一個多層式架構流程控制的應用系統框架

An Application System Framework for Multi-Tier Structure Flow Control

李坤清
世新大學資訊管理系
副教授
Kun-Ching Lee Associate Professor IM Department, Shin-Hsin University
kclee@cc.shu.edu.tw

吳瑞堯
世新大學資訊管理系
副教授兼系主任
Rei-Yao Wu Associate Professor & Director IM Department, Shin-Hsin University
rywu@cc.shu.edu.tw

葉逸礽
世新大學資訊管理系
研究生
Yi-Jeng Yeh Graduate Student IM Department, Shin-Hsin University
gemini.via@gmail.com

摘要

流程控制是程式設計師必須處理的重要工作之一，但這些工作常因不良的設計容易讓系統邏輯散落於程式中的各個角落，這將嚴重影響系統的品質，增加系統維護的困難度，因此如何以透過一個良好的系統架構，並規範系統的流程與邏輯設計，使所有開發者遵循這個規範，如此將提昇系統的軟體品質，使系統達成一致性的目的，並更容易維護。

對於網路應用系統的開發，目前在市場上仍多以 HTTP 通訊協定透過應用伺服器來處理需求，其中 Struts 框架是一套採用 MVC Pattern 相當成熟的產品，在市場上也被廣泛的使用，但這類的產品只適合做一般網頁的處理，對於圖型介面的系統並無法在其架構下運行，目前企業中圖型介面系統仍然佔有一定的比例，但大多數仍採用傳統的系統開發方式來開發。

本研究提出一個不同的方式，為符合各種不同的程式語言，整合企業內部多樣化的應用系統，包括各種不同的 Web 系統或是 Client / Server 方式的 GUI 系統使用，因此改以 TCP / IP 為溝通的協定，再依循 MVC 樣式的規範，提出 GMVC (GUI Model - View - Controller) 框架，並以製造業的 RMA 系統為例，透過此 GMVC 框架來建置一套 GUI 方式的 RMA 雛型系統，並利用這個實作來驗證是否能被順利運作在 GMVC 框架之中。

關鍵詞：MVC 樣式、Struts 框架、GMVC 框架、RMA 系統

Abstract

Flow control is one of important works for the programmer must to do, but these works often have bad design let the system logic scatter in the program each corner, influences system's quality seriously, increases difficulty to maintain the system, therefore, how to use a good system structure to stipulate system's flow and the logical design, let all developers follow this rule, it will promote the system's software quality, archive consistency and will be easier to maintain.

To develop network application systems that were still used the HTTP protocol to communicate with the application server at present. Struts is a mature product to use MVC Pattern, it was used universally, but this kind of product is only suitable to process the web page, and GUI systems cannot be run under the structure, at present, the GUI systems still have certain proportion in the enterprise, but most of system developers still use the traditional system development way to develop.

This research proposes a different method, in order to let each kind of software language can use, integrate each kind of application system in the enterprise including web base system or client / server base GUI system, so we use TCP/IP protocol to communicate with server, then follows the MVC pattern rule, and we propose a GMVC (GUI Model - View - Controller framework and use the GUI RMA system as an example to let the RMA system process in GMVC framework to demonstrate the structure will be run smoothly by this way.

Keywords: MVC Pattern、Struts Framework、GMVC Framework、RMA System

1. 前言

1.1 研究背景

網際網路的活動已深入各個角落，從政

府、企業、學校、到每個家庭，從企業內部的流程控制，到各種商業活動、學術活動、技術交流、客戶服務等等，因此使用各種類型的應用系統開發工具或應用系統框架，幫助程式設計師做系統開發，並提升軟體品質的作法，已經成為應用系統最重要的開發模式。

流程控制是程式設計師必須處理的重要工作之一，但這些工作的處理相當困難，不良的設計容易讓這些處理邏輯散落於系統的各個角落，增加系統維護的困難度，因此，如何以一個良好的系統架構，規範這些控制工作的邏輯設計，決定了系統的軟體品質。

1.2 研究動機與目的

在網路上主要是透過 HTTP 通訊協定，以 Request / Response 的模式完成溝通 ，在互動的過程中，使用者透過瀏覽器對系統發出需求，而系統則針對需求內容回應相對的結果，透過如此不斷反覆重送與接收來完成全部的系統需求。

對於網路應用系統的開發，目前在市場上有相當多的樣式及框架可供選擇，仍多以 HTTP 通訊協定透過應用伺服器來處理需求，其中 Struts 是一套採用 MVC Pattern 相當成熟的產品，在市場上也被廣泛的使用，但由於 Struts 這類的產品所採用的 MVC 實作樣式，只適合做一般網頁的處理，對於透過圖型介面開發的系統並無法在其架構下運作，而目前企業中圖型介面開發的系統仍然佔有一定的比例，大多數仍採用傳統的系統開發方式來開發。

