# 國 立 交 通 大 學

## 資訊學院 資訊學程

碩 士 論 文

# 豐富互動式網際網路應用於現場即時資訊系統

# Shop Floor Information System Using Rich Internet Applications

研 究 生：邱兆民

指導教授：蔡文能 教授

林正中 教授

中 華 民 國 一 百 零一年 十 月

# 豐富互動式網際網路應用於現場即時資訊系統

# Shop Floor Information System Using Rich Internet Applications


研 究 生：邱兆民　　Student：Chao-Min Chiu

指導教授：蔡文能　　Advisor：Wen-Nung Tsai

林正中　　Cheng-Chung Lin


國 立 交 通 大 學

資訊學院 資訊學程

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

October 2012

Hsinchu, Taiwan, Republic of China

# 豐富互動式網際網路應用於現場即時資訊系統

學生:邱兆民　　指導教授:蔡文能
林正中

國立交通大學 資訊學院 資訊學程 碩士班

## 摘要

製造執行系統(Manufacturing Execution System; MES)是工廠用來對於重要生產進行監控的資訊系統,亦是電腦整合製造(Computer-Integrated Manufacturing; CIM) 系統核心。其主要用途是對生產流程的品質及製程資訊的追蹤，對產品品質及產量進行衡量。而現場即時資訊系統(Shop Floor Information System, SFIS)是任何成功的 MES 製造系統所不可欠缺的重要環節。生產單位可以依靠此系統，掌握生產即時訊息，用以輔助作出決策、協調工廠內所有的生產活動、監督生產狀況、控制所有生產資訊流，還可提供需求單位透明且正確的生產資訊。

本研究中將 MES 系統以三層式的架構建置，透過服務導向架構(SOA)建立中介層，各系統可使用其為互相溝通的橋樑。而前端的使用者介面則使用 Flash 來實作，其為豐富網際網路應用(RIA)技術的一種，透過 Flash 中豐富的多媒體元件，可充分展現出現場即時資訊系統所要展現的各項資訊，並可和使用者達成良好互動；以元件化的開發方式，可使得系統在開發維護上更加簡易，大幅增加開發維護的效能。

關鍵字: 製造執行系統、豐富網際網路應用、服務導向架構、現場即時資訊系統、Flash

Shop Floor Information System Using Rich Internet Applications

Student: Chao-Min Chiu

Advisor: Dr. Wen-Nung Tsai
Dr. Cheng-Chung Lin

Degree Program of Computer Science
National Chiao Tung University

Abstract

Manufacturing Execution System (MES) is an information system that many factories use to monitor the important manufacturing process. It also becomes a core system of Computer-Integrated Manufacturing (CIM). The main function of the system is to control the quality of production process, as well as tracking the production information and evaluating the production quality and quantity. The Shop Floor Information System (SFIS) is even an indispensable piece in any of the successful MES manufacturing system. Production unit can rely on SFIS to collect information in time. The collected information can then be used to assist in decision making, to coordinate all of the manufacturing activities, to monitor production situations, to control the entire production data flow, and to provide transparent and correct production information to the responsible departments.

In this research, we use 3 structured layers to build the MES system. A mediator layer is built through a Service-Oriented Architecture (SOA), and each system may use the mediator layer for communication. The front user interface is developed using Flash. It is one of the Rich Internet Application (RIA) technologies. By using the RIA of Flash, we can fully demonstrate the different information of shop floor information system, and achieve successful interactions with the users. We design and develop our SFIS system in component base. This makes the implementation task easier and also makes the system easier to maintain in the future.

Keywords: MES, Rich Internet Application, Service-Oriented Architecture, Shop Floor Information System, Flash.

