東海大學資訊工程學系研究所

碩士論文

基於現有通訊架構上之
商業化網路傳真掃描平台
Commercialized Online-Fax and Scan Platform Based on the Current Communication Network Frame

研究生：陳聖道

指導教授：朱正忠教授

中 華 民 國 一百 年 六 月 二十七 日

摘要

傳真在通訊資料傳遞上佔了相當重要位置，在歷經多年的沿革及改良後，其速度及收取資料時，所列印出的文件清晰度有獲得一定程度提升，但在傳真的收發上，卻一直沒有特別進展，究其根源，為其目前主要方式依舊為兩方各準備一台傳真機，但這對於生活於 M 社會（Mobile Social）現代人而言，是一個非常不方便的工具，本研究則是在於利用現有的基礎建設，導入客製化系統功能開發，直接對於傳真的收、發方式進行改良，並新增使用掃描機的部分，使沒有傳真機的現代人也能使用此功能，並使其符合於現在使用人習慣。在傳真號碼的部分，導入了行動傳真碼(一種來自台灣的新通訊技術，即以電子郵件來接收一般的傳真文件，實現台灣傳真文件全世界都可以收得到的夢想（即 fax to e-mail 的技術）。 在台灣行動傳真碼目前是以 0945 或 0943 為開頭的門號）來讓使用性大大提升，在單機版的程式中，其智慧財產權的防護性相對較低，所以應用了現行技術中常用到的 SOA（Service-Oriented Architecture）及 Web Service 的方式進行系統身分的認證，讓一套程式配合帳號後，只能執行於一台電腦上，並採用了網路身分認證及硬體鎖的方式來確保系統及帳號的安全性；在資訊安全部分，其系統所有對外傳輸均加上 MD5 的加密演算法，使其避免因為資訊外顯，而導致伺服主機遭受到網路攻擊的行為發生，

在整個開發的過程，其概念採用敏捷開發（Agile Development）的設計模式來達到快速產出要求，且應用了軟體工程上常用的工具進行分析、設計、開發此系統。

關鍵字：傳真，傳真信箱，服務導向架構，摘要算法 5，網頁服務

Abstract

The fax machine plays a significant part in transporting information. After many years, the speed of receiving information and the clarity of documents have improved. The methods of receiving and sending, however, have made little progress. Currently the major prevailing method is to prepare two fax machines on both the receiving and sending sides, which is very inconvenient for modern people living in a Mobile Social society.

This study aims at bettering the modes of receiving and sending, with using existing basic constructions and developments of systematic functions made particularly for customers who don't own a fax. Besides, to fit in with modern people's using habits, we first import mobile fax code, a new kind of signal for communication made in Taiwan, receiving and sending general documents with e-mails which makes Fax to E-mail possible. The mobile fax code in Taiwan begins with numbers 0945 and 0943. Also, because the protection of the intellectual property rights is relatively lower in this program, SOA and Web Service are both used to recognize system identification so that one program can only be run in a computer with one account. We also use the internet identification and computer hardware lock to insure safety of the system and the account. Regarding safety of information, MD5 encryption calculation is added in all outward transmissions to avoid an internet attack on the computer server. The whole concept of exploitation adopts Agile Development design mode to meet the needs of high speed production, applying tools frequently used in engineering to analyze, design and develop this system.

Keywords: Fax，Fax to Mail，SOA，MD5，Web Service.

目錄

圖目錄

表目錄

第一章　緒論

1.1 研究動機及目地

在通訊上，傳真一直扮演著重要角色，在多年的延革，目前已成了非常穩定，且大量使用的技術。但由於其時空背景因素及先天性不足，一直都是以一對一的方式來做資料的傳遞，但在目前分秒必爭的時代，這卻是個能導致其使用率降低的致命打擊。

因為傳真的普遍使用性的範圍非常的廣範，而針對其功能功性加強的異動卻不多，固本研究的想法即時從這個地方出發，進而在於利用現有的通訊基礎建設，加上客製化系統功能開發，直接對於傳真的收、發方式進行改良，並導入掃描機的發送方法，使其符合於現代使用人習慣，期能夠發揮其應該有價值，進而達到隨處可收發傳真的理想。

在軟體開發的過程，軟體發展的生命週期就是將邏輯性的系統概念，發展規劃成可實作的系統設計文件後，以設計與撰寫程式碼的方式實現此系統概念，系統完成後，接著交付部署、測試運行、系統上線，最後進入運行維護的發展步驟。

在軟體開發，我們使用了 C#、Java、Jsp、WebService 等的相關技術，用來架構這個部分所需要的軟體功能，導入了敏捷開發（XP eXtreme Programming）及物件導向的觀念增加開發速度，減少維護

負擔，並在版權及安全性的考量上，使用了由 safenet 所開發出來的「聖天諾加密鎖」，與程式進行持續互動檢查，在後續的部分將會介紹是如何的使用這些已經成熟的相關技術來實作本平台。

本研究一開始，將會先介紹相關的研究與技術，並說明軟體的流程架構、開發相關細節及其管控點，接著介紹其目前已經商業化的相關細節，並針對客戶多有反應的部分，提出下一個版本的調整。

1.2 論文架構與研究

本研究論文撰寫的架構共分成五章，以下分別對各章節做簡述。

第一章：緒論及論文架構與流程。

第二章：相關研究及技術，其將會介紹本論文所實作在系統上的各種軟硬體相關技術，包括物件導向技術、資料加密、程式語言、網路服務及系統安全等。

第三章：系統架構與設計，針對本論文之平台架構、系統架構、安全性檢查機制做概念性的說明。

第四章：軟體介紹與功能描述，說明本研究如何應用前章所述相關技術與知識實作出一套商業性的網路傳真系統。

第五章：結論與未來展望，總結研究成果，並提出未來可供後續研究的方向。

第二章　相關研究及技術

在本節，我們將目前有關於網路傳真軟體的整個完整架構，其相關的研究與技術，做逐步的探討。

物件導向程式語言是目前軟體業當紅的核心開發技術，而 Web 化則是目前所發展的技術中，最廣為應用的展示及應用技術，此實作軟體開發程式將會結合兩種技術的使用，來達到其所需要的目地。

在所使用的開發技術上，在發送端部分，除了使用傳真機的使用者外，還有部分是使用電腦配置掃描機的使用者，由於考量到一般的使用者多以 windows 的作業系統為操作介面，所以使用了 C#語言來做為開發，使其在發送端能有較好的執行效率，在傳真的接收端則因需要較高的穩定性，故使用 Java 語言，配合 embarcadero 公司所出品的 Jbuilder，開發 Client 端，並使用 J2ee 的部分，開發 Web Service 的網路資料傳輸及認證的部分，其 web 所使用到的 AP Server 為 Caucho 公司所開發的 Resin 工具，其工具是目前公認為中型系統最佳的使用選擇，在平台收發 Mail Server 上，則是選用了國產公司的村榮資訊所制作的 Raidenmaild，在硬體的安全性部分選擇了由 safenet 公司所發展的「聖天諾加密鎖」來進行程式的安控檢查。

2.1 現有功能相關比較

雖說傳真在日常生活中應用的時間已經非常的久，且後來電子郵件等相關的技術發展出來後，更壓縮了其空間，但是由於其即時且方便使用的原因，其功能已經深入了一般的民間家庭，不論時業務的資料取得、電台的聽眾回應、政府企業的資料收集等，目前還是以傳真為大宗，其可以由周邊的便利商店都有提供傳真功能而得知其使用的普遍性。

