向到以瀏覽器為主的應用系統。在圖(5)中我們可以看的出，各時期的系統模式演進。

然而在進化的過程中，由「大型主機系統」至「主從式架構」我們可以發現，原有的大型主機系統中 UI 的陳現僅僅停留在終端機的文字界面，只能用簡易的圖形符號來陳現資料的表格及文字的輸入，進化到了「主從式架構」的圖形化使用者界面，不僅如此；用戶端在表單處理更能作到即時的資料驗證，操縱也更直覺。

隨著網際網路的腳步，系統的開發愈發向前行，以使原本僅能待在公司內部(Intranet)的系統，一躍龍門之下，以瀏覽器為主新建置的應用系統已不在單單提供公司內部(Intranet)的人使用，更不需於每位使用者的電腦上安裝 Client 軟體，節省了各個 user 端的維護成本。然而這就是我們想要的系統模式嗎？

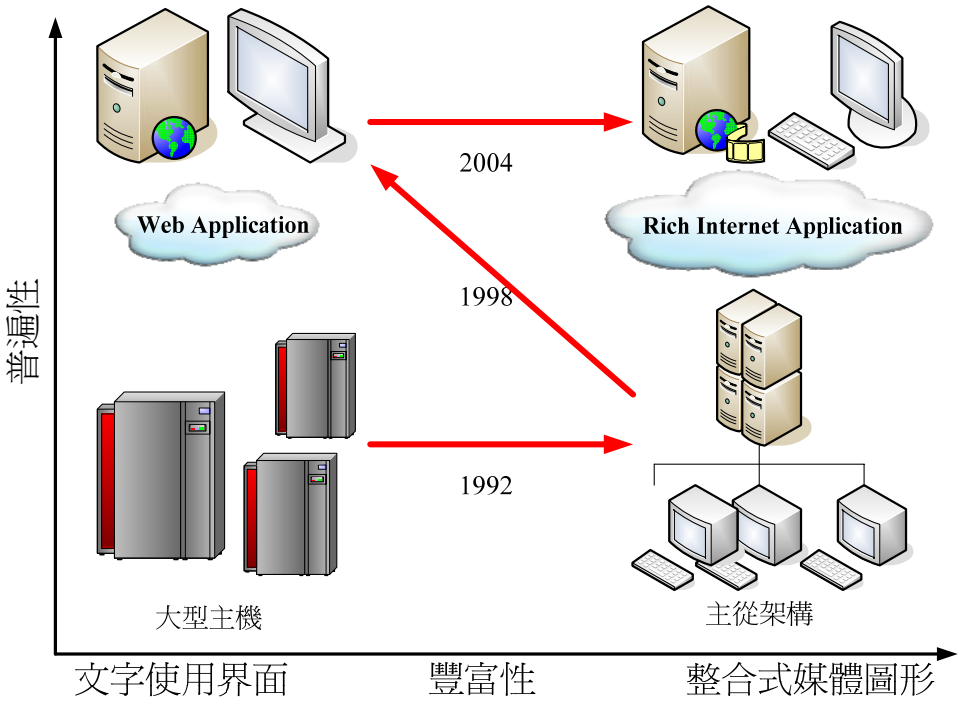


圖 5　使用者 UI 演進圖

「瀏覽器式應用系統」造就了普遍性與方便普遍性使得系統達到了高全域性，卻也因使用瀏覽器為主的建置模式延伸出許多資訊應用系統使用者許多隱憂突顯出他的美中不足。

2004 年以前以瀏覽器為主的建置模式是以傳統的 HTML 為基礎的網頁程開發，每一次的 Active 都由一個需求及回應及更新網頁來完成。而網頁中每一次的傳輸內容以資料的角度來說有 95%都是 Garbage (HTML 的 Tag 標籤) 。

RIA 簡單的說就是 Rich Internet Application 的簡稱；然各家系統廠商也常依其特性及強調重點不同，將 I 作了多種的解釋，分別有 Internet(強調網路連結)、Interactive(高互動)及 Interface(美觀介面及耦合度佳)等[07]，筆者認上述的解釋都是 RIA 特性的延伸，一個號稱 RIA 的使用者介面的應用程式，應三者兼具，欠缺其中的任何一項技術，都不完善的 RIA 前端介面。

而使用者介面隨著資訊系統的面也由大型主機時代的文字使用界面，朝向整

合式 RIA 媒體圖形界面發展。讓原本只是單純文字表單的資料陳現改頭換面成為今日具有直覺化、直接操縱(拖曳..直接加入購物車) 、資料視覺化（各式動態陳現表單）的介面，讓使用者介面不再單純的只是一種展示層，而是一種新新使用者端的應用程式。

2.8. 軟體重整工程 (Software Re-engineering)

軟體重整工程(Software Re-engineering)也稱為軟體再生工程或軟體再造工程。其主要在因應資訊系統因時代的變遷撰寫軟體的程式語言、設計的系統架構、硬體設備的效能、商業邏輯的變更及業務的快速發展導致舊有的資訊系統或稱遺留系統(legacy system)已不符合使用者使用的需求時需針對該資訊系統進行昇級或變更原有的需求，已符合使用者的需求時提昇換轉換資訊系統的常用方法。

為了達到上述的需求，資訊部門不是重新建構一套全新的系統，不然就是在現有的系統上加入新的功能元素，然而在原有的文件及技術傳承較好的公司內，增加新的功能可能手到擒來非常容易，但往往我們會發現，原有的資訊系統，經常因為外包，或是年代久遠原本的經辦或系統規格等相關文件皆已遺失，以至於須自原有的系統及程式碼中找出原有的系統規格。此時就必須運用到軟體重整工程中的逆向工程(Reverse Engineering)，透過逆向工程中對架構分析和程式碼的瞭解(program understanding) 重新組譯等方法獲取正向工程(Forward Engineering)所需的新系統規格。且透過逆向工程的分析過程中，了解原有系統設計模式的優缺點並配合新的系統架構加以修正及發揚，統一匯整原有的程式方法與元件類方消除重覆性，賦予新的介面再利用(Reuse)。

圖(6)用以表達開發新系統的正向工程(Forward Engineering)及從舊有系統中

取得相關系統規格的逆向工程(Reverse Engineering)的概圖，說明了正向工程及逆向工程在流程與架構上的差異性。

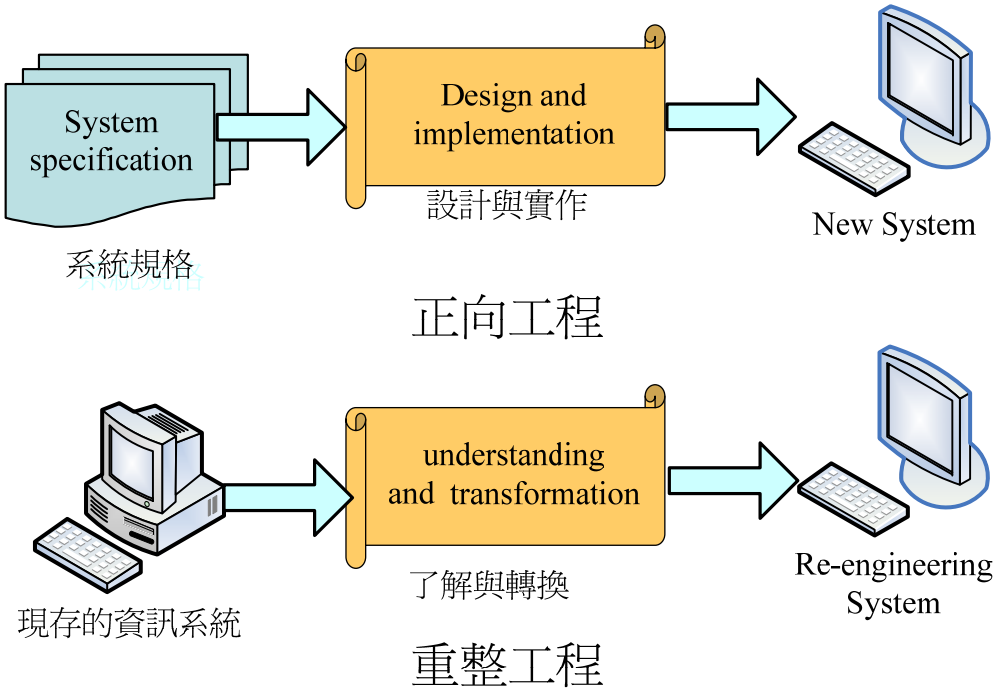


圖 6　新軟體開發與軟體重整工程圖

第 3 章. 個案探討

3.1. S 銀行網路金融服務平台探討

在此我們以目前國內知名的銀行（以下簡稱 S-銀行）之資訊部門為例：該部門現行負責維護的網路交易平台包含了個人網路銀行、企業金融網(FXML)、網路 ATM 及銀行入口網站等部分，其各別建置的時間及使用平台、技術分述如下比較表(1)；由表中可得知三套系統各自獨立，無論是軟硬體架構、系統軟體、應用伺服平台、開發平台等大不相同。