# 誌謝

從大學畢業到進入職場已有一段時間，在這段時間中，待在目前公司的團隊中，所幸有部門內各位學長的不吝指教，令我學習到許多專業的知識及技巧，並在當我犯錯的同時，陪我找出問題並協助回覆系統的狀態，感謝部門學長們的耐心包容及指導，尤其感謝黃昱彰以及薛富中兩位學長，指導了我許多程式的撰寫技巧。

於工作兩年後，和同事談話的過程中，鼓勵我報考交大的研究所，在這段求學階段，和許多不同領域的同學互相學習，認識了李長霖、劉宣佑、賴嬿竹三位學長姐，三位學長姐不時的互相討論功課並提醒我該注意的事項，由於有你們的協助，令我更順利的完成了課業的修習。

在論文撰寫的過程當中，除了公司學長的協助外，同時感謝蔡文能教授不斷的對於撰寫的內容提出意見以及修正，透過每次的論文指導，教授耐心的聽完所有的內容後並提出不合理的地方，從毫無頭緒的過程中，漸漸有了整個論文雛形並到最後的完成，而在指導的過程中，不時的穿插幽默的言談，使得氛圍輕鬆不少，由衷的感謝蔡教授。

邱兆民

謹誌於交通大學資訊學院在職專班資訊組

2012 年 10 月

# 目錄

# 圖目錄

# 表目錄

# 第一章 緒論

隨著科技的日新月異，許多的公司，在面臨到人工成本的提高；原物料成本的上揚。此時，若無法將成本降低，提高產品品質，縮短交貨期限，便無法保持市場的競爭力。有鑒於此，公司需要隨時掌握工廠的最新動態，並配合外來的資訊，才能使管理者定下最佳的決策。由於電腦技術的不斷進步，其逐漸被運用於製造業中，包括了產品開發、設計、製造、庫存管理、辦公室管理等的自動化系統，來配合市場的需求，解決許多存在的問題。然而，於應用上往往每一個系統皆是各自獨立的，彼此之間並無法有效的連接在一起，造成許多相關的訊息無法互相傳遞或資料重複的儲存，而為了解決此問題，電腦整合製造(CIM)便產生了。

Manufacturing Execution System(MES)是生產工廠用來對於重要生產進行監控的資訊系統，亦是CIM的系統核心。其主要用途是對生產流程的品質及製程資訊的追蹤，對產品品質及產量進行衡量，其他的輔助功能則包括生產設備管理、人員管理、庫存管理、品質管理、成本管理、生產關鍵性績效指標(Key Performance Indicator, KPI)衡量，報表製作，機台連線等。MES的系統規模可大可小，可以是單純的在製品(Work In Process, WIP)追蹤，也可是工廠製造監控的整合解決方案；不過不管規模如何,MES的共通特點是處理的對象都是來自於生產製程有關的資訊。

現場即時資訊系統(Shop Floor Information System, SFIS)是任何成功的MES製造系統所不可欠缺的重要環節。生產單位可以依靠此系統，掌握生產即時訊息，用以輔助作出正確的決策、協調工廠內所有的生產活動、監督生產狀況、控制所有生產資訊流，還可提供需求單位透明且正確的生產資訊。

## 1.1 研究動機

現場即時資訊系統對於製造業而言，是一項重要的資訊掌控系統，其可以有效的協助工廠生產。舉凡從客戶訂單的成立後，便會開立工單，製造單位便需協助建立相關的生產流程資訊，生產過程中，現場同仁則需依靠此系統了解各在製品目前的生產狀況，如此可避免針對同一在製品有重複製程或缺漏製程的發生，造成需要重工或報廢的浪費情況出現，無形中增加了成本。對管理者而言，可快速瞭解目前工廠所面臨的生產問題，作出決策。然而針對環境的快速變遷，以及目前公司內開發者對於系統的掌控上，遇到了如下的問題，迫切的需要將舊有的系統改造翻新，

於是便有了本研究的產生。

◆ MES 系統建置成本過高

對於大環境產品的快速改變，公司需要不斷的投資於新產品的製造，於是便有不斷建廠的情況發生，然而目前市面上所販售的 MES 系統，動則上億元的購買成本，在建廠初期所要投入的成本就大幅度的增加，如何節省下此部分的成本，就成為了很重要的議題。

◆ 無法動態調整系統資源

工廠的環境為 24 小時生產，因此相對的現場即時資訊系統，亦是 24 小時皆有使用者不斷的針對各項的資訊進行作動，除了年度維修外，系統是不允許停止服務，但因人為的操作不當，故而有機會發生系統無預期的程式錯誤，導致整個系統就此停擺，因此系統若是可以動態的切換到備源機制，可大幅減少生產停止的狀態。