不過由於傳真在傳送接收上還是有不足的地方，所以有人就提出了使用網路來當中介的 Internet fax 的概念，在經過一段時間的延格後，現在市面上也有相關的產品有進行提供的服務，不過由於其多由大型公司所推出，並不一定適用於中、小企業及一般使用者，以下，將針對市面最受大眾使用的中華電信產品先做比較。

項目	本系統	中華電信 Hibox
接收即時性	立即接收印出	儲存在信箱中
資料來源	傳真、掃描、郵件	傳真
輸出前預覽	提供即時點選預覽	無
垃圾傳真處理	自動判別無效傳真	無
同時連線釋	單一帳號 200 線	單一帳號 3 線

提供傳真碼	全球統一號碼	區域性號碼
存留時間	取決於本地硬碟空間	與信箱共用
資料備份	提供每通的備份	不提供備份
傳真轉發	可自定義轉發人員數	不提供轉發
轉發手機	有	有
本地資料安全	配合硬體加密鎖	無
自動轉檔	可轉為 JPG 圖型檔	無
多輸出支援	可支援同時 7 個輸出設備	支援單一輸出設備
多 ISP 支援	目前支援 3 個 ISP	本身中華電信

表 1.市售產品功能比較表

中華電信的優勢在於其強大的網路資源及大量的機器設備支援，但本研究所鎖定的為一般中小型的使用者，其能對其需求進行客制化，無論是反應時間、使用彈性等，均較中華電信為優，且由於本研究是使用第二類電信線路，但其基礎是基於中華電信所提供的線路，故能產生比其更具價值的效益。

2.2 敏捷開發（XP eXtreme Programming）

由於在開發的過程中，會遇到使用者時常提出需求的異動修改，如果使用一般軟體工程的開發模式，先進行需求系統分析、產出系統文件、轉成設計文件、進行開發、進行測試、產出測試文件的這種循

序的方式進行開發，則會無法反應出其需求調整的速度及彈性，固採用了敏捷開發的方式，讓文件和開發脫勾，系統的開發設定伴隨著只有一定程度的相關文件，待開發完後，讓使用者能在短時間內看到其需求調整的改變，進行提供未來繼續開發的依據，再開始做文件的補齊。以下，是針對敏捷開發所做的介紹。

速度是企業競爭致勝的關鍵因素，軟體專案的最大挑戰在於一方面要應付變動中的需求，一方面要在緊縮的時程內完成專案，所以軟體團隊除了在技術上必須日益精進，更需要運用有效的開發流程，以確保團隊能夠發揮綜效。這正是 Agile Process (敏捷的軟體開發流程) 於近年來興起的主要原因。

敏捷開發，是一種從 1990 年代開始逐漸引起廣泛關注的一些新型軟體開發方法，是一種應對快速變化的需求的一種軟體開發能力。它們的具體名稱、理念、過程及術語都不盡相同，相對於「非敏捷」，更強調程式設計團隊與業務之間的緊密協同作業、面對面的溝通（認為比書面的文件更有效）、頻繁的更新軟體版本、緊湊而自我組織型的團隊、能夠很好地適應需求變化的程式碼編寫和團隊組織方法，也更注重做為軟體開發中人的作用[11]。

Agile Process (敏捷的開發流程) 是一種軟體開發流程的泛稱，Agile Process 具有下列幾項共通的特性：

（1）客戶與開發人員形成密切合作的團隊，因為客戶無法於初期定義完整的規格，而開發人員於開發過程中也常常無法知悉外在環境或業務的變動，所以需要兩者密切合作方能開發適用的軟體。

（2）專案最終的目標是可執行的程式，因此所有的中間產品必須經過審慎評估，確認有助於最終目標，才需要製作中間產品。

（3）流程可以簡單，但規劃與執行必須嚴謹。

（4）強調團隊合作，賦予高度的責任，團隊有自主權得以因應變化做調整。

eXtreme Programming, XP 極限開發最早由 Kent Beck, Ward Cunnungham 與 Ron Jeffries 所提出，這樣的方法首次在 C3（Chrysler Comprehensive Compensation System）克萊斯勒綜合報酬系統的開發上使用，這個專案的負責人就是 Kent Beck。雖然 C3 這個專案並沒有獲得最後的成功，但是他們所提出的方法卻一直活躍在軟體開發的領域。

XP 極限開發可以說是敏捷開發 (Agile Software Development) 最受歡迎的一項手法。如果我們敏捷開發形容為絕世武功的心法，那

麼 XP 就是練武的招式，這樣的概念也可以套用在 CMMI（心法）這樣的軟體哲學，更確切的說 XP 是一種軟體開發模式。

2.3 聖天諾加密鎖

關於「聖天諾加密鎖」的部分，在軟體的版權保護上最為多人使用，其為新一代 ASIC 技術，其除了可以提供與程式的互動外，還能將程式的核心寫在其所擁有的記憶體，使軟體的版權防護達到更高一層的效益。

它的獨立記憶體，可以做多種方式的使用。如：用來作為租賃軟體的使用次數計數器、相關隱私權較高的使用者相關訊息、帳號、密碼。開發人員可以把獨有的演算法放在軟件保護鎖，亦可在加密後的應用程序在運行時，向保護鎖發送查詢數據串，對其回傳的資料進行判斷。在本研究中，加密鎖被拿來做為執行時期身份確認所使用，系統在執行的生命週期中，將對本鎖進行存在與否的檢查。

2.4 SOA（service-oriented architecture）

SOA[12]是一種架構模型，由網站服務技術等標準化元件組成，目的是為企業、學校或提供網路服務單位建構一個具彈性、可重複使用的整合性介面，促進內外部如內部應用程式、用戶、與部門（系所）等相關單位完美的溝通，盡速達到網路服務提升的目標。

何謂 SOA？首先讓我們釐清 SOA 的迷思。正確來說：

（1）SOA 不是新玩意：多年前即有資訊部門或公司成功地用 SOA 方式來建構、運行應用程式，且當時 XML、web service 都尚未提出。

（2）SOA 不是種技術：它是種建構、組織的方法，用來建立應用程式的運行環境，以及讓學校的業務程式能以「功能化」方式發展、累積。

（3）就算購買最新的 XML、web services 產品（如開發工具、執行平台、軟體元件等），也不表示就可以建構出 SOA 式的應用程式。

簡單來說，SOA 是一種遵循典範，是針對應用程式的設計、開發、布建及管理所提出的遵循典範。要實現 SOA，需要程式設計師改採「持續累積服務」的觀念與角度來開發應用程式，或許即便這麼做在短時間內看不到顯著好處，程式師還是必須跳脫、超越過往對應用程式的想法，改以「既有服務可否再運用？」或者是「能否沿用其他同仁開發過的服務再建構？」的觀點來面對程式開發。

SOA 主張「程式開發技術」與「程式建構方法」的交替並用，以類似傳訊溝通的作法，將數個所需的「業務服務」進行連結，以此來

實現一個新的應用程式，而非「從頭開發」。透過適當的程式組構及傳訊式的程式連結，可讓用戶的需求與改變，新的應用程式只要透過「傳訊微調」即可實現，而非「重新撰寫」。

SOA 不單只是程式開發的方法論，也提供行政管理層面的依循。例如它並非是以應用程式個體為角度來進行管理，而是直接將過往程式師開發出的程式視為「服務」來管理。而對「服務」間的「互動傳訊」進行分析，SOA 便可讓程式設計部門的主管瞭解何時該執行哪個業務邏輯，以及為何要執行，如此資訊管理者與分析師便可對服務程序進行最佳化調適。