因此，本研究提出一個有別於傳統的方式，為符合各種不同的程式語言，整合企業內部多樣化的應用系統，包括各種不同的 Web 系統或是Client / Server 方式的 GUI 系統使用，並提昇系統的即時性，改以 TCP / IP為溝通的協定，再依循 MVC 樣式的規範，提出 GMVC (GUI Model - View - Controller) 框架，並以製造業的 RMA 系統為例，透過此 GMVC 框架來建置一套 GUI 方式的 RMA 雛型系統，並利用這個實作來驗證是否能順利運作在 GMVC 框架之中。

2. 文獻探討

本章文獻探討主要針對 MVC 設計樣式及 Struts 的功能特性，另外還會介紹一些其他較常用的樣式如 DAO (Data Access Object)、VO (Value Object)等。

2.1 Model-View-Controller 功能與特性

Sun (昇陽電腦)在開發 Web 應用程式的技術，提出了 JSP Model 1 和 JSP Model 2，圖2-1為 JSP Model 1 的架構，其中 JSP 負責處理由用戶端所傳送過來的 request 與送回給用戶端的 response，資料的存取是透過 Java bean 處理。

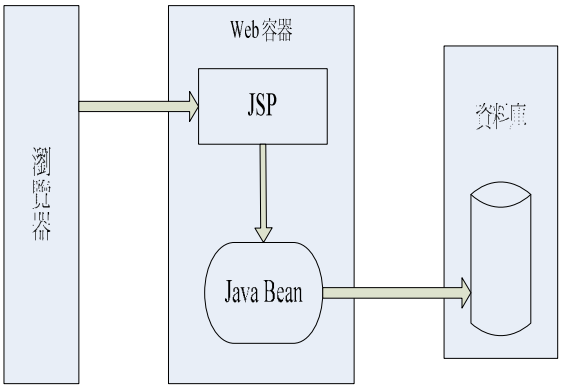

圖 2-1　JSP Model 1 [12]

但這樣的架構對於較複雜的應用程式，網頁中還是充滿了許多 JAVA 的程式碼與 Script 語言，商業邏輯與網頁呈現還是耦合在一起，對於程式開發人員和製作網頁人員來說都是相當不容易維護的，因此 Sun 另根據 MVC 樣式提出另一種模型 JSP Model 2 來解決商業邏輯與網頁呈現的耦合問題。

JSP Model 2 的架構 (如圖2-2)，Servlet (Controller) 可以用來處理控制流程，並負責處理由用戶端傳送過來的請求，然後依所需的請求選取適當的 JavaBean (Model) 元件，跟資料庫溝通後取得資料，再根據使用者所作的動作來決定該轉送到哪一個 JSP (View)，並且把 Servlet 中所得到的資料放入 JSP 中，JSP 只是單純做網頁的呈現，並沒有處理商業邏輯。

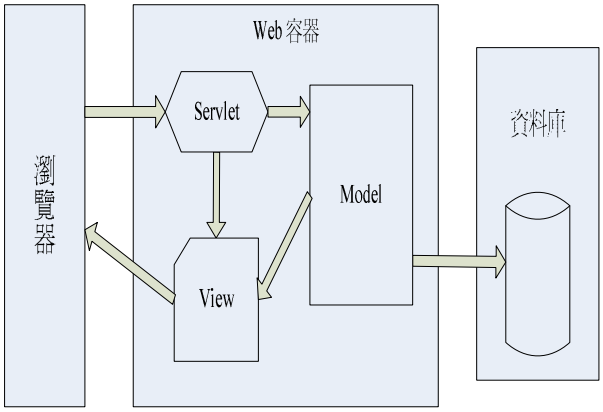

圖 2-2　JSP Model 2 [12]

2.2 Struts 應用系統

Struts 是以 Model 2 為基礎的 Web 應用軟體架構，核心技術包括 Java Servlets、Java Beans、XML、JSP、Tag Library 等等，圖2-3 為 Struts 主要元件展示。

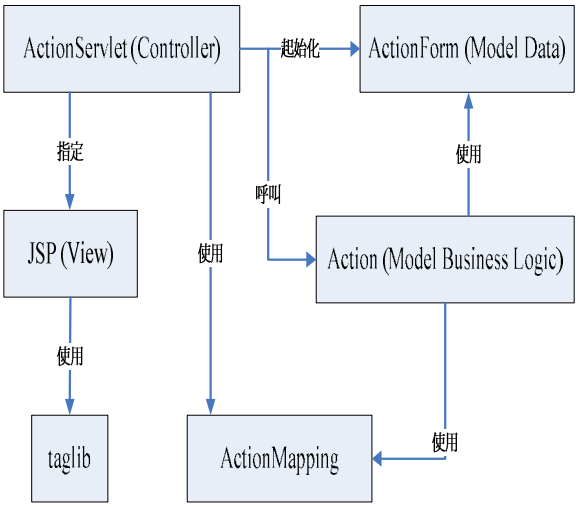

圖 2-3 Struts 主要元件 [5]

(1) ActionServlet：Controller，主要負責處理用戶端傳送過來的 request，分析這個請求後產生適當的 Model 元件來處理資料，處理完後決定適當的 View (JSP) 元件。