◆ 無法掌握所有 MES 系統核心

現行廠內 MES 系統中，部分的程式原碼是廠商在當初購買時所沒有給予的，因此在維護系統或製造單位提出需求時，無法有效的針對需求作出修正，必須另闢途徑來達到同等效果。

◆ 使用者介面不友善

當查詢產品生產流程資訊時，目前場內 MES 系統，無法充分對於圖形化的訊息作出顯示，現場同仁使用時，須輔以另一系統，才可得到完整的資訊，因而造成使用上的不便利，需頻繁切換系統，造成使用上的不便。

## 1.2 研究目的

目前，工廠生產時，絕大多數的時間點上，使用者需要透過現場即時資訊系統來了解目前廠內的生產狀況，為迅速了解狀況，勢必就需要完整的資訊來輔助使用者掌控所有的訊息，因此，如何將系統改善為簡單又方便使用者掌控各種的資訊，就成為一項很重要的課題。再者，面臨到不斷的建置新廠的情況發生，如何降低建廠時所需的成本，對於企業而言，也是一項很重要的指標，在此多重的問題考量下，自行研發一套可由廠內資訊人員完全掌控開發的系統，就變得勢在必行。

為了因應上述的問題，在本研究中，期望可以達到以下目標：

- 快速的程式開發方式

  為了快速的開發程式，在初期建置系統時，即考量以元件化的方式進行開發，針對每項功能，只需符合基本的程式撰寫方式，再配合呼叫系統所提供的函式，即便後續開發者並非當初的程式開發人員，也可輕鬆完成需求。

- 友善的使用者介面

  因應目前系統無法有效的以圖形化的方式呈現資料，而部分的資訊，並非簡單用表格或文字即可方便描述解讀，為此希望新系統可達成針對圖形化介面的資料有簡單明瞭的介面呈現。

- 方便的系統資源調整

  因廠內為二十四小時不斷生產的環境，使用者於生產的過程中，即有查詢或使用此系統的需求，然而系統難免會遇到進行維修或 Core dump 的情況發生，因此為了能夠使系統正常運作，有效的切換系統資源便不可或缺。

## 1.3 論文架構

第二章將背景知識中，將會介紹豐富網際網路應用程式(Rich Internet Application, RIA)的發展以及定義，其中包含了本研究所使用的開發語言 Flash；另外將會對於本研究中有使用到之各項技術做出簡單的介紹，其中包含：.Net Framework、Web Service、Windows Communication Foundation…等內容。

第三章文獻探討的部份，分別會介紹到 MES 的定義與功能、以元件化為基礎所架構的 MES 系統、以服務導向架構建立一平台，透過此平台整合 CIM(Computer integrated manufacturing)中的各項系統、最後介紹一篇以網頁為 MES 呈現方式的專利。

第四章中將會針對本研究架構進行解說，其中包含：MES Host Server、Application Server、Client UI、主動訊息廣播程式、主動訊息接收及周邊硬體控制程式。

第五章將首先會介紹各系統的開發運行環境，簡易的系統過帳流程，系統實作之成果，以及最後的系統評估。

第六章為本研究之結論與未來研究方向，其中將包含了本研究中，系統設計開發所作的貢獻以及對於未來研究方向的論述。

# 第二章 背景知識

為了完成本研究的系統，過程中使用到了多項的技術互相的配合開發，最後成功建置了系統。在現場即時過帳系統中對使用者而言非常重要的一環，就是使用者介面的呈現以及互動。本研究中，選擇了 Flash 為開發使用者介面的工具，其屬於豐富網際網路應用程式(RIA)的其中一種。

隨後介紹的為.Net Framework，其是本研究中建構中介程式 Application Server 所使用到的開發平台，而其他於本研究中將會使用到的各項技術，也會陸續的做出闡述，其中包含了 Web service、XML、Windows Communication Foundation 等。

## 2.1 豐富網際網路應用程序(RIA)

### 2.1.1 RIA 發展與定義

豐富網際網路應用程式(Rich Internet Application, RIA)，源自於 2002 年 3 月 Macromedia 公司(已於 2005 為 Adobe 併購)的白皮書。RIA 一開始只是 Macromedia 推動的網頁技術指南，其理念為一種具有近似於傳統桌面應用軟體系統、使用者介面豐富、多媒體以及資料庫的結合，並以此來開發出新一代的網站體驗。亦即是說，網頁(應用程式)不單能夠簡易的使用者互動，還必須符合人類的直覺與經驗，並結合網際網路應用程式易於開發與低成本的概念。RIA 企圖以可向量化的動態圖像的前端工具，結合後台各式各樣應用程式來打造跨平台的網路應用程式。