SOA 如何運作？

SOA 服務導向架構是一種新興的系統架構模型，主要概念是針對學校或企業需求組合而成的一組軟體元件。組合的元素通常包括：軟體元件、服務及流程三個部分。當面對外部要求時，流程負責定義外部要求的處理步驟；服務包括特定步驟的所有程式元件，而軟體元件則負責執行工作的程式。SOA 已成為現今軟體發展的重要技術，透過 SOA 讓異質系統整合變得容易，程式再使用度也提高。不必自行開發或擁有所有程式元件，發展者可以視其需要組合網路上最好的服務。不受限於特定廠商的產品功能或是平台，達到真正的開放性

（Openness）。從分散式元件架構到 SOA 概念上，SOA 如同物件導向、軟體元件等軟體技術一般，運用小的零組件組合成應用系統。但 SOA 強調的是如何將彼此關係鬆散的應用系統功能元件在網路上發行、組合及使用。SOA 具有下列技術特性：

（1）分散式架構（distributed）－SOA 的組成元件是由許多分散在網路上的系統組合而來，可能是區域網路，也可能是來自廣域網路。例如網站服務技術（web services）就是運作 HTTP 來相互連結的 SOA。如此的作法，也使得網站服務技術很快的就成為所有支援網際網路的系統平台均能使用的技術。

（2）關係鬆散的界面（loosely coupled）－傳統系統主要是將應用系統功能需求切割成相互關聯的小零組件（模組、物件或元件），發展者要花費極大的心力瞭解零組件是如何設計及使用，以確保不會違反零組件連接關係限制。如此一來，若要以不同零組件替換原始設計，就成為一件困難的事。SOA 的作法是以界面標準來組合系統，只要符合界面要求，零組件可以任意替換，藉以大幅提高系統變更的彈性度。

（3）依據開放的標準（Open standard）－使用開放標準是 SOA 的核心特色，過去的軟體元件平台如 CORBA、DCOM、RMI、J2EE 採用專屬協定作為元件連結的規範，使得不同平台的元

件無法相通。SOA 則著重於標準與互動性，將可避免不同平台（.NET web services 與 Java web services）開發程式間相互整合的困擾。

（4）以流程角度出發（process centric）－在建構系統時，首先了解特定工作的流程要求，並將其切割成服務界面（包括輸入與輸出資料格式），如此其他的發展者就可以依據服務界面開發（或選擇）合適的元件來完成工作。

因此 SOA 的實作，就是將所有程式邏輯及服務內容全部包裹在服務內部，並實作一個標準的介面與外部作溝通，這種做法跟傳統的元件導向做法非常類似，唯一的差別是介面定義的方式、資料格式、與溝通管道必須是產業標準（http、XML、SOAP 等），也就是說只要能實作出如此的介面，不論介面後面是什麼，都可使成為 SOA。

綜合以上的介紹，SOA 能帶來的幫助，有以下好處：

（1）增加企業盈收，或提升學校的服務品質。

（2）提供可變動的網路服務型態。

（3）降低開發的成本。

（4）降低開發服務的時間。

（5）整合網路服務技術資源。

（6）降低整體風險及意外。

SOA（service-oriented architecture）的架構是近年來最受矚目的架構，從企業應用整合（EAI, Enterprise application integration）、企業與企業間的整合（B2B, Business-to-Business）以及企業流程管理（BPM, Business Process Management ）大致上都漸漸的採用 SOA 架構來實作，其原因在於 SOA 強調低耦合（Loosely coupled）的界面開發，使的開發時可將功能元件化（Component-based），且現今的企業大部分都是分散式的架構來設定系統環境，講求資源能夠有效的運用，這使得 SOA 很容易被挑選來使用於整合企業所能使用到的資源。加上 web service 技術的成熟，讓客戶端（Client side）可以透過各種方式來達到效果，舉凡如 HTTP、XML、SOAP、WSDL、UDDI、JMS 對服務端（Server side）所進行的服務要求。

2.5 MD5 (Message-Digest Algorithm 5)

用於確保資訊傳輸完整一致，是電腦廣泛使用的雜湊演算法之一（又譯摘要演算法、雜湊演算法）。主流程式語言普遍已有 MD5 實作。MD5 雜湊長度通常為 128 位元，隨著電腦運算能力提高，找到「碰撞」是可能的。因此，在安全要求高的場合，一般不使用 MD5 來做為完全性的保證。在本系統，主要是用於比較使用者的身分認證

及相關的系統必要參數設定的完整性，並沒有使用在與個人資訊相關的資料加密。

2.6 AES (Advanced Encryption Standard)

高階加密標準（Advanced Encryption Standard，AES），是美國聯邦政府採用的一種區塊加密標準。這個標準用來替代原先的 DES，已經被多方分析且廣為全世界所使用。AES 在軟體及硬體上都能快速地加解密，相對來說較易於實作，且只需要很少的記憶體。正因為這個速度快且又是個國際性的加密標準，所以本研究採取他做為資料加密的方式，以避免在大量資料做運算時，使系統的效能下降[13]。

2.7 物件導向技術

物件導向程式語言是目前於業界、學界都主推的核心基礎技術，維基百科對於物件導向程式設計的定義如下：物件導向程式設計（英語：Object-oriented programming，縮寫：OOP），指一種程式設計典範，同時也是一種程式開發的方法論。它將物件作為程式的基本單元，將程式和資料封裝其中，以提高軟體的重用性、靈活性和擴充功能性。當我們提到物件導向時，它不僅指一種程式設計方法。它更多意義上是一種程式開發方式。在這一方面，我們必須瞭解更多關於物件導向系統分析和物件導向設計（Object Oriented Design，簡稱 OOD）方面的知識。

何謂物件導向技術/方法：

（1）將真實世界視為由各種物件所構成；以人類直覺的方式將問題模擬成各種物件。

（2）真實世界的運作是由各個物件之間的互動來產生。

（3）將真實世界的問題對應（抽象化）成分析設計的模型，再轉換成相對應的程式碼。

（4）將實際生活中所遇見的物件應用於軟體設計裡，進而演化成的一種軟體發展的模式。

物件導向技術有三種重要觀念：物件、訊息和類別。簡單描述物件導向的三種重要觀念：

（1）物件（Object）：提供資料和處理資料程序（Java 語言是方法）的封裝。

（2）訊息（Message）：在物件之間的溝通方式，可以建立互動和支援多型。

（3）類別（Class）：物件的分類，可以實作類別架構的繼承。

物件是物件導向技術的關鍵，以程式的角度來說，它是電腦用來模擬現實生活的東西或事件，也是組成整個程式的元件。物件是資料與相關處理資料的程序和函數結合在一起的組合體，資料就是變數，程序和函數在 Java 語言稱為方法。

物件的方法是對外的使用介面，資料和相關方法的實作程式碼都包裹隱藏起來，稱為「封裝」(Encapsulation)。對於程式設計者而言，並不用考慮物件內部方法的程式碼是如何撰寫，只需要知道這個物件提供什麼介面和如何使用即可。[14]

隨著物件導向的發展，愈來愈多人使用物件導向程式設計，而現今主流為微軟（Microsoft）的 Microsoft Visual Studio .NET 與 Java 兩大陣營，.NET 與 Java 同時都有著相同的特色，一個虛擬的執行環境及豐富的函數庫。