(2) ActionMapping：代表所傳送過來請求中 URI 路徑與企業邏輯元件 (Action) 之間的對照。

(3) ActionForm：Model 元件，ActionServlet 會分析用戶端傳送過來的請求產生所需要的 ActionForm 物件。

(4) Action：這也是 Model 元件，用來表示企業邏輯部分，處理 Form Bean 的資料與流程。

(5) Taglib：這元件用來表示 Struts 中的標籤函式庫。

(6) JSP：View 元件，負責展現部分。Struts 建議 JSP page 只是很單純作頁面的呈現。

(7) struts-config.xml：Struts 設定檔，定義了 Form Bean 和 Action Mapping 的 Action 設定等等。

上述七大元件是 Struts 整個架構的核心，透過它們就能夠讓系統開發者將複雜的程式碼予以簡化，並且能夠讓不懂程式架構的美工設計人員，只要經過使用 Struts 標籤的教育訓練後便可以輕易上手的工具。

2.3 其他樣式

本篇研究也使用一些其他常用的樣式，包括：DAO、VO。

(1) DAO Pattern：將商業邏輯從資料存取邏輯分離出來，有關資料存取都會被封裝在 DAO 的類別中，因此當資料庫欄位改變時，並不需要去改變主程式或服務，只要更動 DAO 的方法，就可以很輕鬆的完成這個異動需求。

(2) VO Pattern：使用定義為 private 的私有變數及許多定義為 public 的 set / get 方法供呼叫使用，本篇研究會使用 DAO 及 VO Pattern 來達成簡單的永久性目的。

3. 研究架構

3.1 研究架構

本研究藉由現行實務與相關文獻探討，歸納整理後提出提出一套系統整合方案，以統一化塑模語言做為使用者與資訊人員溝通的工具、以 MVC 樣式為規範，以 DAO、VO 樣式做資料庫的溝通與永久性管理，來實作 GMVC 框架做為整個系統的骨幹與方法，以物件導向的語言 JAVA 來開發 RMA 雛型系統。研究架構如圖 3-1。

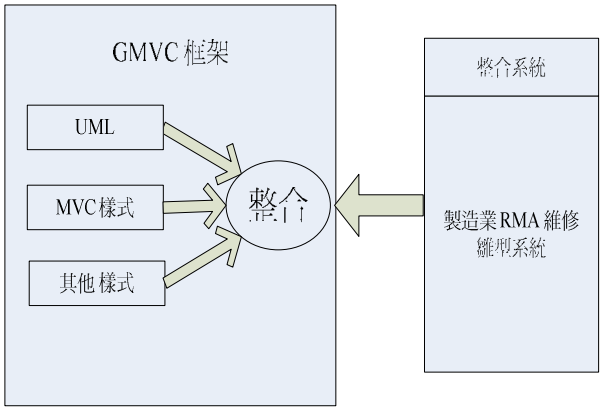

圖 3-1 本研究架構

3.2 研究方法

本研究的主要目的在於將 RMA 維修系統實際運用於 GMVC 框架之下，屬『系統展示』型研究，亦即藉由實作的『雛型系統』來達到系統展示的目的，包括系統設計、架構建立、產生雛型。

3.3 研究流程

綜合研究架構及研究方法的內容，本研究流程說明如下：

(1) 確認研究的動機與目的：本研究首先確認前述的研究目的。

(2) 相關文獻探討：對相關的文獻收集與整理，包括：UML、MVC 樣式等。
(3) 方案整合：接下來對所有文獻進行整合研究。
(4) GMVC 框架系統實作：實作整個 GMVC 框架系統。
(5) RMA 維修系統實作與測試：進行 RMA 雛型系統系統設計、開發與測試。
(6) 提出結論與未來建議：根據雛型系統實作與測試結果，提出對後續研究方向的建議。
(7) 研究流程如圖3-2。

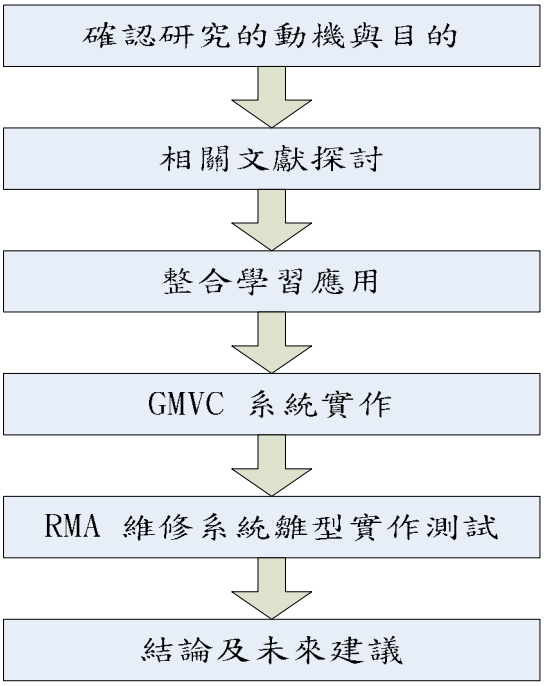

圖3-2 本研究實作流程

4. 系統實作驗證

4.1 系統描述

本研究主要的目的乃在於透過建立製造業 RMA 雛型化系統部份功能，來驗證 GMVC 框架的可行性，因此本研究在系統實作會包含兩個部份，GMVC框架和 RMA 雛型系統。