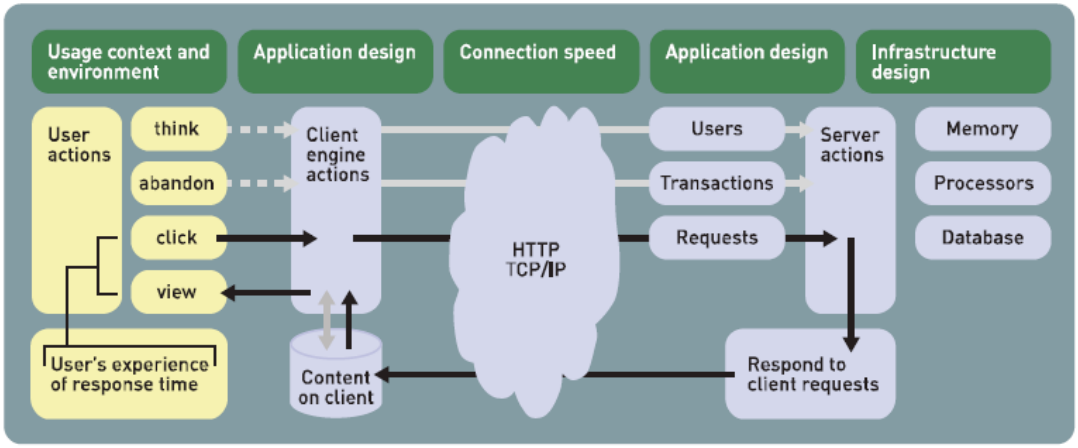


圖 1、RIA 行為模組 [1]

傳統的網路應用程序將所有交互應用都集中客戶端的Browser/Server架構上(圖 1 中的黑實線)。在這樣的系統中，所有處理操作均在伺服器端執行，客戶端僅僅是用於顯示靜態的信息內容（如HTML)。這種系統最大的缺陷是所有的交互操作都必須經由伺服器端進行，首先客戶端要將請求數據上傳至伺服器端，然後伺服器端作出響應並傳回結果，最後客戶端在重新載入顯示信息。RIA的架構則是在客戶端的使用者操作各項程序時，使用者可能不會馬上就執行某些動作來要求資訊，可能會有思考以及終止放棄的行為，而RIA應用程式則會在這些時間點上，預先和後端Server要求可能需要使用到的資訊(圖 1 中的虛線)，從而可以加快處理速度。[1]

隨著時間的遷移，網際網路標準正在逐漸地改變著，因此很難為豐富網際網路應用程式劃定出一個明確的概念範圍。但是，所有的豐富網際網路應用程序都有一個相同的特徵：它們在客戶端與伺服器端之間引入了被叫做「客戶端引擎」(Client Engine)的中間層。這種客戶端引擎通常作為應用初始化的一部分被下載，也可能隨著應用程式的運行在後續程式碼中作為補丁(Patch)被下載並補充進來。客戶端引擎充當瀏覽器的一個擴展，而且通常還會接管呈現用戶界面和與伺服器端進行通信的職責。透過客戶端引擎和伺服器的通訊，客戶端引擎預先便會將所需的資訊載入，有別於傳統的輪詢式架構，透過此種方式可減少客戶端和伺服器間的溝通，減少網路資源的浪費，另一方面，通過使用在客戶端執行指令的技術，RIA 可以有效的避免延遲，實現程序與用戶操作的同步。