Common Language Runtime（簡稱 CLR）：CLR 為.NET Framework 的基礎，是一種以物件導向為核心的虛擬程式執行環境，可以將 CLR 視為程式在執行時期的管理程式碼的代理者，提供如記憶體管理、執行緒與遠端處理功能等核心處理程序，在 Java 的環境，這個則稱之為 JVM（Java Virual Machine）。其開發出來的程式並不被編譯成為能夠直接在電腦上執行的二進制原生代碼。與 Java 相同的，它被編譯成為中間代碼（Microsoft Intermediate Language），然後透過.NET Framework 的虛擬機器執行（如圖 1 所示）。

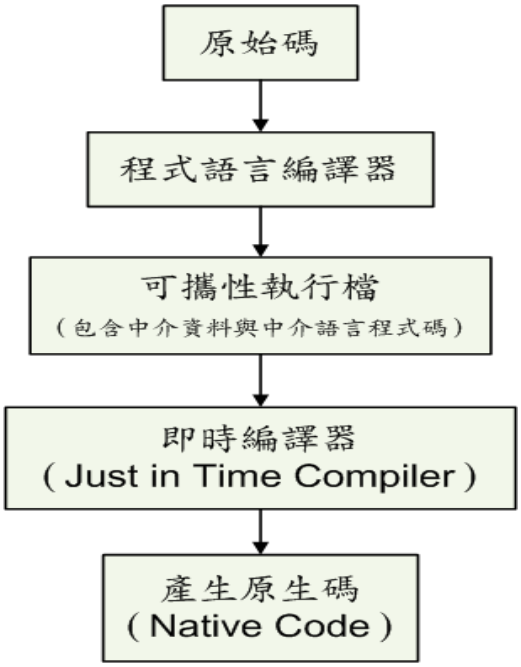

圖 1.CLR 在程式時期所扮演的角色

在程式執行時，虛擬的執行環境將中間代碼翻譯成為二進制機器碼，從而使它得到正確的執行。最終的二進制代碼被儲存在一個緩衝器（Buffer）。所以一旦程式使用了相同的代碼，將會呼叫緩衝器的版本。這樣如果一個.Net 程式第 2 次被執行，那麼這種翻譯不需要進行第 2 次，速度明顯加快。

第三章　系統架構與設計

在本節中，我們提出網路傳真掃描軟體的系統架構，分成幾個部分：(1) 平台架構 (2) 系統架構 (3) 安全性檢查機制架構，並說明之。

3.1 平台架構

本研究實作之平台- Internet Fax Platform (簡稱 IFP)，其平台架構圖如圖 2 所示。

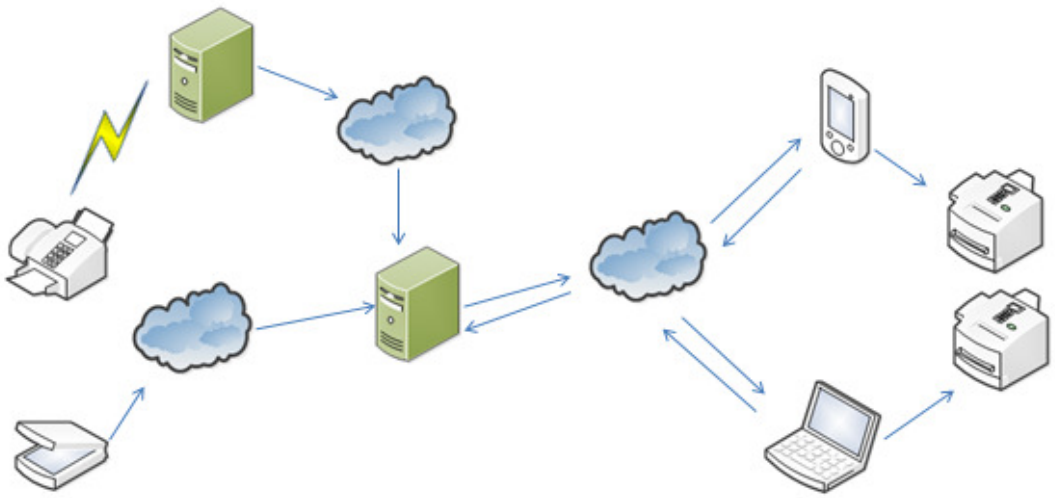

圖 2.IFP 平台架構圖

平台內容包含以下部分：

(1) 發送端：主要使用傳真機、掃描機或是事務機來將其所需要的資料發送到主機端。當發送端是傳真機時，其傳遞資料的方式則是將資料轉成類比的信號後，發送到專門接送類比訊

號的伺服器，再藉由該伺器將其類比信號轉成 tiff 的數位圖形檔，發送到資料儲存伺服器。若發送端是掃描器或是事務機時，則是利用該設備本身所具有的能力，將所要傳遞的資料數位化後，利用 IFP 所提供的掃描端專用軟件，將資料傳遞至儲存伺服器。

（2）伺服器端：伺服器的組成有類比資料接收伺服器、資料儲存伺服器及應用程式伺服器三種，類比資料收接伺服器主要是在接收發送端使用傳真機發送上來的類別信號，其會將這種信號傳遞過來的資料轉成圖形檔後，再轉送到資料儲存伺服器上。資料儲存伺服器主要的功能是在於接收由發送端是掃描器、事務機或是類比伺服器傳送而來的數位信號。應用程式伺服器則是負責與接收端的 client 端程式進行身份及系統的驗證。

（3）接收端：接收端會有一支可跨作業系統執行的應用程式，其相關的功能即是將存放於儲存伺服器上的資料進行管理及取回，程式在啟動及運行的過程，將會一直與應用程式伺服器進行溝通，當發生重複開啟或是身分認證失敗的情況時，將會中止程式的執行，直到進行修正相關的問題。

3.2 系統架構及說明

接收端使用的系統為「網路傳算管理系統 Internet Fax Scan Management System」（簡稱 IFSMS）的系統架構圖如圖 2 所示。

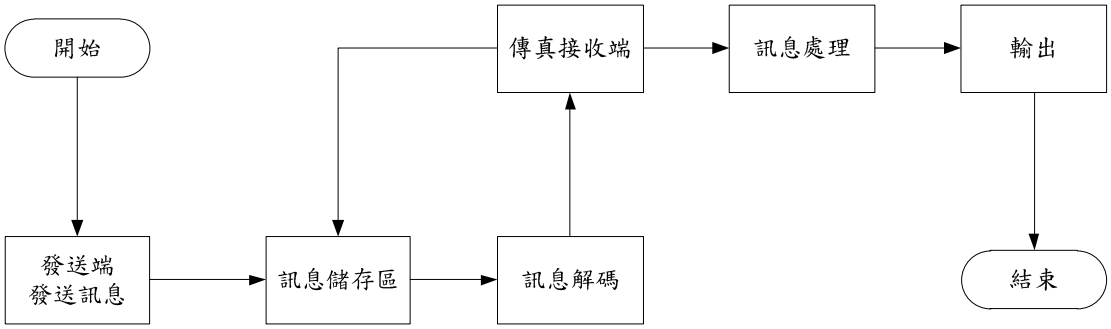

圖 3.系統流程圖

傳真接收端是系統的主要處理核心，所有的流程都是以其為中心在進行調整及處理（完整流程如圖 3），其說明介紹如下：

（1）傳真發送端分為傳真機使用及掃描器使用，傳真機使用會經由行動傳真碼將所要傳真的資料送到訊息儲存區，行動傳真碼一般由電信相關公司所提供，其在內主要是一個可以用來將類比轉為數位的伺服器，當其進行轉換後的數位檔案才能讓系統中的儲存伺服器可以對其進行儲存的動作，不過由於伺服器設計上的限制，其並沒有辦法將一通傳真分別轉成不同的數位檔，而是以通為單位來進行轉換，此時如果發生轉誤失敗的情況，則整通傳真不分張數全部都會遺失。若為掃