表 1 S 銀行現行網路服務平台開發架構

專案項目	個人網路銀行	企業金融網	網路 ATM
建置時程	民國 92 年至今	民國 93 年至今	民國 95 年起案至今
硬體	IBM R6 伺服器	HP 伺服器	HP 伺服器
系統軟體	AIX	WINDOWS SERVER 2000	WINDOWS SERVER 2003
開發平台	IBM WSAD	Borland J-BUILDER	Borland J-BUILDER
應用伺服平台	IBM WebSphere 5.0	Tomcat	IBM WebSphere 6.0
網頁伺服平台	IBM Portal server	Tomcat	Tomcat
應用系統採用架構	J2EE + STRUTS	JSP + servlet	JSP + EJB
Web Model	Model1、2 混用	Model1	Model1、2 混用
資料庫	IBM DB2	Microsoft SQL server	Microsoft SQL server
交易認證機制	ID+PASSWORD 動態密碼卡	ID + 憑證	晶片金融卡

依表(1)看來，以上三套系統除了使用的開發語言同為 JAVA 外，其餘軟、硬體架構及開發平台、應用伺服平台、網頁伺服平台、資料庫等，因各個建置的時間點、廠不同，對系統的組成也有所不同及差異，尤其影響最深的應用系統採用架構及與主機通訊閘道均大不相同，使得該部門系統維護人員在開發及維護上有極大的困擾。且使用的技術南轅北轍，導致三組人力難以互相支援，造成該部門

開發之維護成本及人力上的浪費。

表 2　電子化銀行通路及共通性常用功能比較表

電子化銀行通路

電子化銀行通路	電話銀行	WebATM	網路銀行	FEDI	FXML
服務時間	廿四小時	廿四小時	廿四小時	廿四小時	廿四小時
服務方式	自助式	自助式	自助式	自助式	自助式
跨行單筆 金額限制		200 萬	200 萬	2000 萬	2000 萬
訊息內容	單筆	單筆	單筆	多筆	可帶 99 筆
交易對象	一對一	一對一	一對一	可多對多	可多對多
約定方式	約定帳號	約定帳號	約定帳戶	約定憑證	約定憑證
交易模式	不可預約	不可預約	可預約	可預約	可預約
安全驗證	約定轉帳	金融卡	轉帳密碼 動態密碼 約定轉帳	金融憑證	金融憑證
共通性常用功能					
帳戶總覽	✘	✘	✓	✓	✓
存款餘額查詢	✓	✓	✓	✓	✓
存款交易明細查詢	✓	✓	✓	✓	✓
轉帳交易	✓	✓	✓	✓	✓
台幣轉外幣	✘	✘	✓	✓	✓
外幣轉台幣	✘	✘	✓	✓	✓
外幣轉外幣	✘	✘	✓	✓	✓
轉繳中華電信電話費	✓	✓	✓	✓	✓
轉繳信用卡費	✓	✓	✓	✓	✓
台幣稅款繳納	✘	✓	✓	✘	✘
外幣匯出匯款	✘	✘	✓	✓	✓

其中尤以主機通訊閘道最為困難，個人網路銀行因開發公司工程人員相繼離職已無法維護之外，且企業金融網及網路 ATM 技術均掌握在廠商，使得每當新增交易時，均須支出一定費用。而三個平台所提供的服務多有雷同，如轉帳、餘額查詢、明細查詢等，所不同的是三個平台所使用的交易認證機制不同，但每當

共同的交易必須修改或新增時，卻需分別向三家廠商支付費用，如此除了開發經費必須負擔三倍價錢，投入的人力亦需三倍，長久下來對於 S 銀行將造成極大的損失。為促使該銀行在電子商務的發展上能提昇競爭力，解決目前的困境，重新建構一個新的電子金融整合平台已是迫在眉睫的工作。

3.2. S-Bank 網路銀行面臨的挑戰

依該銀行的 92 年多階段規劃建置的網路銀行系統而言，在 95 年完成現有資訊單位所規劃的功能後，歷經了銀行合併及推廣行銷活動後，其銀路銀行申請人數自 95 年一月至 97 年一月，圖(7)表示了 S 銀行在短短二年之間，網路銀行客戶申請人數的成長曲線，其網銀申請人數已由剛建置完成時的 12000 名客戶，增加至 180000 名客戶，成長了近 15 倍的客戶量。目前系統之硬體營運效能雖尚流暢，然在每月高達三千人以上的成長量下，終有一天舊的網銀硬體系統也將無法負荷。

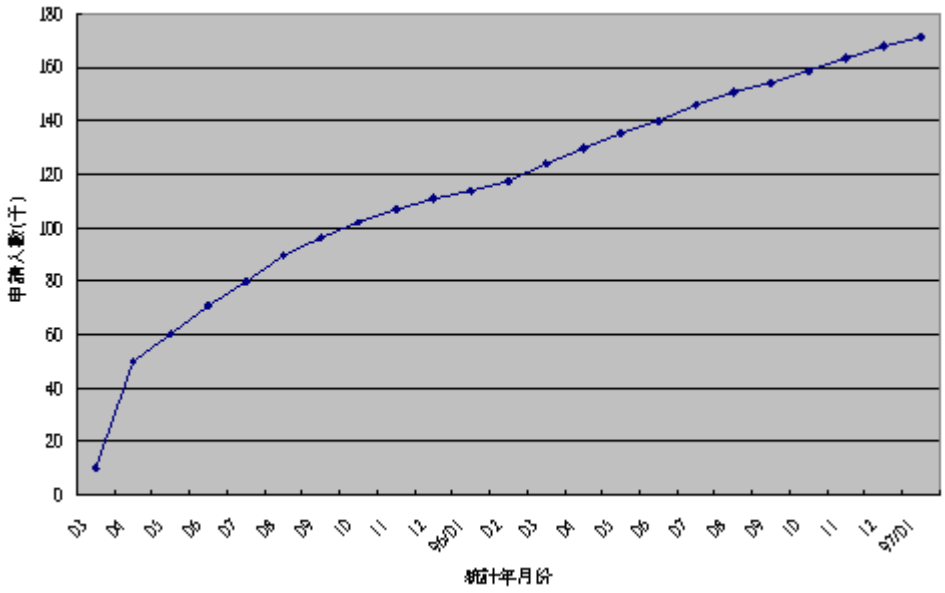


圖 7 S-Bank 網路銀行人數成長

圖(8)所示是現行的 S-Bank 網路銀行現有的系統架構，而依照現有的架構，若要提昇硬體系統效能勢必要更換主要的伺服器，且無法在營運的狀態下提昇效能，而更換下來舊伺服器無法在同一系統中發揮其功用，且有可能形成伺服器的浪費，提高了經營及維護的成本。