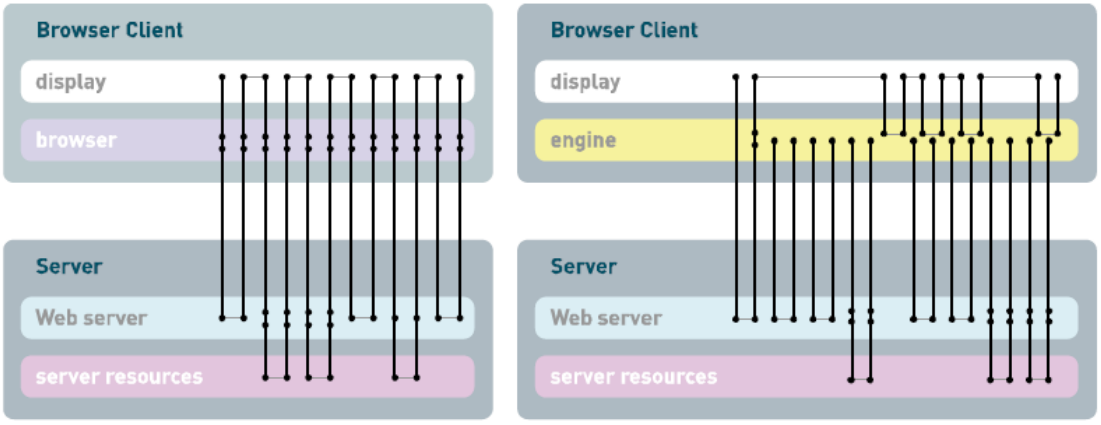


圖 2、(左)傳統網頁程式通訊模型、(右)RIA程式通訊模型 [1]

## 2.1.2 RIA 優點

RIA 擁有許多的優點使得開發人員較願意使用其來開發運行在瀏覽器中的應用程序，其安裝簡便的特性，使其在更新上相對於桌面程式或作業系

統來得容易，使用者只需連上網路，即可快速的下載更新到最新的版本，且可以在多數的瀏覽器上執行、而許多的行動裝置，也皆有支援。

RIA 應用程式在使用上，其可以在客戶端運算執行許多的資訊，因此使得伺服端不再像傳統的網頁應用程式一直處於需要運算許多要呈現在客戶端的資料，如此一來可大幅度的減輕伺服端負載，達成 Client/Server 間負載平衡。

透過 Client Engine，使用者不需要操作按鈕或連結，客戶端引擎便會將未來可能需要的數據先行下載，透過自行和伺服端溝通交握，而使用者可以趁此階段瀏覽其他頁面或操作其他行為，藉此來提升後續請求的回應速度。在和伺服端交換何數據時，RIA 也因傳輸的數據量變少，相對而言可以減輕網路的負載。

## 2.1.3 Flash 簡介

Adobe Flash (前稱為 Macromedia Flash)，簡稱 Flash，前身為 Future Splash，既指 Adobe Flash Professional 多媒體創作程序，同時也指 Adobe Flash Player。自從 Macromedia 公司於 2005 年 12 月 3 日被 Adobe 公司收購，Flash 也就成為了 Adobe 旗下的軟體。

Flash 運用向量圖形(Vector Graphics)技術開發，其檔案格式相對較小，所以常被大量運用在網際網路頁面的開發上。Flash 從早期的開發便一直不斷吸納各項語言的優勢，例如導入物件導向的概念，以及後續逐步整合 FLV 視訊，提供 Web service 和 XML 的預建資料連接等，搭配 Action Script，設計人員可使用它來使得程式中的元件和使用者間互動，而大量的聲音、圖像、動畫可使程式開發人員快速的建立起應用程式，而 Flash 亦支援大多數的網頁瀏覽器，因此使得 Flash 的應用越來越廣泛和普及。圖 2 為 Flash 開發過程中，各元件的關係流程圖。

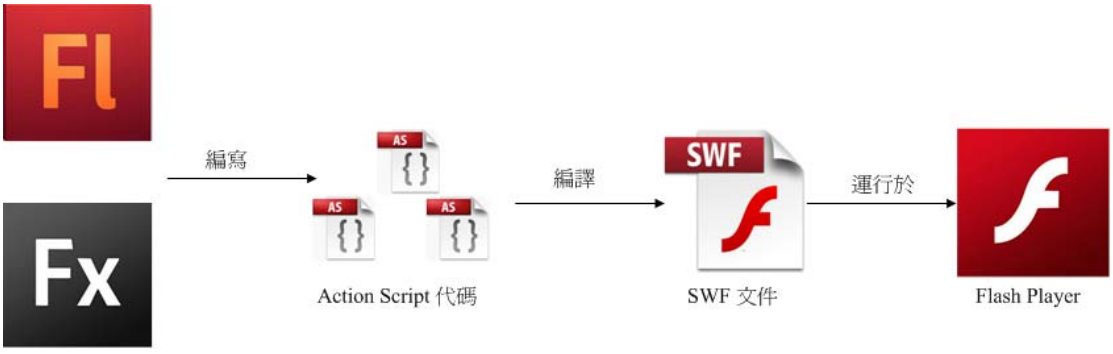


圖 3、Flash 環境開發、運行流程[9]