描器的使用者，則其將會透過一個前端的掃描器服務軟體來將所掃描產出的檔案經由網路送至訊息儲存區。

（2）在資料進到訊息儲存區時，會經由安裝在儲存區的軟體程式，對進來的資料進行加密的動作，以防止資料在儲存的過程中被有意人事拿取，同時在這個時候，會取得發送端上的相關轉發訊息，是否有要給其他的接收端用戶，當判斷有相關訊息時，會在儲存區轉發到其他接收端的暫存區中。

（3）傳真接收端將會向負責訊息儲存的 Server 發出資料取得的要求，其會包含著此接收端所擁有的帳號、密碼、權限及資料時間範圍等，其資料取回後，會將伺服器端的資料進行刪除，如果特別有需要備份的需求，需在伺服器端另外處理。

（4）當訊息由 Server 端取回後，傳真接收端將會針對其被打包的資料行取出及解碼轉檔的動作，並且將處理完的包裝檔進行刪除。

（5）原始格式產生後並不能直接進行讀取，其需要再經過一道轉檔並加工的手續，將檔案轉成一般常見的檔案類型，並在轉檔後，依系統的要求將其進行格式化的處理，如浮水印、表頭資訊調整等，資料存在本地端的是否加密的情況取決於使用者本身的需求，若要求以加密狀態存在，則資料要檢視時，

一定需要經過傳真接收端的軟體來進行解密檢視的動作，若不需要加密，則會以 jpg 圖形檔案格式儲存於本地端中。

最後將已經處理完畢的檔案依其傳真的來源方、件數及張數等資訊，於列印時動態的寫於所產出檔案的最前端，以供使用者在閱讀時判別之用。

3.3 安全性檢查架構

安全性的檢查分成傳真發送端及傳真接收端兩個部分，在傳真發送端的部分如果是使用掃描器系統的，其程式在啟動時會做傳送資料檢查。而傳真接收端的系統啟動時，會有做一系列的啟動檢查，其主要是用來要求在同一個執行期，相同的帳號一次只能有一個實例是屬於啟動狀態，避免因網路的便利性，而導致接收端無法正確收到其所屬的資料。

3.3.1 傳真發送端

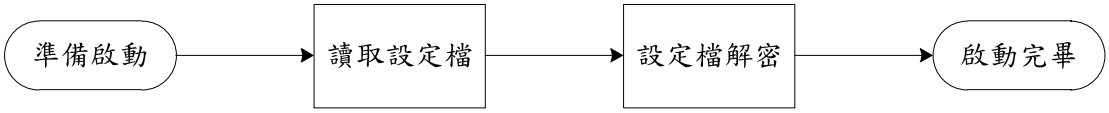

圖 4.掃描發送端程式啟動流程

掃描發送端啟動時，會將系統設定檔的相關資料讀入（系統啟動的流程如圖 4），設定檔的編碼來源為「系統設定檔產生器」，其會依據使用者的需求（如對應的帳號、密碼）及相關資訊進行 AES

（Advanced Encryption Standard，簡稱 AES）加密計算（其用於將資料加密的程式碼如圖 5），當計算產生後的編碼資料將會寫在一個系統的參數檔，再與軟體一同交由使用者（其加密後的內容如圖 6），當程式啟始時，則會把其已經編碼的資料進行解密動作，並將其相關的資訊反應到畫面與使用者互動（使用於解密的程式碼如圖 7），考慮到傳真發送端的使用者大多為對電腦使用不甚深入的使用者，考量其上手的適應性及執行效能，這個部分使用 MS Visual Studio C#進行開發。

```
/// <summary>
/// AES加密
/// </summary>
/// <param name="plainStr">明文字符串</param>
/// <returns>密文</returns>
public static string AESEncrypt(string plainStr)
{
    byte[] bKey = Encoding.UTF8.GetBytes(Key);
    byte[] bIV = Encoding.UTF8.GetBytes(IV);
    byte[] byteArray = Encoding.UTF8.GetBytes(plainStr);
    string encrypt = null;
    Rijndael aes = Rijndael.Create();

    using (MemoryStream mStream = new MemoryStream())
    {
        using (CryptoStream cStream = new CryptoStream(mStream, aes.CreateEncryptor(bKey, bIV), CryptoStreamMode.Write))
        {
            cStream.Write(byteArray, 0, byteArray.Length);
            cStream.FlushFinalBlock();
            encrypt = Convert.ToBase64String(mStream.ToArray());
        }
    }
    aes.Clear();
    return encrypt;
}
```

圖 5.將資料進行加密

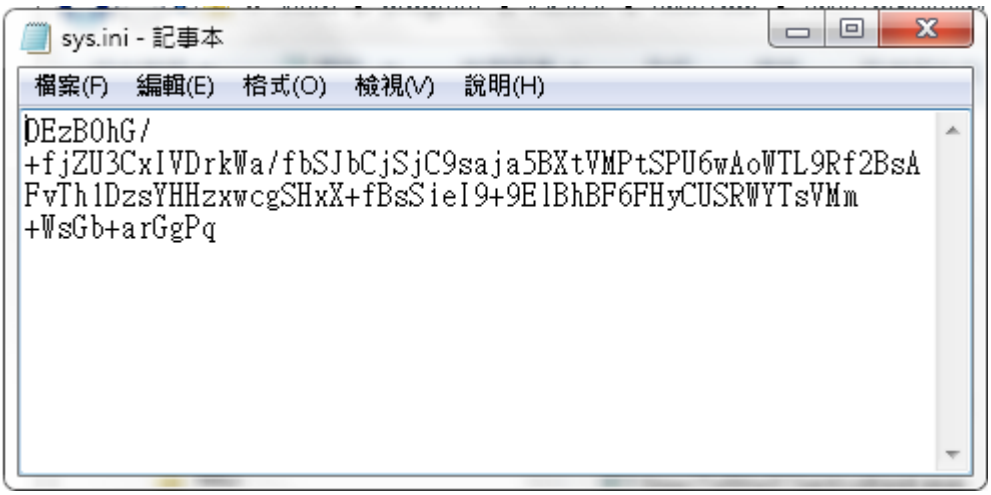

圖 6.加密後的密文資料

```
/// <summary>
/// AES解密
/// </summary>
/// <param name="encryptStr">密文字符串</param>
/// <returns>明文</returns>
public static string AESDecrypt(string encryptStr)
{
    byte[] bKey = Encoding.UTF8.GetBytes(Key);
    byte[] bIV = Encoding.UTF8.GetBytes(IV);
    byte[] byteArray = Convert.FromBase64String(encryptStr);
    string decrypt = null;
    Rijndael aes = Rijndael.Create();
    using (MemoryStream mStream = new MemoryStream())
    {
        using (CryptoStream cStream = new CryptoStream(mStream, aes.CreateDecryptor(bKey, bIV), CryptoStreamMode.Write))
        {
            cStream.Write(byteArray, 0, byteArray.Length);
            cStream.FlushFinalBlock();
            decrypt = Encoding.UTF8.GetString(mStream.ToArray());
        }
    }
    aes.Clear();
    return decrypt;
}
```