4.1.1 系統環境

本 GMVC框架及 RMA 雛型系統以 JAVA 程式語言及JBuiler 整合開發工具以 MySQL 5 為資料庫。

(1) JAVA SDK 1.6.0_03：JAVA 是 SUN 開發的一種跨平台的物件導向的語言。
(2) JBuilder 2006：Borland 公司的產品，提供一個多功能的 JAVA 開發軟體，是一個功能強大的開發工具。
(3) MySQL 5：目前佔有率相當高的一套 Shareware 的關連式資料庫。

4.1.2 系統功能

GMVC 框架：

圖4-1為 GMVC 框架運作流程，本架構透過 TCP/IP 來做即時資訊收送，雖然依循 MVC 樣式亦包含 Model、View、Controller 三大部份，但作用並不完全相同，說明如下：

(1) Controller：負責做 TCP/IP 的溝通，並解析 XML File，Controller 會自動依 XML 中的設定呼叫對應的 Model 元件，並不需要在程式中指定，最後完成邏輯處理後指派 View 回傳。
(2) Model：處理所有系統邏輯並負責資料庫的存取。
(3) View：在 MVC 的 View 為被指派的網頁，在 GMVC 框架下的 View 指的是被呼叫的程式名稱。

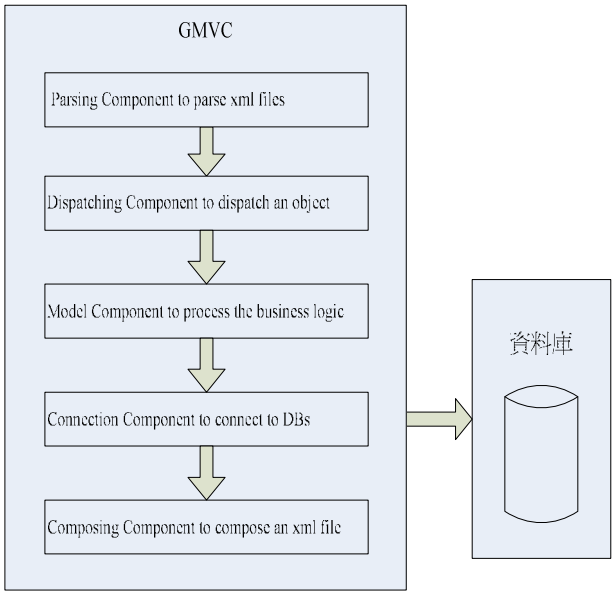

圖4-1 GMVC 框架運作流程

RMA 雛型系統：

RMA 系統為製造業中的維修系統，其流程為從客戶提出壞品維修申請，到完修後送回給客戶的所有流程，本雛型系統將包含 4 個 RMA 最主要的功能，說明如下：

(1) 申請 RMA：客戶透過系統發出 RMA 申請單。
(2) 確認作業：當收到 RMA 申請單時，主管需先審核放行。
(3) 收貨作業：放行後的 RMA 單會再主動通知客戶寄送壞品。
(4) 出貨作業：維護完成後再出貨給客戶。

4.2 系統設計

本篇研究使用 UML 的使用者案例圖及

循序圖等來描述系統各種狀況，最後使用佈署圖來做系統佈署的工作。

(1) 使用者案例圖 (Use Case diagram)

以使用者的觀點來描述系統的功能，在此圖形中包括：行為者(Actors)代表系統的使用者，Use Case 表示系統所提供給使用者的功能。

本例雛型系統所提供給使用者的功能有『申請 RMA』、『確認作業』、『收貨作業』、『出貨作業』以及各服務所延伸的子服務，如圖4-2 所示。

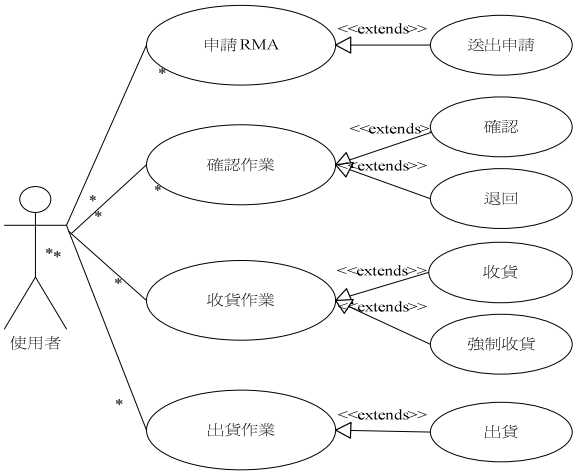

圖 4-2 雛型系統 – 使用者案例圖

(2) 循序圖 (Sequence Diagram)