如何能建構一個負有彈性又能符合銀行營運成本，達到 24 小時不中斷服務的系統已成為該銀行不可忽視的挑戰。

原網路銀行系統架構圖

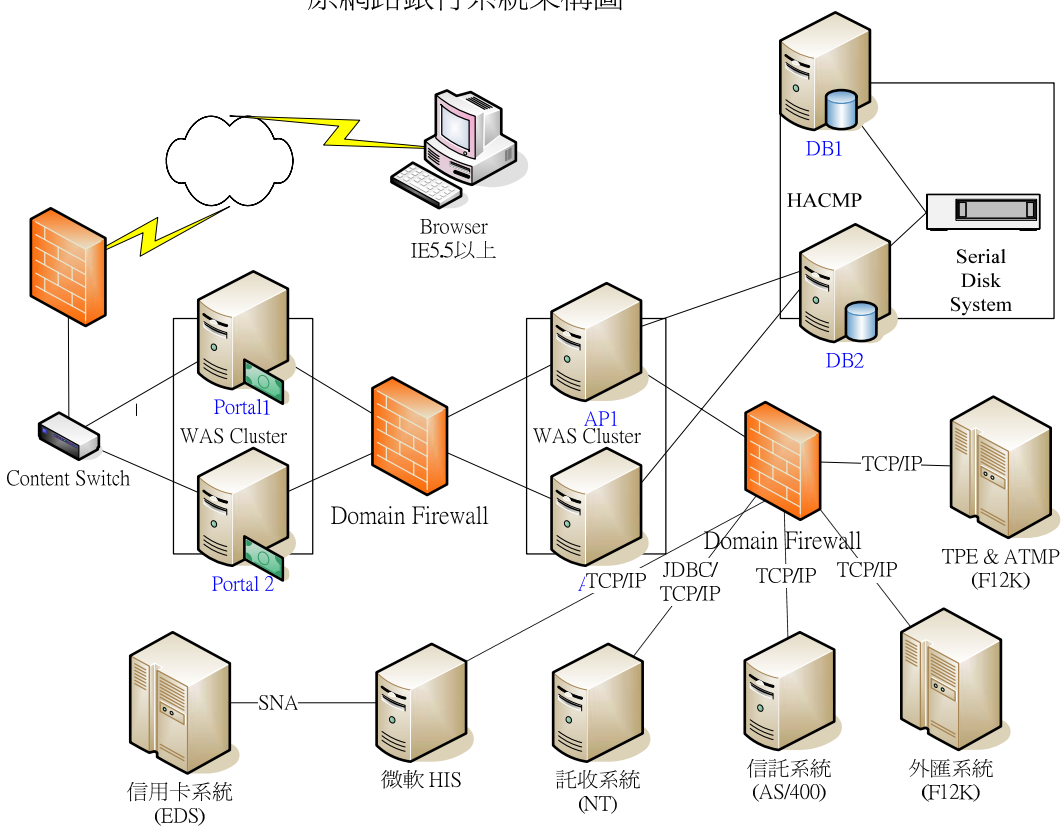


圖 8　S-Bank 網路銀行現有的系統架構

第 4 章.　實作規劃與建議

4.1.　系統需求與目標

以 SOA 服務的概念打造具有下列特性的網路金融交易平台:

A.具鬆散耦合(Loosely Coupled)可延展性的智慧型使用者介面及各自獨立的應用系統服務。

B.符合金融機構辦理電子銀行業務安全控管作業基準之全方位電子化金融安全交易平台。

C.建置彈性提昇及擴充架構，增加維護效能，達每週七天每日二十四小時全年無休不停機之電子金融交易平台。

D.整合後端各主機平台，建立模組化及公用化的整合模組。

4.2.　整合式電子商務金融服務平台

圖(9)是依據本研究概念所需繪製出的整合式電子商務金融服務平台 Financial E-Commerce Integration Platform (FECIP) 概圖，主要共區分五大區塊：包含了前端各種不同的金融交易介面、處理商業流程的商業管理層、異質系統間的資料轉換及層(Middle Ware)、銀行的核心系統以及各子系統模組化公用化的交易溝通介面等模式。

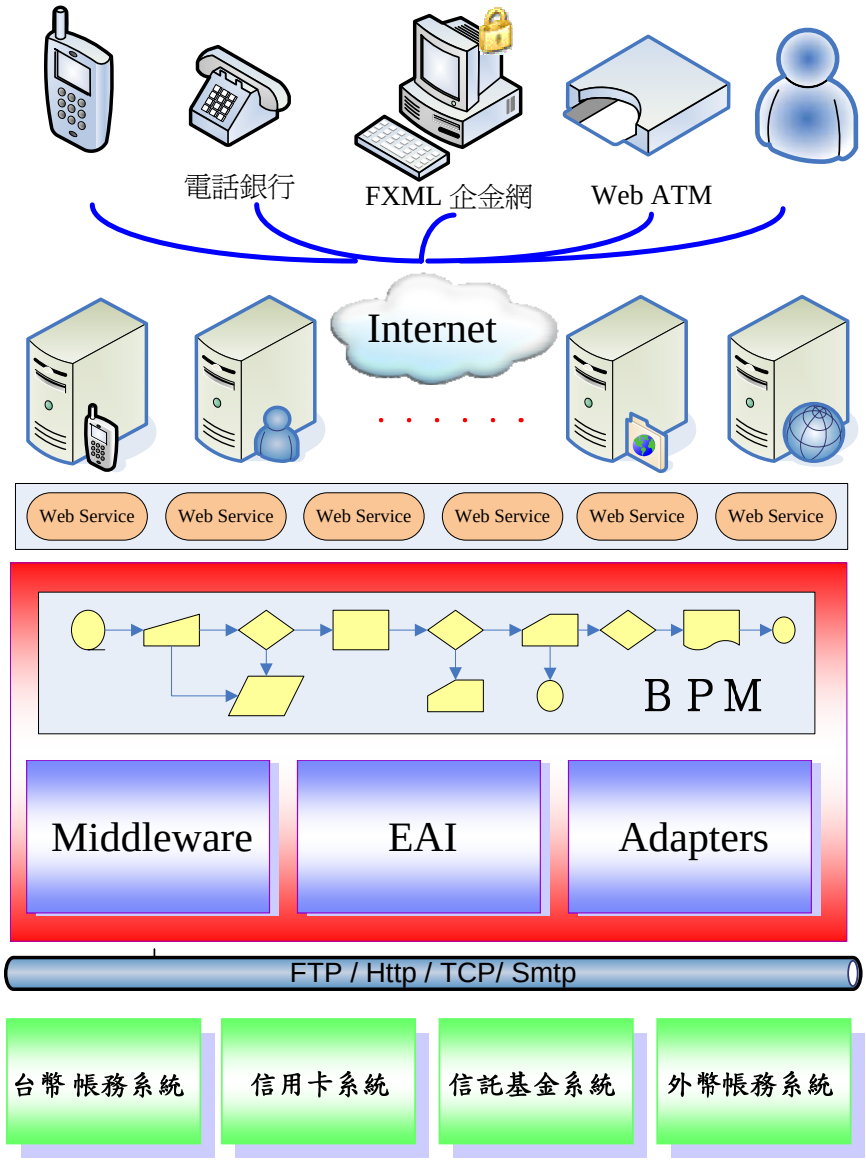


圖 9　FECIP 系統架構圖

FECIP 建構在 Internet/Intranet 以 XML 為基礎的公開標準下，建造一個 BPM 的處理平台利用此平台的靈活性處理商務性的流程。如圖(12) 所示利用圖形化介面來實作 BPEL 以便能快速的反映符合業務拓展的商務需求。

FECIP 將原有圖(1) 各子系統須錯綜複雜的建立 Gateway 與各後端平台 Gateway 作連接，改以建構一 Middle Ware、EAI、Adapters 等資料交換整合處理平台作為統一的訊息交換平台，減少系統與系統間 Gateway 的資源浪費。