圖 7.將資料進行解密

3.3.2 傳真接收端

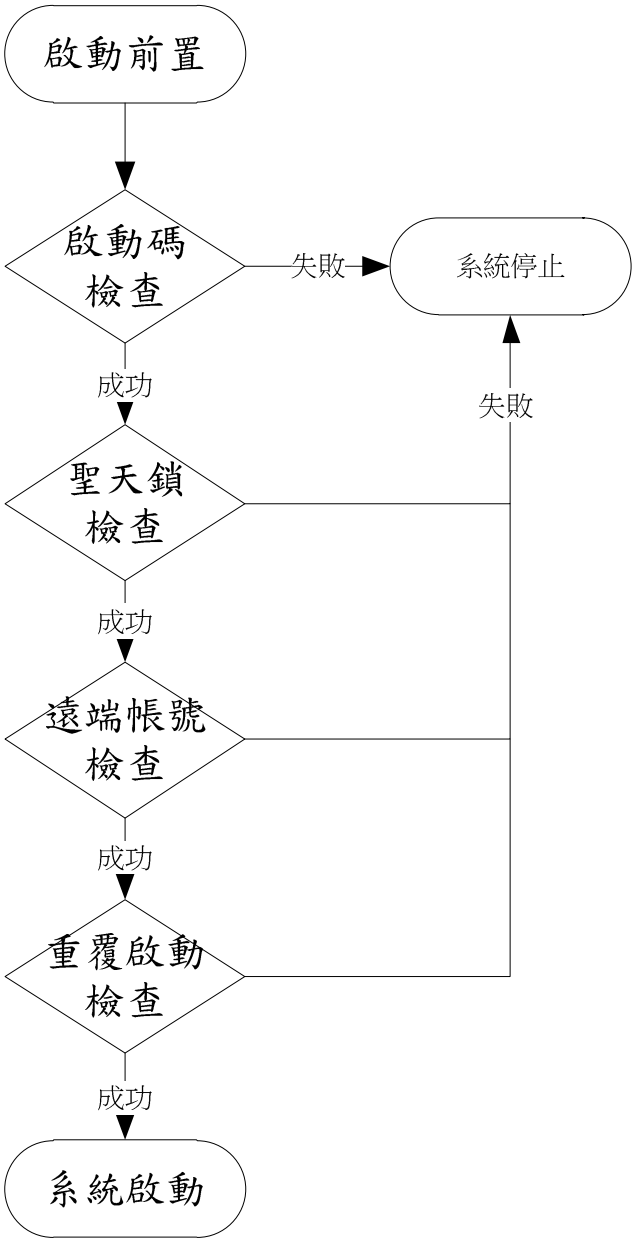

圖 8.安全性檢查流程圖

傳真接收端一般由提供服務的公司所使用，由於考量到提供服務的公司端大多有一定的資訊能力及架構要求，所以其開發語言採用 Sun Java 來進行開發，以提供給使用非 Windows 視窗的 OS 主機使用（系統的啟動流程如圖 8），其相關的說明如後：

（1）啟動碼檢查：系統在啟始時，會針對所載入的帳號、郵件伺服器、連結的輸出設備的數量及一個用於計算的種子 key 進行啟動碼的計算，如果計算的結果與系統設定檔裡的啟動碼不相符的話，則拒絕系統的啟動（用於判斷的計算程式碼如圖 9）。

啟動碼的計算主要是使用java功能，主要是將計算過後的物件進行赫序演算法的計算（hash code algorithm），其針對物件進行計算後，會得出這個物件特有的一段赫序碼（hash code），程式在啟動時就會將目前參數所記錄的相關資訊進行演算，並和記錄的赫序碼（hash code）進行比對，依照赫序碼（hash code）的原理「如果兩個物件相同，那麼它們的hash code值一定要相同；如果兩個物件的hash code相同，它們並不一定相同」可得知，當計算來源的物件為相同時，其產出的值也應該相同，所以系統可由此來判斷其是否為正確的啟動碼，但雖然由另一個定義可以知道，在不同的資料物件計算下，也是有可能產生同樣的赫序碼（hash code），但是以其發生的機率與設計一個難破解的演算法，或是評估一個能阻檔攻擊的演算法所需花費的投資報酬率來評估，是可以忽略不計的。

```
if(mail != null &&
        mail.length() > 0 &&
        server != null &&
        server.length() >0 &&
        pwd != null &&
        pwd.length()>0 &&
        "123456789".equals(pwd)){

        int code = ("######"+mail+server+"#####").hashCode();
```

圖 9.判斷資訊正確性程式碼

(2)通過了一開始的啟動碼檢查後，會進行版權控管的硬體檢查，其將會與硬體鎖「聖天諾加密鎖」進行互動，系統將會每 2 分鐘偵測 1 次硬體鎖的存在與否，當硬體鎖不存在時，則禁止系統使用(圖 10 為使用來判斷硬體鎖是否存在所使用的程式碼)。

```
private boolean checkKeyExist() {
    SentinelKeys keys = new SentinelKeys();

    int[] softwareKey = {
            0x18, 0x9B, 0x30, 0x5E, 0x19, 0x7E, 0xFA, 0xD8, 0x5E, 0xA6, 0x1A, 0x6A, 0x3D, 0x77, 0x9C, 0x80,
            0x27, 0xAC, 0x8E, 0x85, 0xA2, 0x4F, 0x74, 0xB5, 0x51, 0xD0, 0x3E, 0x86, 0x56, 0xC5, 0xC6, 0x34,
            0xC9, 0x94, 0xE9, 0x0A, 0x39, 0xE3, 0x71, 0xD8, 0xFA, 0xA7, 0x18, 0x3A, 0x78, 0x43, 0x1A, 0x95,
            0xD0, 0x58, 0x22, 0x3B, 0x58, 0x86, 0xE2, 0x7B, 0x7D, 0x81, 0x80, 0x93, 0x8E, 0x94, 0x9B, 0xC4,
            0x65, 0x98, 0xA2, 0xED, 0x96, 0x5D, 0xF4, 0x75, 0xEC, 0xC8, 0x2A, 0xA1, 0xA3, 0x4A, 0xDF, 0xDF,
            0x00, 0xCD, 0x4D, 0x6F, 0x16, 0xDD, 0xC0, 0x1B, 0x24, 0x9E, 0x66, 0x1A, 0x5F, 0xDC, 0x5B, 0x68,
            0x74, 0x4C, 0x64, 0x38, 0x1C, 0x87, 0x17, 0xA3, 0x16, 0x5B, 0x47, 0xF0, 0x18, 0xC5, 0xD2, 0x63
    };

    long    DevID       = 0x2EB92EB9;
    long    licID       = 0x6F26;
    long    Flags       = 65;
    int[]   licHandle   = new int[10];
    int     status      = 0;
    byte[]  softKey     = new byte[112];
    int     i           = 0;
    for (i = 0; i < 112; i++) {
        softKey[i] = (byte) softwareKey[i];
    }

    status = keys.SFNTGetLicense(DevID, softKey, licID, Flags, licHandle);

    return status == 0;
}
```

圖 10.檢查硬體鎖程式碼

（3）當傳真接收端啟動系統的檢查都通過後，會將帳號及自動產生的 1 組與主機資訊相關的 key 傳送到遠端的 Web Service Server 上，其會用來判斷是否同 1 帳號會有多開的情況，並檢查是否有在不同的主機上執行同 1 個帳號的登入。當發現 1 個帳號有多個登入者時，會保留最新的登入者，且將舊有的登入者剔除（其用於判斷的程式碼如圖 11）。

```
//看有沒有在sesion裡
if(data == null){
        //沒的話，就新增
        application.setAttribute(id, sys_id);
}else{
        //是不是一樣的sys_id，不是就換成新的，等舊的上來就把他給關了
        if(!sys_id.equals(data)){
                existId.add(data);
                application.setAttribute(id, sys_id);
        }
        if(existId.contains(sys_id)){
                existId.remove(sys_id);
        }
```