循序圖用來描述物件之間的互動情形，它強調時間上的先後順序，循序圖通常是一個或數個使用者案例的詳細流程。本例雛型系統，申請 RMA 後送出資料，主管可退回或核准申請，核准後客戶開始寄送壞品，依客戶寄送次數反覆收貨，待維修完成可分批出貨送回給客戶，如圖 4-3 所示。

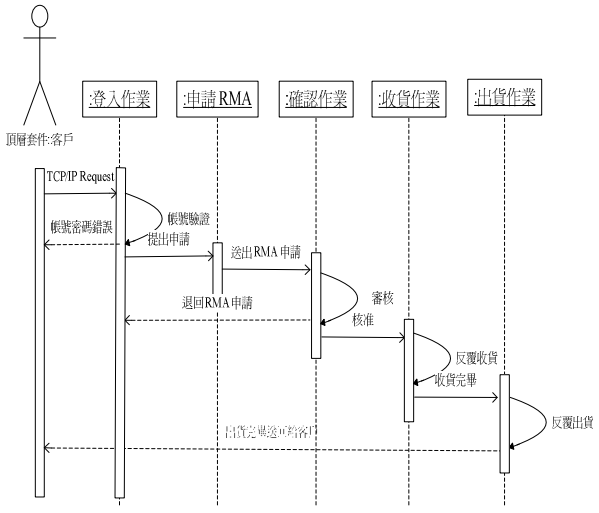

圖 4-3 雛型系統 – 循序圖

(3) 部署圖 (Deployment Diagram)

部署圖是描述軟體實作元件與硬體資源(客戶端、程式、伺服器等) 間的配置關係，如圖4-4 所示。

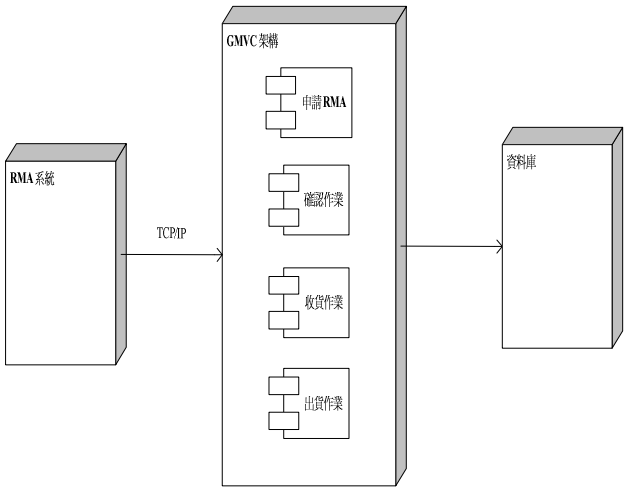

圖 4-4 雛型系統 – 部署圖

4.3 系統實作與測試

框架有其一套固定的程序與作法，來提供系統開發人員遵循的標準，透過這樣的標準才能使企業內部所有的基礎程式碼達成一致性，同時也能擁有相同的系統品質，更能增加系統的穩定度，對於後續的維護也更方便。

本節將說明 GMVC 框架及 RMA 雛型系統的開發過程相關定義及設定規格說明，並列出關鍵原始程式碼，程式撰寫完成後做系統測試與展示。

4.3.1 系統實作

(1) GMVC 框架

本架構主要是透過 XML 設定檔和 Properties 設定檔來初步完成整個 系統架構，再經由 RMA 系統傳送過來的 XML 檔案來完成設定 MODEL 和 VIEW 之間的關係，,GMVC 框架相關的設定,如下：

(a) 整個 GMVC 框架系統的目錄結構，如圖 4-5。

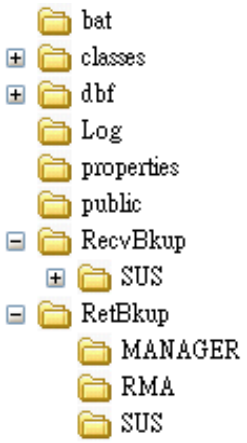

圖 4-5　目錄結構圖

bat：部份需執行的批次目錄
classes：所有編譯後程式所在目錄
Log：系統記錄存放目錄
properties：系統設定檔目錄
public：系統設定檔目錄
RecvBkup：Request 備份目錄
RetBkup：Response 備份目錄

(b) Server.xml 設定檔，本檔設定系統使用的基本資訊和備份路徑，另外亦設定排程啟動機制等等，如圖 4-6

```
<?xml version='1.0' encoding='big5' ?>
<ShopFloor_Utility_Server>
  <Main>
    <ServerPort>30000</ServerPort>
    <RecvBkup>D:/Project/ShopFloor_Utility_Server/RecvBkup</RecvBkup>
    <SUSRetBkup>D:/Project/ShopFloor_Utility_Server/RetBkup/SUS</SUSRetBkup>
    <RMARetBkup>D:/Project/ShopFloor_Utility_Server/RetBkup/RMA</RMARetBkup>
    <ManagerRetBkup>D:/Project/ShopFloor_Utility_Server/RetBkup/MANAGER</ManagerRetBkup>
  </Main>
  <Timer>
    <StartTimer>false</StartTimer>
    <StartTimerInterval>1440</StartTimerInterval>
  </Timer>
</ShopFloor_Utility_Server>
```