圖(12)為依據傳統的銀行電子交易系統及依本文之整合式電子商務金融服務平台（FECIP）服務導向業務整合特性，銀行資訊人員對於往後的系統開放與建置的差異比較圖。

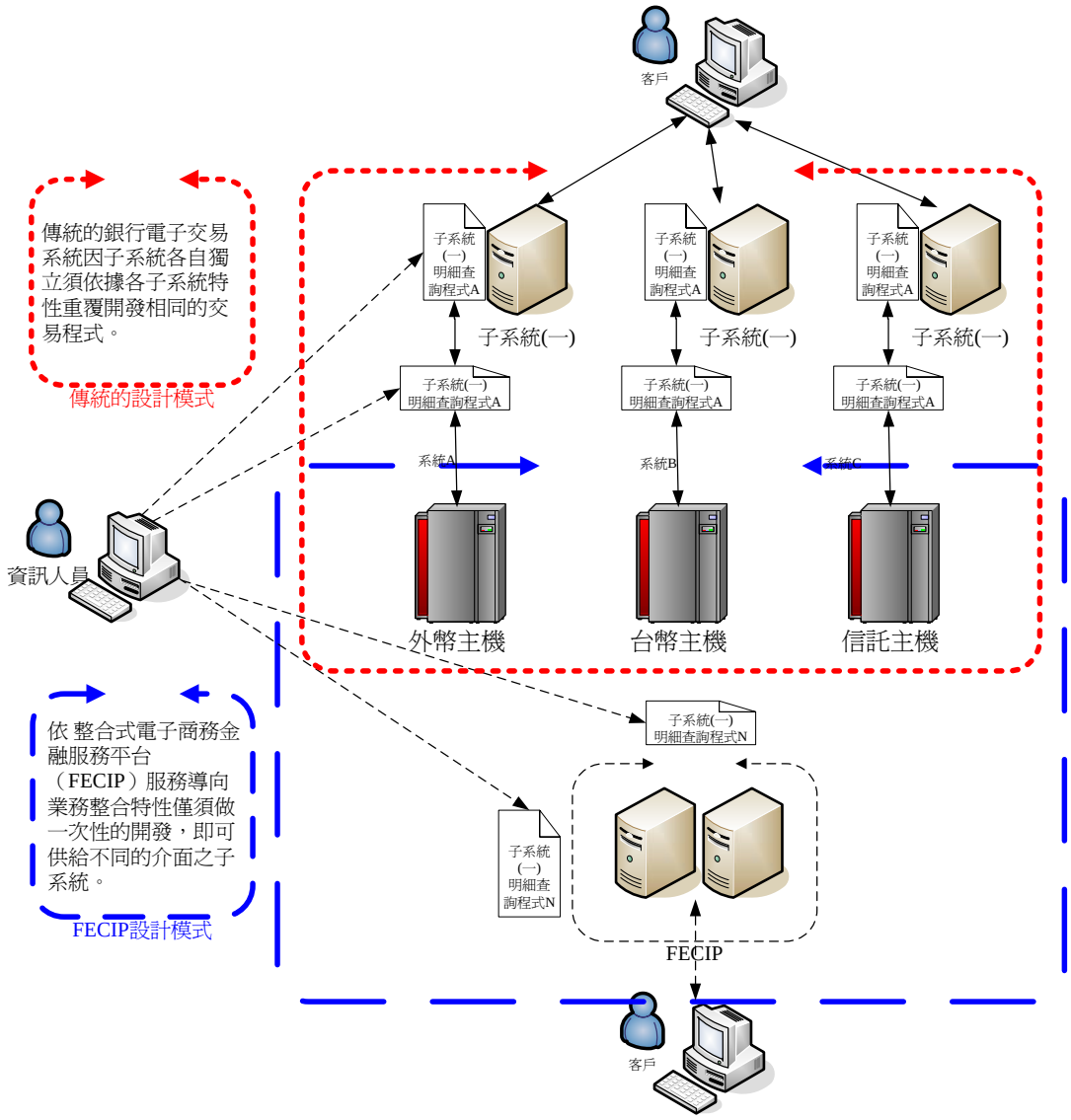


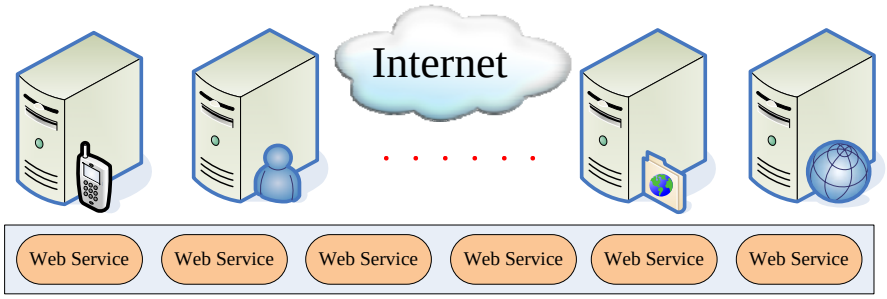
圖 10 傳統系統與 FECIP 開發架構的差異圖

圖中我們可以觀察到，紅色的框架中為在過去傳統的銀行系統，因各子系統各自獨立，須依據各子系統特性重覆開發相同的交易程式。相同的工作卻須花費 N 倍的人力及資訊成本去維護。然而若在系統規劃時便依整合式電子商務金融服務平台（FECIP）的角度重整電子金融平台（藍色的框架），對未來的銀行資訊人員僅須做一次性的開發，即可供給不同的介面之子系統。

4.2.1. 前端金融交易介面層

金融交易介面層，主要是依據各業務需求所開發的各子系統使用者介面，在整合式電子商務金融服務平台（FECIP）的開發架構下，可彈性的加入各個子系統，以使用者的角度而言指的就如傳統的個人化網路銀行、企業網路銀行、網路 ATM…等銀行系統。

圖(11)為依據整合式電子商務金融服務平台(FECIP)所之業務特性所訂定出來的業務藍圖，其開發的模式不再以以往的系統屈就於系統的限制，專為特定系統開發特定的功能，而改採取服務(業務)導向進行系統的設計模式，將原本的網路銀行，企業金融網，FXML，FEDI，網路 ATM 等各大系統的功能加以分析，降低重覆性質的交易功能的開發，再利用共同定義的溝通介紹（Interface）進行系統的整合，以符合快速變遷的業務需求。