圖 11.判斷是否有多重登入

第四章　系統實作

本軟體主要是針對於傳真的使用範圍及領域，提供一種新的應用思維，其也包含了發信方及收信方跳脫現有使用方式的操作方法，其來源及接收端並非一定要是傳真相關設備，且由於其接收端開發的語言為 JAVA，因此可以簡單的移轉到其他的平台上（如掃描機、傳真機、電腦、手機等），都能成為資料的發送端，而印表機、電腦、傳真機、手機等，都能成為資料的接收端，而且發送端以 C# 來做開發主要是為了考量提升一般使用者的操作便利性。

4.1 掃描發送端

掃描發送端的程式分為兩種，一為產生系統軟體所需要的啟動設定檔，另一個則為發送端的主程式。

4.1 設定檔產生器

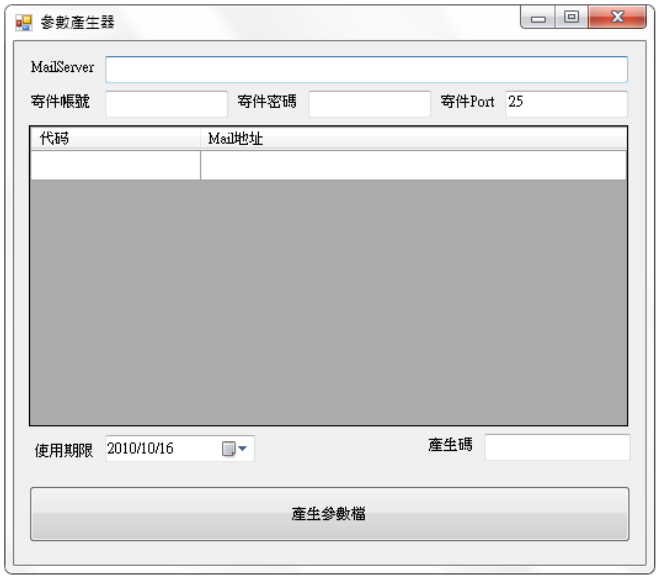

圖 12.系統設定檔產生器

產生器的主要功能是用來產生發送端所需要的系統參數檔（系統相關畫面如圖 12），其中比較重要的是寄件人及收件人的相關資訊，「使用期限」的功能為用來做為商業用途的管控點，而產生碼則是功能要執行前的判斷，主要用於避免程式外流後，造成公司的損失。系統將所有的資訊蒐集後，會直接把物件進行序列化（serialization）的動作，再交由 AES 編碼的函數來對其加密，並寫入純文字檔，為了商業因素的考量，這個部分並不能直接使用明碼來做為資料交換，所有的資料產生皆需要透過同一個資料來源，且其資料是需要被確認、驗證過後，才能確保使用者的使用是受到管控的。使用序列化的因素有二，一是在操作上，只需要還原成原先的物件後，就可以在發送端的程式使用其相關的資料，避免了重新建立物件所花費的時間；另一是如果有異動，會出現還原失敗的錯誤訊息，這個也能達到一定程度

的防護能力。

4.2 掃描發送器

圖 13.掃描發送器

發送器的主要功能是掃描器的使用者端，能將所要傳送的資料進行轉發的動作（其操作的使用者介面如圖 13），在程式的啟動時會將設定檔進行讀入的動作，經過了 AES 解密的動作後，會將資料進行反序列化（Deserializer），並還原成物件，這主要是在於可以免除一些物件操作的初始化及相關設定，並提升系統的執行效能。在發送器的部分設計成每經過一段時間後，會主動去檢查比較「來源路徑」及「處理後文件路徑」兩個資料夾裡面的值，其會將「來源路徑」下的檔案減去「處理後文件路徑」下的檔案後，所剩下來的當成本次所要發送的資料來源，在發送時，會取得操作介面右邊挑選的對像，目前有支援同時發送多人的部分，將資料傳到目的端。在傳送完畢後，並將此次有發送的資料由「來源路徑」移到「處理後文件路徑」。

其寄送的相關資訊都已由設定檔產生器先行設定完成，系統在寄送掃描信件時會自動引用。在寄件的部分，系統設計成每掃描到 1 張新的傳真時，就立即發送 mail，以避免 1 次大容量的郵件被收件方的伺服器退件，且使用者在接送端的等待時間也不會過長，不過由於寄件的次數較為頻繁，有可能會被伺服器端當成是攻擊看待，所以需要由伺服器端對登入者的資訊及權限做特別的設定。

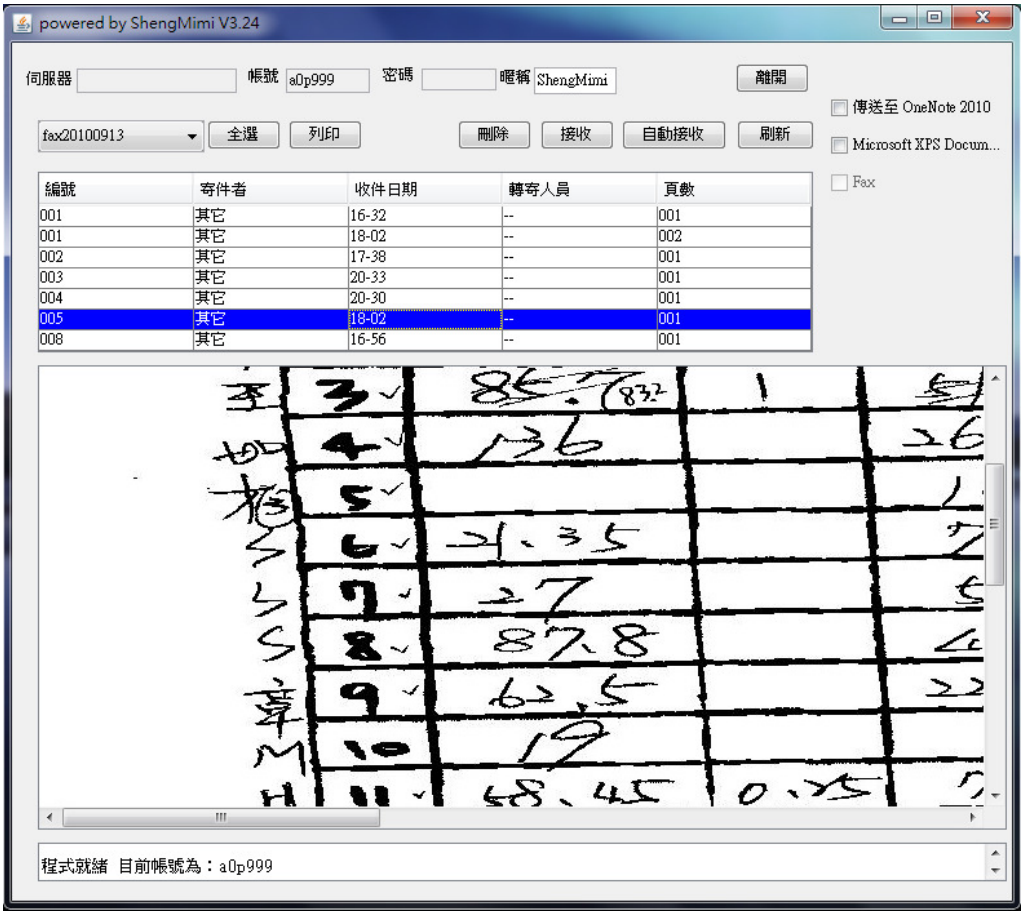

圖 14.軟體主畫面

圖14為傳真接收端的系統主畫面，其主要的功能即為用來接收其他資料來源端所發送的訊息，在介面上同時提供了對於目前帳號的提

示、對輸出對象的選擇、對所收取到的資料的清單及針對特定項目的預覽。當使用者點選了「自動接收」後，在系統解碼存檔時，會直接將收件的部分進行列印，如果此時挑選的輸出端有多數的話，則會對其進行輪流輸出的動作。「刷新」則是系統在進行處理時，如果不能即時的對畫面進行自動刷新，可以使用人工介入的部分進行處理。