圖 4-6　Server.xml 設定檔

ServerPort：Socket Server Port
RecvBkup：設定系統收到需求的備份目錄路徑
RMARetBkup：系統回給 RMA 系統的備份目錄路徑

(c) System.properties 設定檔，本檔亦設定系統的基本資訊，最重要的是設定所有資料庫連結的設定，及郵件伺服器的設定等等，如圖 4-7

```
ShopFloor_Utility_Server_XML=D:/Project/ShopFloor_Utility_Server/public/ShopFloor_Utility_Server.xml
RecCount = 300
BoardCount = 200
BoardDB = MySQL.2
LogDir = D:/Project/ShopFloor_Utility_Server/Log

isPerdataBkup = false
PerdataBatFile = D:/Project/ShopFloor_Utility_Server/bat/CopyPerdata.bat

# MySQL
MySQL.NUM = 3
MySQL.INIT.1 = true
MySQL.jdbcDriver.1 = com.mysql.jdbc.Driver
MySQL.URL.1 = jdbc:mysql://221.222.222.21:3306/sus
MySQL.UID.1 = root
MySQL.PWD.1 = 1234
MySQL.minimumConnections.1 = 20
MySQL.maximumConnections.1 = 30


host = 221.222.222.6
UserName = flytech
Password = flytech
fromMail = frank@flytech.com.tw
toMail = eric@flytech.com.tw
```

圖 4-7　system.properties 設定檔

RecCount：回傳前置系統的資料筆數
MySQL.*：設定連結 MySQL DATABASE 的所有資訊

(d) NewRMA.java，此為 Model 部份，以申請 RMA 為例，所有 Model 程式都必需繼承自 ServiceType，解析 RMA 系統傳來的要申請 RMA 的需求 XML 檔，將解析後的資料傳給 DAO 元件使用，最後呼叫 COMB 元件重組 XML，將結果傳回給 RMA 系統，如圖 4-8

```
public class NewRMA extends ServiceType {
  public String start() {
    String _sRetStr = "";
    try {
      Vector _vRead = new Vector();
      String _sTmpNo = xmlParseComponent.getTagData(document, "TMPNO");
      String _sRmaNo = xmlParseComponent.getTagData(document, "RMANO");
      String _sCustomerId = xmlParseComponent.getTagData(document, "CUSTOMERID");
      String _sRequestId = xmlParseComponent.getTagData(document, "REQUESTID");
      String _sFailDesc = xmlParseComponent.getTagData(document, "FAILDESC");
      _vRead.add(0, _sTmpNo);
      _vRead.add(1, _sRmaNo);
      _vRead.add(2, _sCustomerId);
      _vRead.add(3, _sRequestId);
      _vRead.add(4, _sFailDesc);
      NodeList lineNodeList = document.getElementsByTagName("LineRec");
      int _iLineCnt = Integer.parseInt(xmlParseComponent.getTagData(document, "LineCnt"));
      Vector[] _vLine = new Vector[_iLineCnt];
      for (int j = 0; j < _iLineCnt; j++) {
        _vLine[j] = new Vector();
        String _sSn = xmlParseComponent.getTagData(lineNodeList.item(j), "SN");
        String _sModel = xmlParseComponent.getTagData(lineNodeList.item(j), "MODEL");
        String _sRequestQty = xmlParseComponent.getTagData(lineNodeList.item(j), "REQUESTQTY");
        _vLine[j].add(0, _sSn);
        _vLine[j].add(1, _sModel);
        _vLine[j].add(2, _sRequestQty);
      }

      NewRMADAO cmdDAO = new NewRMADAO(conn, conn2, ShopFloor_Utility_Server.log, _sLoginId);
      _sTmpNo = cmdDAO.addNewRMA(_vRead, _vLine);
      NewRMACOMB cmdCOMB = new NewRMACOMB();
      _sRetStr = cmdCOMB.combineGetTmpNo(sock, getClass().getName(), _sTmpNo);
```