整合式電子商務金融服務平台(FECIP)業務藍圖

圖 11　服務導向的交易需求

4.2.2.　FECIP 的商業管理層

為了配合突發性多變化的業務行銷機制，電子商務金融服務平台須快速迎合市場行銷及業務拓展部門的需求，FECIP 導入了圖形化的 BPEL 介面，使非資訊

技術人員也可以藉此圖形化介面訂定新的執行流程，快速的符合業務單位的需求，使系統更具彈性。

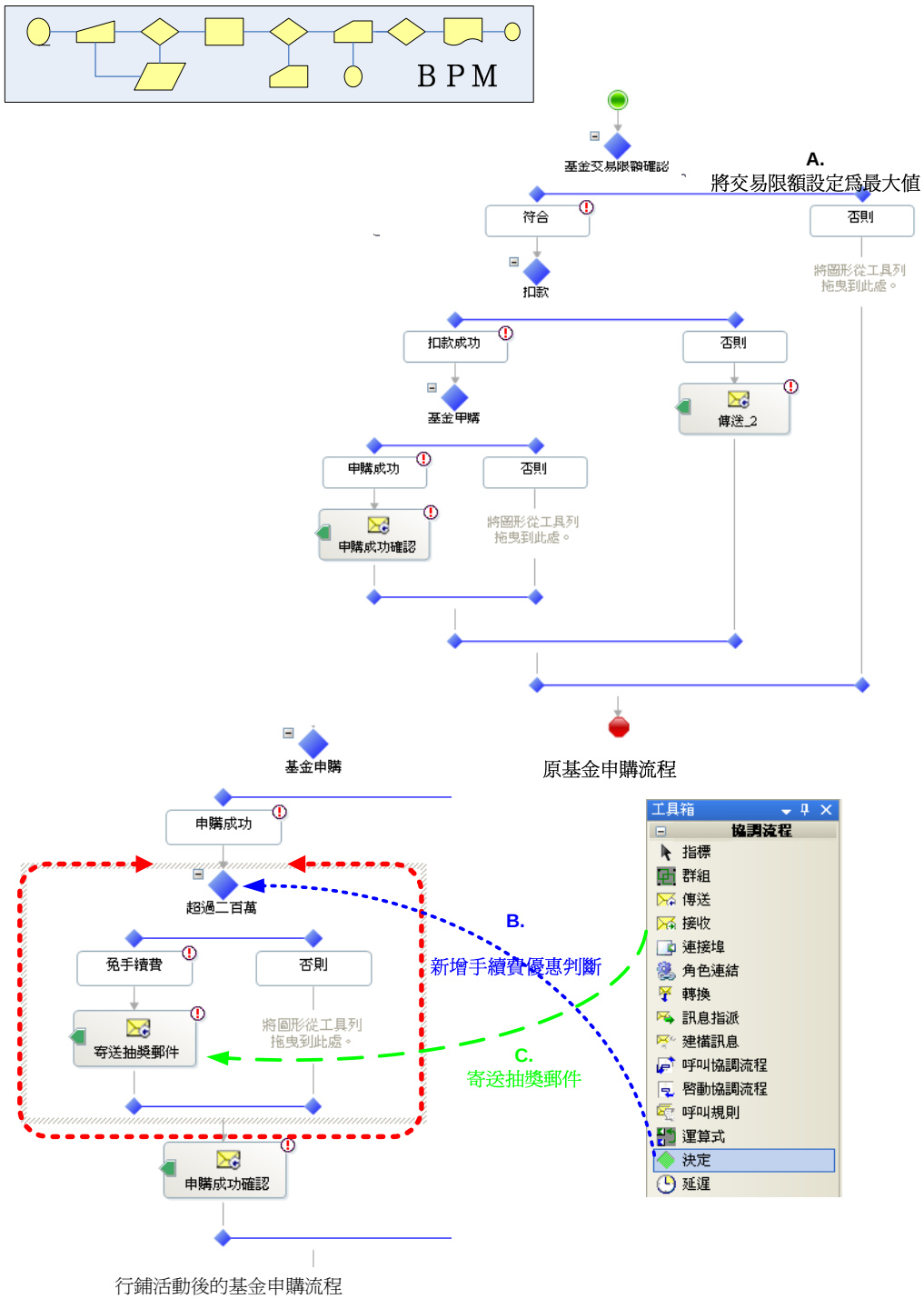


圖 12　圖形化的 BPEL 實作商業流程管理

圖(12)以基金交易為例，銀行的信託部門為了配合股票市場的熱潮，配合大額的投資客戶的需求及行銷活動，須即時調整網路銀行單筆基金下單方式，須將原訂定的台幣購買基金二百萬台幣的限制性取消，並依據其投資金額減免其手續費用，符合申購優惠資格的投資客戶將可收到電子郵件的抽獎資格。

在以往過去以傳統的 Coding 方式，須由資訊人員或外包廠商協助修改所屬系統的程式，並配合於活動的時程調整程式的內容。本論文規劃的 FECIP 模組，希望借由（BPEL）以圖形化的介面，透過信託部、業務部或管制企劃等非資訊技術人員來定義及更新優惠方式。

4.2.3. 資料轉換及交易執行服務化

圖(13)為本系統 EAI 的資料轉換及交易元件的執行概圖，前端的各項金融交易透過Web Service、http 、 FTP 及 Socket 等不同方式與銀行後端的 Core Banking 進行交易的執行。

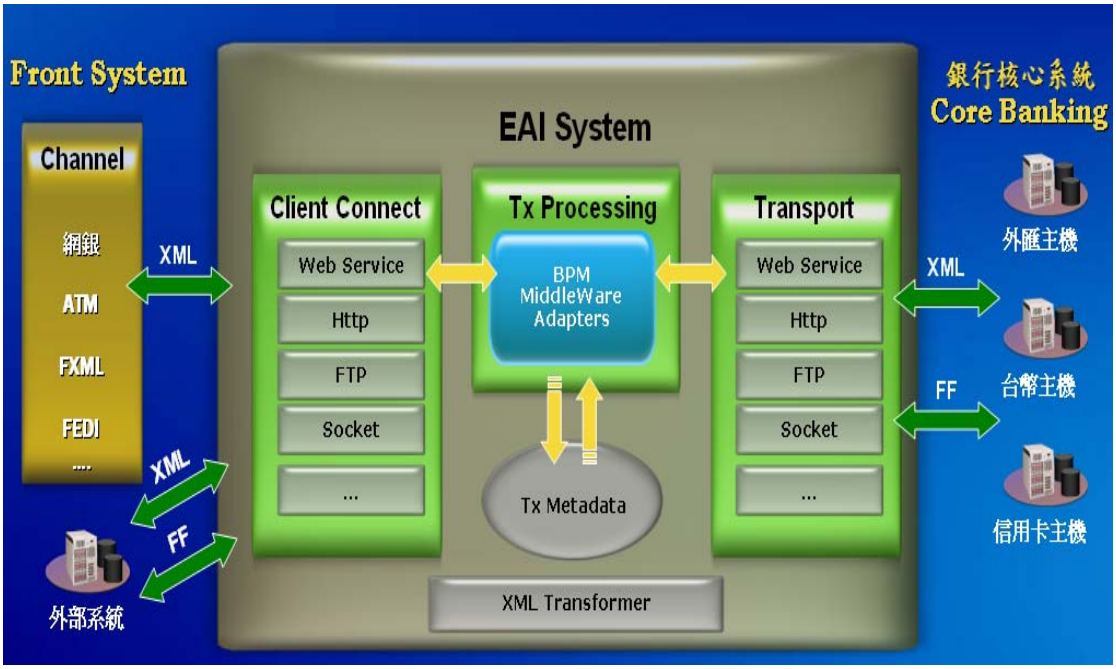


圖 13 EAI 的資料轉換及交易元件的執行

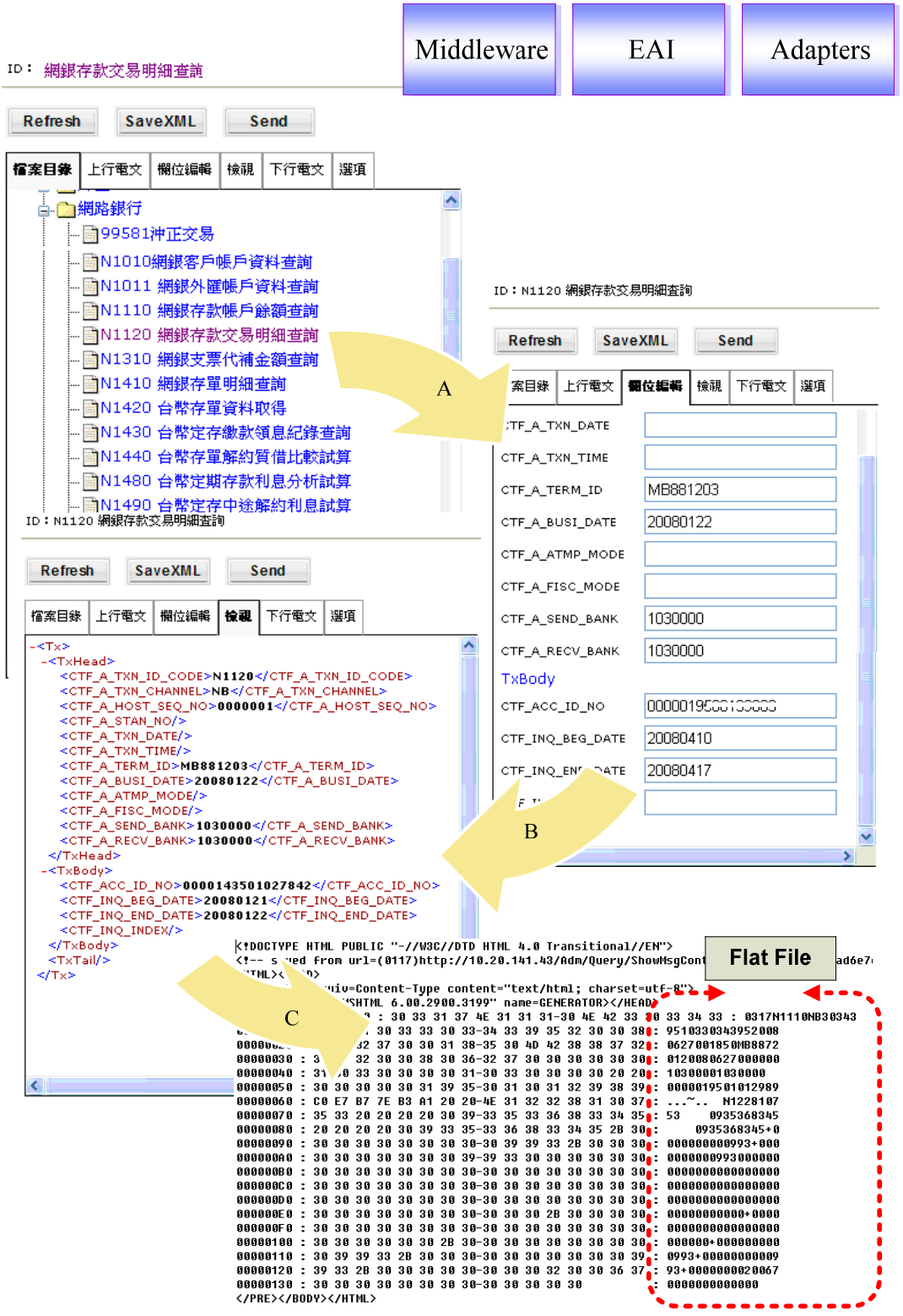


圖 14　單項交易的 web service 測試

圖(14)為電子商務金融服務平台(FECIP)依據業務藍圖的各項業務的每一個交易設計出的 web Service 呼叫測試介面，在本系統中我們將整合平台業務藍圖中的各項業務所定義的系統模組賦予每一個交易一個電文的交易代號，而前端的系統程式再依據所需的交易定義在 BPM 的商業流程中彼此透過 web service 進行溝通，透過圖示的可以看出 demo 介面以將要傳達的交易內容透過 XML 及 Web Service 的執行傳送電文至後端的 Core Banking 完成交易的執行。