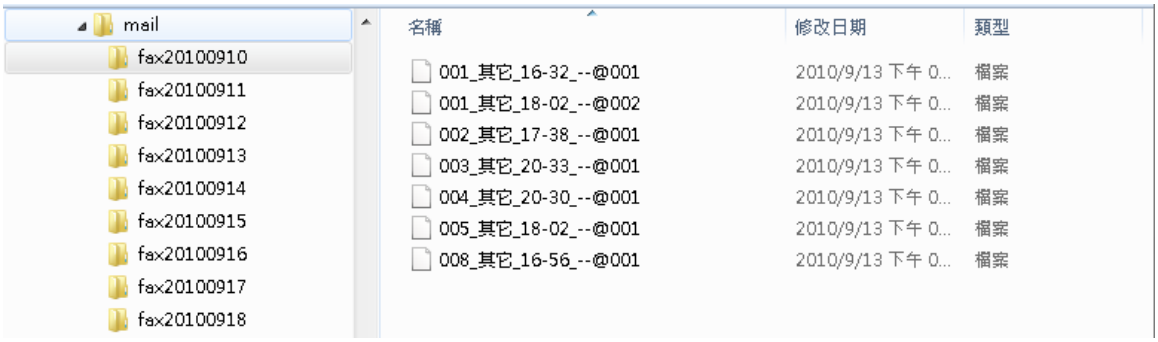

圖 15.檔案結構

目前將資料解碼並拆解後，其相關的資料將以完整的原始檔案儲存於目前的本地端，其主要目地是為了讓使用者可以簡易方便使用產出做不同使用，在命名的部分是檔案的目錄及檔名來做區分，其檔名的拆解方式為「來源通數_資料來源_時間_註解@張數」，通常一通傳真可能包含多個檔案，所以需要以張數來做為比對的依據及判斷（實體的檔案儲存方式如圖15）。

第五章　結論與未來展望

本研究所提出的相關實作功能，在目前現實的生活商業化的應用已經有一段時間，在使用這套系統的相關人員都對這個系統所提供簡化操作的功能及記錄的方便性有著深刻的印象，並常會在交流溝通時提出多種不同角度的看法及建議，其中比較適合大眾使用的部份將會在下一版的系統中進行實作的動作。

本文中系統的設計一開始主要是依據目前現有的傳真接收端的功能開發，進而依照使用者的需求進行延伸設計，進而達目前的眾多功能及規模，利用物件導向的程式設計方式來避免程式維護及延伸的負擔，使用 WEB 相關技術的概念達到資訊於網路上共享及交換需求，利用不同思維將傳統的傳真機使用方法做不同方式延伸，期許能為商業價值做出更高的貢獻，下表為目前使用一般傳真機與使用本軟體的相關比較表。

項目	一般傳真機	本軟體	備註
傳真佔線	佔用電話線	使用網路不佔線	
文件品質	不清淅且會失真	完全與傳出的文件相同	
資料安全性	無保護	可加密	

單次傳真可接收量	1 張	200 張	
單張傳真接收時間	1 張約 10 秒	1 張約 5 秒	取決於印表機速度
同時可輸出量	1 張	至多 7 張	

表 2.傳真機與傳真軟體比較表

目前的設計為單一應用程式對應 1 組帳號密碼及電話，在目前開發中的次一版本，將會實作整合性的操作介面，同時提供 1 個介面多組帳號的操作方式，並考慮到眾客戶的執行環境，改用 C#實作，以期能在 windows 的平台下，產生最高的效益及降低使用者硬體設備等級的要求，並將同質性較高的掃描器也納入可使用的功能，目前能快速掃描的硬體設備已是越來越常見，相信不久的將來其市場及應用性將會遠遠的超過傳真機。

參考文獻

[1] G. Back and D. Engler, “MJ—A System for Constructing Bug-Finding Analyses for Java,” Stanford Univ. Computer Systems Laboratory, 2003.

[2]Yung-Chiang Huang, “Design and Implementation of a Web-Based Fax Server,” Yenchao, Kaohsiung, Taiwan, Republic of China, 2006.

[3]His-Jia Huang, “A Web-based Service Developing and Executing Platform using OSA,” National Cheng Kung University, 2004.

[4]Deng,W.,Yang,X.,Zhao,H.,Lei D., and Li, H.(2008) ,“Study on EAI Based on Web Service and SOA,” 2008 International Symposium on Electronic Commerce and Security,2008.

[5]Bajwa,I.S., Samad A., Mumtaz S., Kazmi R., and Choudhary, A(2009),”BPM Meeting with SOA: A Customized Solution for Small Business Enterprices,” 2009 International Conference on Information Management and Engineering ,Kuala Lumpur, Malaysia,2009

[6] Chih-Hung Huang, Cheng Wen and Kuang-Chiung Chang, “ Web-based Fax Server for Home or Small Business Use,” Lungwha Univ. of Sci. & Technol., Taoyuan ,2007.

[7] Lin, S.C. and Kao, T.-W. “FAX broadcasting system-The software aspects,” Automated Intelligent Syst. Inc., Torranace, CA, 1995.

[8] Pillai, P.; Campbell, J.; Kedia, G.; Moudgal, S.; Sheth, K. “A 3D Fax Machine based on Claytronics ,” Intel Res. Pittsburgh, PA , 2006.

[9] Jain, V.; Aravind, B.R.; Jagmohan, M.S., “Resynchronization techniques for V.34 FAX,” Texas Instrum. India, Bangalore, India ,

2004.

[10] Hui, S.C.; Chan, K.Y.; Leung, M.K.; Qian, G.Y., “A distributed fax messaging system ,” Div. of Comput. Eng, Nanyang Techol. Univ., 1995.

[11]林耀珍，敏捷的軟體開發流程，2003。

[12] 林傑斌、林清源、張太平，SOA 服務導向架構原理/方法/實務，2009。

[13] 粘添壽，資訊與網路安全技術（第二版），2009。

[14] 楊仁和，深入淺出物件導向分析與設計，2007。

網站：

中華電信 HiBox：

http://hibox.hinet.net

Microsoft Visual Studio .NET 平台簡介：

http://vr.me.ncku.edu.tw/xms/content/show.php?id=448

維基百科：

http://zh.wikipedia.org/zh-tw

微軟 MSDN：

http://msdn.microsoft.com/zh-tw/default

http://www.microsoft.com/taiwan/msdn/columns/soa/SOA_overview_2004112901.htm

簡西村，「服務導向架構應用」，

http://www.microsoft.com/taiwan/msdn/columns/soa/SOA_overview_2004112901.htm，2004 年