圖 4-8　NewRMA.java

(e) NewRMADAO.java，DAO 元件負責對所有資料庫的存取，本例會將 NewRMA.java 傳過來的申請資料存入資料庫，如圖 4-9

```
public synchronized String addNewRMA(Vector vHead, Vector[] vLine) throws Exception
{
  ResultSet rs = null;
  try {
    String sql = "select max(tmpno) as tmpno from rma_header";
    PreparedStatement pps = conn.prepareStatement(sql);
    sql = "Insert into rma_header ("+
        "tmpno,rmano,customerid,requestid,faildesc)values(?,?,?,?,?)";
    pps = conn.prepareStatement(sql);
    pps.setString(1, (String) vHead.get(0));
    pps.setString(2, (String) vHead.get(1));
    pps.setString(3, (String) vHead.get(2));
    pps.setString(4, (String) vHead.get(3));
    pps.setString(5, (String) vHead.get(4));

    for (int i=0;i<vLine.length;i++){
      sql = "Insert into rma_line (" +
          "tmpno,sn,model,requestqty)values(?,?,?)";
      pps = conn.prepareStatement(sql);
      pps.setString(1, _sTmpNo);
      pps.setString(2, (String) vLine[i].get(0));
      pps.setString(3, (String) vLine[i].get(1));
      pps.setInt(4, Integer.parseInt((String) vLine[i].get(2)));
      pps.executeUpdate();
    }
    conn.commit();
```

圖 4-9　NewRMA.java

(f) NewRMACOMB.java，COMB 元件負責將 DAO 處理結果再組成 XML 檔案傳回給前置系統，本例會將 NewRMADAO.java 處理後產生的 RMA 單號，組合成 XML 檔案傳回給 RMA 系統，如圖 4-10

```
public String combineGetTmpNo(Socket sock, String sServiceType, String tmpno) throws Exception {
  try {
    StringBuffer retIniStr = new StringBuffer(
       "<?xml version='1.0' encoding='UTF-8' ?><RsXMLData><Header><ServiceType>" +
       sServiceType + "</ServiceType><ErrCode>" + _sOKCode +
       "</ErrCode><ErrDesc>" + _sOKCodeDesc +
       "</ErrDesc><ClassName/><Exception/></Header><Text><RecCnt>1</RecCnt><Group>");
       retIniStr.append("<RsRec>");
       retIniStr.append("<TMPNO>" + tmpno + "</TMPNO>");
       retIniStr.append("</RsRec>");
       retIniStr.append("</Group></Text></RsXMLData>");

    String _sFile = GetDateTime.getTodayDate("") + "_" +
       GetDateTime.getTime("") + "_" + sock.getInetAddress().getHostAddress() +
       "_" + sServiceType + ".xml";
    PrintStream outt = new PrintStream(new FileOutputStream(sRetBackup + "/" + _sFile), false, "UTF-8");
    outt.println(retIniStr.toString());
    outt.flush();
    outt.close();
    return retIniStr.toString();
```

圖 4-10　NewRMACOMB.java

(2) RMA 雛型系統

本研究主要是透過此 RMA 系統來驗證整個 GMVC 框架，經由 XML 設定檔來完成連結 GMVC 框架的 MODEL 和 VIEW 之間的關係，相關設定如下：

(a) 申請 RMA，當客戶輸入相關資料後，RMA 系統會將畫面中的欄位資料組合成 XML 檔案，檔案中將載明 GMVC 框架的相關資訊，組合完成後送往 GMVC 框架，如圖 4-11。