4.3. 彈性提昇與無限擴充架構的電子商務金融服務平台

如圖(15) 所示，FECIP 伺服器系統架構中將各伺服器所負責之任務獨立化單純化及區塊化，各個區塊伺服器皆配備二台以上伺服器建立線上備援機制。在每一個區塊伺服器前皆架設網路負載平衡（Network Load Balancing，NLB），藉由網路負載平衡架設，可以讓使用者之要求（Request）經由一邏輯運算選擇網路負載平衡叢集內某一伺服器予以回應（Response），因此各伺服器平均分攤了所有使用者的要求，即達到所謂的 IP 負載平衡功能。網路負載平衡架構更是一種提供輕易增減伺服器的技術，尤其在發現伺服器負擔過重時，能輕易以追加機器的方式，立即增加可服務之人次數，而不需大幅度修改系統配置甚或大費周章汰換硬軟體有效提昇維護效率，降低營運成本。

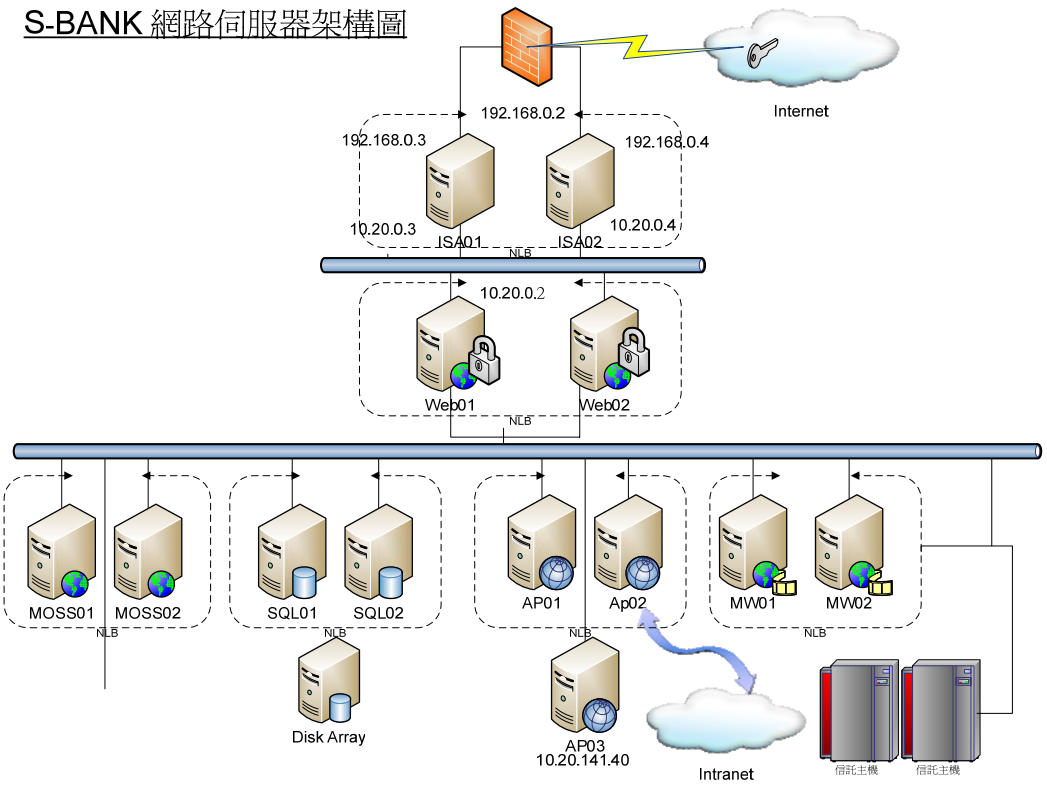


圖 15　FECIP 伺服器系統架構

依金融電子商務平台(FECIP) 的伺服器規劃架構，便可於日後系統異動時，獲得較大的擴充彈性。如上一章節的 S-Bank 網路銀行面臨的挑戰中提及到的，S-Bank 在短短二年之間，網路銀行客戶申請人數，已由剛建置完成時的 12000 名客戶，增加至 180000 名客戶，成長了近 15 倍的客戶量，因使用人次的不斷增加，而發現某一伺服器區塊的伺服器不堪負荷，需置換更高級之伺服器，或追加伺服器，以提供更大的服務量，此時則能以最小更動幅度完成複雜的擴充功能協助解決目前系統擴建的不易性，如圖(16)。

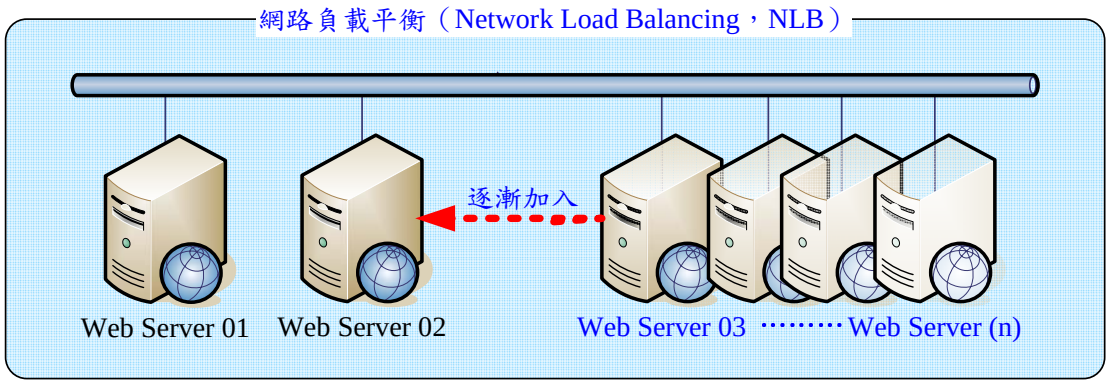


圖 16　負載平衡模擬設計圖

4.4.　全方位電子化金融安全交易

依據創市際市場研究顧問公司於 2007 年公佈的台灣地區金融服務使用行為調查報告[09]中針對未使用過「網路銀行」服務的網友。探詢沒有使用的原因中指出，網友們不使用網路銀行的主要的原因為「擔心資料安全」，佔了百分之 58 的，其次則是「比較習慣親自至銀行辦理相關服務」(37.1%)。根據該調查指出的「網路金融的安全性」與「使用習慣」，是銀行業者在推廣網路服務時，需要突破的關鍵因素。

然而就銀行資訊系統而言「網路金融的安全性」不僅僅是保障客戶的權益，更是銀行資訊部門的重大責任。在此本研究針對影響網路銀行安全性的重要因素分成「伺服端的系統安全」及「使用者交易安全」二方面去探討。

4.4.1.　伺服端的訊息傳遞的安全模式

本系統架構中，由於伺服器與功能運用之任務均以獨立化單純化及區塊化，以 SOA 實現 Web Service 平台架構下，系統與系統間依賴著網路(Internet、Intranet)達到資料交換傳遞的目的，因此在網路的傳輸過程中，便會有安全的危機產生，