## 2.2 .Net Framework

.Net Framework 是由 Microsoft 繼 Windows DNA 之後的新開發平台，提出的分散式系統架構，以 Common Language Runtime(CLR)為基礎，支援多種語言開發(C#、VB .NET、C++、Python 等) 的編碼平台。

.Net Framework 中主要有兩個重要的部份：Common Language Runtime (CLR)以及.NET Framework 類別庫。當一般程式在執行的過程中，Common Language Runtime 像是一個管理這些程式的代理，其提供了類似像記憶體管理、執行緒管理和遠端處理等核心服務，同時執行嚴格的型別安全 (Type Safety) 以及加強安全性和強固性的其他形式的程式碼正確率。微軟為了要讓不同的程式碼間可以互相的共享，提出了 CLS (Common Language Specification) 規範，依據.Net 架構的精神，只要所使用的程式語言可以符合 CLS 規範，彼此就可以互相叫用，從而做到跨語言的目的。


圖 4、.Net Framework 架構圖[14]

而為了達到 CLS 規範，會先將程式轉換成另外一種語言：MSIL (Microsoft InterMediate Language)，其為一種中介的語言，但其還是需要一個執行的環境，而這個執行的虛擬環境就稱為 Common Language Runtime。CLR 同時提供了執行中的程式一些管理機制，例如垃圾回收(Garbage Collection)、一般型別系統(Common Type System)、例外處理(Exception Handing)、執行緒(Thread)支援…等。這些都是為了提供 MSIL 一個穩定的執行環境，讓程式可以順利執行。

而簡單的來說，凡是執行在 CLR 的程式，皆可稱為 Managed code，而不是運行在這個環境上的程式，則是 Unmanaged code。而另外一種的說法，則是程

式運行在.Net Framework 中來判定，但其實這兩者所指都是同一件事情。目前 Run Time 環境下，其也支援 Managed code 和 Unmanaged code 程式碼間的互通性，讓開發人員可以繼續使用必要的 COM 和 DLL。

.Net Framework 中另外一個重要的元件為類別庫，它是 Microsoft .NET Framework SDK 中所包含的類別、介面和實值型別 (Value Type) 程式庫。此程式庫提供系統功能的存取，並設計來做為建置 .NET Framework 應用程式、元件和控制項的基礎。其功能可加快程式開發過程和程式的最佳化，同時並提供對系統功能的存取。而為了達到語言間互通性，.Net Framework 類別庫型別也都符合 CLS 規範的標準。

.Net Framework 提供當豐富的介面，以及抽象和具體類別，其包含的面向有字串管理、資料收集、資料庫連接、檔案存取、主控台應用程式、Windows Form 應用程式、ASP .Net 應用程式、XML 網路服務、Windows 服務…等。而使用者也可以為了開發上的便利，自行衍生出自己的類別，透過使用這些類別庫，大幅降低了開發上的難度以及開發時間。

## 2.3 可延伸標示語言(XML)

### 2.3.1 XML 簡介

可延伸標示語言(Extensible Markup Language, XML)是一種標籤語言，它是由全球資訊網標準製定組織(World Wide Web Consortium, W3C) 所制定的，於 1998 年 2 月正式推出。XML 繼承了標準通用標記語言(Standard Generalized Markup Language, SGML)的優點，只取用 SGML 系統中的文件結構的核心部份，XML 繼承了 GSML 的擴展性，簡化了 SGML 一些複雜的語法設定，因為 SGML 是被發展用來解決編輯及保存內容龐大複雜而且互相連結的技術文件，所以 SGML 系統必須提供各種不同的語法，相對的它就顯的非常複雜。