圖 4-11　新申請 RMA 畫面

(b) NewRMA.xml，本檔案由上述申請 RMA 的欄位所組合而成，如圖 4-12，除了畫面上的欄位，檔案中還有部份不在畫面欄位上，說明如下：

System：GMVC 框架為可多系統共用之架構，因此須告訴 GMVC 框架是由哪一個系統所發出來的需求。

ServiceType：GMVC 框架會根據這個標籤的設定呼叫對應的 Model 元件。

DBKind：告訴 GMVC 使用的是哪一個資料庫。

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RqXMLData>
    <ToIp />
    <ToPort />
    <System>RMA</System>
    <ServiceType>NewRMA</ServiceType>
    <DBKind>MySQL.3</DBKind>
    <LOGINID>ericyeh</LOGINID>
    <LOGINPASSWORD>71</LOGINPASSWORD>
    <TMPNO />
    <RMANO />
    <CUSTOMERID>andy</CUSTOMERID>
    <CUSTOMERNAME />
    <CONTACT>andy</CONTACT>
    <CONTACTEMAIL />
    <REQUESTID>ericyeh</REQUESTID>
    <SALESID>sarah</SALESID>
    <SALESNAME />
    <REPAIRCENTER>1</REPAIRCENTER>
    <RMATYPE>1</RMATYPE>
    <STATUSCODE />
    <TOTREQUESTQTY>1</TOTREQUESTQTY>
    <FAILDESC>tUyqa7Z9vvc=</FAILDESC>
    <LineCnt>1</LineCnt>
  - <RqRec>
    - <LineRec>
        <SN>123-456-789</SN>
        <MODEL>M123456789</MODEL>
        <REQUESTQTY>1</REQUESTQTY>
        <UNITPRICE>1250</UNITPRICE>
      </LineRec>
    </RqRec>
  </RqXMLData>
```

圖 4-12　NewRMA.xml

上述是組成 GMVC 框架重要元素，本篇研究成功地透過 RMA 雛型系統，以 TCP/IP 來傳送資料，達到 GUI 系統 MVC 的目的，從 RMA 系統上我們可以發現純粹只是在處理畫面的運作，商業邏輯及資料庫存取都在 MODEL 上面，並依此規範整個系統架構，使整個系統能符合 MVC 的特性，商業邏輯從 UI 上分離達到高度去耦的目的，一致的標準讓開發者依循，系統可保有高度的一致性，如此當然就非常容易做後續的維護。

4.3.2 系統成果展示測試

本研究實作的 GMVC 框架已成功開發完成，所有程式在編譯後 (不含外部引用的函式庫) 相當輕巧僅僅 1.36M，因此耗用系統資源相當少，在 RMA 雛型系統方面也已實作完成，包括在系統設計階段規劃的四大功能，登入作業、申請 RMA、收貨作業、出貨作業等，另外還有提供使用者查詢申請資料等功能。

本 RMA 雛型系統特以網頁式 GUI 系統方式來開發，方便使用者以瀏覽器進入系統，使用者並不需要先行安裝系統，有別於以往 GUI 程式需先安裝的特性，惟本系統以 JAVA 語言開發，若使用者尚未安裝 JAVA 執行環境，系統在第一次執行時將會主動提示下載安裝 。

目前整個架構包括 GMVC 框架及 RMA 雛型系統，已在國內某家上市公司工業電腦大廠內部測試完成，執行的過程相當順暢，在效能的展現上也相當不錯，該公司已計劃繼續加強整個 GMVC 架構，並支持繼續實作此 RMA 系統，希望成為公司內部的正式系統。

5. 結論與未來建議

5.1 結論

流程控制是程式設計師必須處理的重要工作之一，但這些工作的處理相當困難，因此，如何以一個良好的系統架構，規範這些控制工作的邏輯設計，決定了系統的軟體品質。

本研究使用 GMVC 框架來架構 RMA 雛型系統，雖然在功能上並非相當完善，但經過整個分析、設計與實作過程，可以從每個環節裏看出，以 UML 來做系統分析與設計，將 RMA 雛型系統建構在 GMVC 框架之下仍然能達到 MVC 架構所強調的商業邏輯與畫面的低耦合，讓程式具有一致性來確保系統品質，如此就更易於維護。

5.2 未來建議

本研究雖然成功透過實作 RMA 系統來驗證 GMVC 框架，但仍可看出有些地方尚不完整，因此本研究提出下列幾項建議：

(1) GMVC 框架雖然可整合各種不同的應用系統，但在處理 GUI 系統時卻多一層考量，因其多使用在 Intranet，若於 Internet 上執行，其效能是否能跟實證的結果有一樣的品質。

(2) 在本研究的範圍裏，並沒有特別考慮網路的安全性，部份重要資訊存在 XML 檔案透過網路傳送容易被截取，應該可以針對這方面提出有效的防範措施。

參考文獻

[1] 林俊孝，「一個支援作業流程控制的Web應用程式設計框架」，碩士論文，國立台北科技大學資訊工程系，台北，2003

[2] 邱炫儒，「以文件式 Model-View-Controller 設計樣式為基礎的應用系統開發方法」，碩士論文，私立中原大學資訊管理學系，中壢，2002

[3] 黃石欽，「Model-View-Controller (MVC) 架構之整合設計與實務應用」，碩士論文，私立中原大學資訊管理學系，中壢，2003

[4] 陳泓志，「物件導向企業框架之研究—以會計總帳系統為例」，碩士論文，國立政治大學資訊管理系，臺北，2000

[5] 陳冠儒，「網路應用程式架構 Struts 之分析工具」，碩士論文，私立中原大學資訊工程學系，中壢，2004

[6] Sinan Si Albir，陳志昌 (編譯) ，「UML 技術手冊」，美商歐萊禮公司臺灣分公司，臺北，2000

[7] Souza, Vítor Estêvão Silva and Ricardo de Almeida Falbo; An Agile Approach for Web Systems Engineering; *Proceedings of the 11th Brazilian Symposium on Multimedia and the web*; New York; ACM; 1-3; 2005e

[8] Luo, GuangChun, Yanhua Wang, Xianliang Lu and Hong Han; A Novel Web Application Frame Developed by MVC; *ACM SIGSOFT Software Engineering Notes*; 28, 2: 7; 2003

[9] Hansen, Stuart and Timothy V. Fossum; Refactoring Model-View-Controller; *Journal of Computing Sciences in Colleges*; 21, 1, 120-129; 2005

[10] Murray, Leesa, David Carrington and Paul Strooper; An approach to specifying software frameworks; *Proceedings of the 27th Australasian conference on Computer*

science - Volume 26; Dunedin, New Zealand; ACM; 185-192; 2004e

[11] Fayad, Mohamed and Schmid Douglas, C.; Object-Oriented Application Frameworks; *Communications of the ACM*; 40, 10: 32-38; 1997

[12] Sun 教育訓練手冊，「SL-314 Web Component Development With Servlet and JSP Technologies Student Guild」，Sun Microsystems, Inc.，Taipei，2003