因為在網路的傳輸過程中的資料有可能遭到竊取。遺失，或是修改，為了確保在資料傳遞時的安全，執行 Web service 時至少須符合下列五項的安全性。

一、身分確認性（Authentication）：確保存取資料的系統擁有取得該資訊的權力。

二、資料完整性（Integrity）：確保資料在傳輸過程中不會被竄改。

三、資料隱密性（Confidentiality）：確保資料在傳輸過程中不會被竊取。

四、不可否認性（Non-repudiation）：係指傳送/接收方無法否認其傳送/接收訊行為。

五、存取控制（Access Control）：存取控制，屬於最近才逐漸被認可的資訊安全元素，需得到存取控制的權限，才能授權使用相關的 web service。

FECIP 系統架構建議各個子系統與子系統之間，伺服器區塊與區塊之間的訊息傳遞交換皆須通過 SSL 的驗證，SSL 將伺服器憑證服務應用於 Server 與 Server 之間所傳輸內容經過加密保護、保護訊息完整性及雙方建立交談時身分辨識，確保訊息在網路(Internet、Intranet)上流通時，不會被網路上的駭客或不肖的內部人員中途攔截解讀(資料保密)或被網路上的駭客擅自修改資料(資料完整)防止第三人經由網路竊知傳輸之任何訊息。

4.4.2. 使用者自我意識下之安全控管

網路交易的日益興盛，相對的網路上交易卻也屢傳駭客利用木馬程式盜取用戶密碼與憑證的事件，雖然財政部金管會銀行局銀行公會等單位訂定了許多金融機構辦理電子銀行業務安全控管作業基準(如:約定轉帳、非約定轉帳限額、增加

圖形化驗證碼...等)來提高網路銀行使用者的交易安全，2004 年 5 月，財金公司釋出一千萬推動網路 ATM 的方案，希望借助晶片金融卡的驗證功能加強網路交易的安全。依金融監督管理委員會銀行局最新修訂的「個人網路銀行業務服務定型化契約範本」[01]中也修正了原有的條文「第三人破解使用電腦之保護措施或利用電腦系統之漏洞爭議，由銀行就該事實不存在負舉證責任。」加強保護網路銀行使用者。然相對之下須由銀行負擔其危害的舉證之責，銀行勢必推出更多的客戶保衛計劃。

然而在網路銀行使用者對網路上交易的自我保護不了解的情況下，依然出現了使用電子簽章及網路 ATM 時存款被盜領的事件，雖然電子簽章及網路 ATM 的晶片金融卡可確保無法否認傳送訊息部分及證明該筆交易之存在外，卻無法證明該筆交易是否出自於使用者之意願 (網銀使用者將電子簽章存於硬碟、瀏覽器或將晶片金融卡置卡讀卡機中未取出) 。在本系統就使者介面的驗證模式上提出了「使用者自我意識之安全控管」的建議，所謂的使用者自我意識之安全控管機制是必須建立在人為親自介入的狀態下方可完成該交易，以確保交易的完整性。然而如何達到此種效益呢！在此希望銀行建議客戶在使用網路進行轉帳、購買基金及個人資料變更時，均能使用下列的驗證方式執行交易，強化網路交易的安全性。

4.4.2.1 動態密碼(One Time Password System)

動態密碼（One-Time Password，OTP）是一個隔絕外部攻擊的安全系統。而 OTP 不是防火牆，也無法真正防止網路中間人(man-in-the-middle) 竊取您的資料，也不能防範來自內部網路的攻擊，那麼 OTP 何以為網路銀行使用者提供保護呢？

OTP 動態密碼是一種優良的密碼安全機制，是抵擋竊取密碼攻擊(人為窺視取及木馬竊取)的第一防線，網路入侵最常見的方法就是竊取用戶帳號密碼，一般的靜態單一密碼系統，簡單易記的密碼容易被猜測出來，而複雜的不易記憶，

種種使用上的人為因素導致不安全的結果。因應網路交易安全而生的動態密碼最大優點是，使用隨機亂數產生的密碼，根據事件每次產生不同的密碼，而且只使用一次(One-Time Password)。即使受到網路駭客的攔截到這一次的密碼，也無法應用到下一次的交易。

且經由 OTP 動態密碼的交易，必須由使者本身親力親為來產生動態密碼，符合了本論文提倡的「使用者自我意識之安全控管機制」

圖(17)為 S 銀行使用的 Token 型動態密碼卡，網路銀行客戶可由 Token 型動態密碼卡產生動態密碼存戶輸入密碼連續錯誤三次，或動態密碼卡使用次數空跳超過十次以上者，為保護交易安全，銀行便會逕行取消動態密碼卡功能。

圖 17　OTP 動態密碼 Token

資料來源：S 銀行電子商務部提供

圖 18　晶片金融卡、動態密碼二合一卡

資料來源：銀行公會

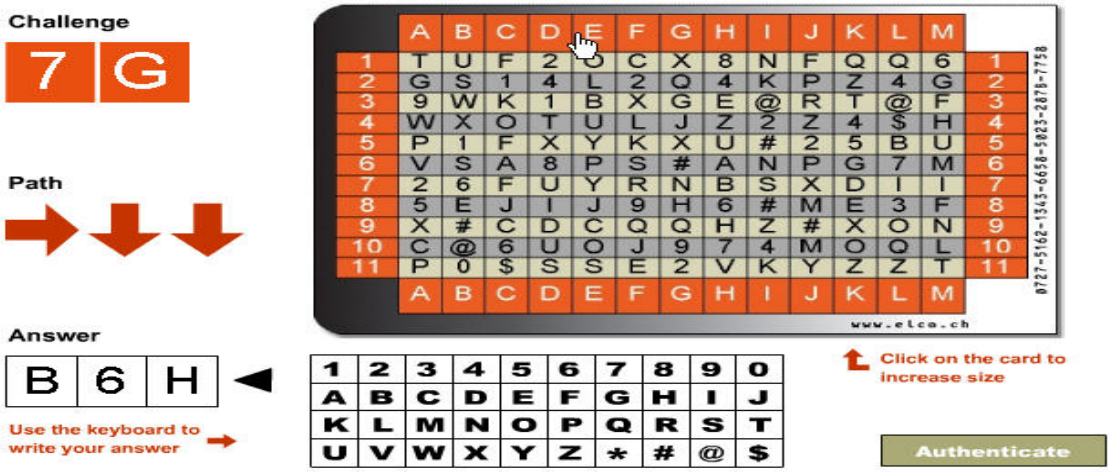


圖 19　簡易型動態密碼卡

資料來源：銀行公會

4.4.2.2 晶片金融卡搭配確認型讀卡機

在網路銀行交易的過程當中，須進行親力親為的驗證時，驗證方式可由晶片

金融卡搭配具有確認鍵及數字鍵的讀卡機來進行交易的確認，當系統收到交易訊息時，可確定該筆交易確實出於使用者意願發出，而非由木馬或遠端遙控程式所代為發出。若晶片內含 RSA 憑證簽章功能更能大大提昇網路線上交易的安全性。

圖 20　晶片金融卡確認型讀卡機

資料來源：S 銀行電子商務部提供

4.4.2.3 指紋辨識晶片金融卡

日前於法國巴黎舉行的 CARTES & IDentification 2007 大會中，已有系統廠商展示出了業界最新型且安全的新一代指紋辨識晶片金融卡；如圖(21)。其可運用範圍廣泛，可涵蓋各項高風險性之交易指示，該指紋辨識晶片金融卡必須經由使用者於指紋感應晶片感應確認，方可完成所賦予到交易，更完美的達到確認使用者之意願的需求。惟成本頗為昂貴，尚未有銀行能真正的運用在一般客戶之上，若能突破價格的束縛，將是網路安全交易的一大利器。

圖 21　指紋辯識型晶片金融卡

資料來源：銀行公會

4.5.　客戶的服務導向架構(SOA)新解

傳統的銀行業著重於銀行的金融業務，執著且保守固步自封。然而多數的 SOA 服務導向架構的設計模式也一直是以 IT 人員的觀點去設計規劃。如何賦予銀行的金融交易平台一個新生命，在實作規劃與建議的最後的一章節中，筆者希望賦予 SOA 服務導向架構的一個新解。

一直以來服務導向架構雖一直強調以服務「客戶交易的連貫性」作為系統設計開放的依歸，就客戶的角度而言，銀行的電子交易平台愈 Friendly 愈留住客戶，然而許多的客戶卻也因習慣而對特定的交易平台產生依賴。

如何讓爭取到「習慣」其他家銀行的客源，真正對銀行的電子交易平台產生依賴性，而不再只是因為「習慣」其銀行的操作介面而已。如何真正讓客戶經常性且不自覺的「習慣」使用銀行所提供的網路平台，增加使用該銀行的交易機率讓銀行的網路平台與客戶的生活相依在一起，達到真正的以客戶服務導向的電子

金融交易平台。

筆者建議銀行金融金控等業者能從觀念的轉變做起，設計一套在電子金融交易平台的設計觀念中不應單純的只存在有為了金融交易而建立，打破建置上網路的金融交易系統的思維，

引進社群的觀念，透過平台提供的附屬功能「理財知識園地」及「股票資訊討論區」的交流中於關鍵字中產生連接至線上的基金交易、「私人秘書」(行事曆)、「通訊錄」、「簡訊系統」並藉由主動依客戶自訂的行事曆，銀行的信用卡結帳、支票待補等，利用現有的紅利點數轉換成簡訊通知，並自動設定成轉帳扣款，無形中在銀行中形成一股「循環式的消費模式」。配合既有的企業網路銀行的使用者權限架構，轉化成家庭化的網路銀行，吸收整個家庭的成員。

銀行業者若是在網路上透過上述的經營模式再依據各客戶的性質搭配進行各項的金融行銷活動，應可觸及到不少潛在目標族群，提昇銀行的收入。

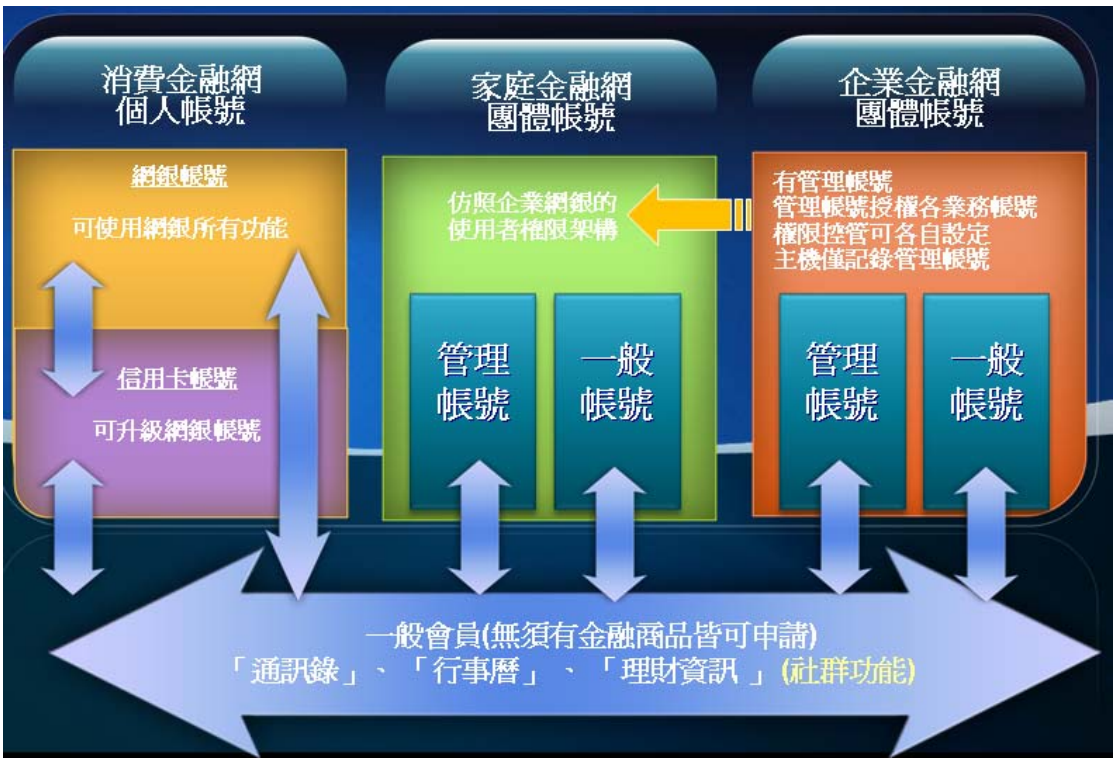


圖 22　社群化的會員架構

而上述的功能更可藉由客製化的 IM（Instant messenger，即時傳訊）軟體，使客戶一登入電腦後便可享受金融業者的各項服務。即時透過 IM 軟體與理專、客服溝通提供高品質的客戶服系統，提高線上客戶的服務滿意度。

客製化的 IM 軟體搭配 web ATM 讓每一台電腦都具有「Desktop ATM」虛擬 ATM 的行銷通路，利用頁籤主動提供金融行銷，講座等活動增加額外的廣告收益。

現今政府機關一直在提倡政府 e 化及 My Gov 等設計理念。而整合電子交易平台的設計觀念更應有 individual 個人化獨特性的精神，讓每一個網銀客戶皆有專屬（我的ｉ）網路銀行。使網路銀行不只是被動式的提供服務，而是主動化的為客戶解決金融上的各項問題，讓每一個網路銀行皆以客服為中心，真正達到（我的銀行 iBank）。拉大與其他銀行在線上服務的差異性。

第 5 章.　結論

SOA 在各大系統整合廠商的渲染下，對於各家銀行及企業來說，相當程度的也擔心 SOA 只是系統整合(SI)公司的一種資訊系統「行銷名詞」、一種「願景的行銷」：高不可攀、深不可測、無法評估，勢必得投入大量的金錢與人力成本才可實行的事。然而對一個銀行的資訊部門主管(CIO)來說，難免也會質疑實現 SOA 的成效？銀行內部的上百台舊系統是否有足夠的彈性可以變成 SOA 模式嗎？

回歸到 Gartner Group 研究機構對 SOA 的原始定義:「SOA 是一個概念、原則，不是標準操作程序。只要企業能運用 XML 的技術，把應用程式包裝成 Web Service 的形態，進而讓 Web Service 之間，或是 Web Service 與前端人機界面之間，依照商業需求彼此溝通，就算是一個 SOA 的系統」。就像一棟舊房子可以透過裝潢而翻新，不見得需要整個拆掉重建，在架構不變的狀況之下，漸漸的將介面變成服務模式來進行溝通。SOA 的精神並不在於推倒重來，應該是要在現有的系統之上活用。由小型系統專案開始進行服務架構、服務模式的改變，逐步證明其可行性，做出效果以後漸漸進入下一個階段，使整個銀行朝向 SOA 的全能銀行邁進。

由於本研究之假設是基於 S 銀行現行(2008 年)的系統架構、技術人員及企業成本為考量下作為評估的基點，反映了 S 公司該部門的情形，期盼所定義的系統架構模式能協助銀行解決金融平台整合上的問題。

未來的研究重點有：

1.繼續整合分析階段中的元件層之物件導向分析，成為完整的分析方法論。

2.繼續發展服務導向設計方法論，成為完整的服務導向軟體工程。

第 6 章.　參考文獻

中文部分

01.行政院金融監督管理委員會 (1996)，定型化契約範本，銀行局。

02.財金資訊股份有限公司(財金公司)，業務介紹。http://www.fisc.com.tw/

03.全球資訊網聯盟(W3C)， Web Services Activity。http://www.w3.org/

04.Microsoft MSDN　網站(2004)，實踐服務導向架構，林耀珍。

05.IBM，軟體之友 ，什麼是 SOA， http://www-07.ibm.com/。

06.ZDNet Taiwan (2006) 企業應用名家專欄，蕭百齡。http://www.zdnet.com.tw/

07.蔡學鏞 (2007)，RIA 說文解字，http://jerrylovesrebol.blogspot.com/2007/12/ria.html。

08.中華民國銀行商業同業公會全國聯合會；(銀行公會) http://www.ba.org.tw/ 。

09.創市際市場研究顧問，金融服務使用行為調查報告(2007)，
http://www.insightxplorer.com/。

外文部分

10.W3C. SOAP version 1.2. Technical report， W3C Web Site， June 2003.

11.E. Christensen， F. Curbera， G. Meredith， and S. Weerawarana. Web services description language (WSDL) 1.1. Technical report， W3C Web Site， March 2001.

12.E. Newcomer. Understanding Web Services - XML， WSDL， SOAP and UDDI. Addison-Wesley， 2002.

13.OASIS , Web Services Business Process Execution Language Version 2.0, http://www.oasis-open.org/committees/download.php/14616/wsbpel-specification-draft.htm , 2005

14.Active BPEL , website. http://www.activevos.com/community-open-source.php.

15 L.Clement， A. Hately， C. v. Riegen， and T. Rogers. UDDI version 3.0.2. Technical