近年流行的HTML也是屬於標籤語言的一種，雖然HTML具備了良好的展現能力，不過在格式完整性和自我表述能力方面則顯得十分薄弱，前者造成解譯器(Parser)的撰寫不易，後者則使得文件內文的語意難以理解。XML設計的目標為使電子文件能具有結構性並廣泛的應用於全球資訊網(World Wide Web, WWW)上，因此越來越多的應用程式使用XML來設計資料內容[8]。XML文件是屬於一種樹狀階層的結構文件，如下圖所示

```
<log4net>
  <!-- 輸出到檔案 -->
  <appender name="A2" type="log4net.Appender.RollingFileAppender">
    <file value="C:/log/logfile.log" /> <!-- 輸出檔名 -->
    <appendToFile value="true" />
    <maximumFileSize value="204800KB" /> <!-- 每個檔案最大size -->
    <maxSizeRollBackups value="5" />
    <rollingStyle value="Date" />
    <datePattern value="yyyyMMdd-HHmm" />
    <layout type="log4net.Layout.PatternLayout">
      <conversionPattern value="%level %logger (%file:%line) - %message%newline" />
    </layout>
  </appender>

  <root>
    <!-- 輸出訊息等級 -->
    <level value="INFO" />
    <appender-ref ref="A2" />
  </root>
</log4net>
```

圖 5、XML 文件內容

## 2.3.2 XML 的特點及應用

XML文件有以下幾個重要的特徵[7][8]：

1. 格式良好的(Well-formed)：XML 文件中的標籤，一定要有結尾標籤(end-tag)來首尾呼應、互相對稱，如上圖中<root>標籤後會伴隨著一個</root>標籤。
2. 需要被驗證的(Validity)特性：因 XML 文件皆為使用者自行定義，為確保其正確性，XML 文件須參考某驗證文件如文件類型定義(Document Type Definition ,DTD)或是 XML Schema 的驗證後，此時的 XML 文件才是合法的。

除了上述兩點主要特徵外，XML 文件還具有以下的特性：

- 簡單：XML 附帶有標籤及屬性，因此具有高度的文件結構化，XML 內部的標籤成對出現，使得整份文件能夠輕易的編寫及閱讀。
- 統一性(Unified)：透過 XML 做為交流媒介，只要有 XML Parser 就能解讀資訊，異質系統也可以互通。
- 高可攜性(Portable)：資料以純文字儲存，可跨平台使用，也容易讓不同的應用程式交換資料。
- 擴充性(Extensible)：使用者可自行定義標籤，或使用他人定義的標籤，在 XML 文件中，可定義出無限量的標籤，並提供標示結構化資料的架構。

- 自我表述(Self-described)：透過定義出含語意的標籤，可以輕易的解讀出文件的意義。

## 2.4 Web Service 與 SOAP

### 2.4.1 Web Service

Web Service 是一種建構於服務導向的應用程式架構，其可以讓不同平台的電腦系統間相互溝通、傳遞訊息，根據 W3C 的定義，Web service 是透過統一識別資源(Uniform Resource Identifier, URI)存取的軟體程式，經由 XML 定義、描述或搜尋其介面與連結方式，然後藉由網路為基礎的通訊協定(HTTP、SOAP、SMTP)，以 XML 訊息與其他軟體程式溝通。

從技術的觀點來看，Web service 是一種以網路為基礎的物件封裝與元件模組化技術，運用 XML 相關的技術標準，可以大幅降低企業或組織之間內、外部系統資訊整合的困難度，最終帶來工作效率的提升。

Web Service的架構如圖 6，主要區分為以下三個部份[2]：

1. 服務提供者 (Service Provider)：當開發人員撰寫好服務的程式之後，必須將它發佈成為一個公開而且可供呼叫的網絡服務，因此，服務的提供者會到註冊中心登錄網絡服務的必要資訊，當服務使用者查詢時，就可以查到網絡服務。
2. 服務使用者 (Service Client)：當服務使用者需要某項服務時，透過網路到註冊中心查詢所需要的服務。
3. 服務註冊中心 (Service Registry)：它提供一個公開且標準的機制，讓所有提供服務的 Web Service 來登錄相關資訊，同時也讓服務的使用者來查詢所想要的服務。

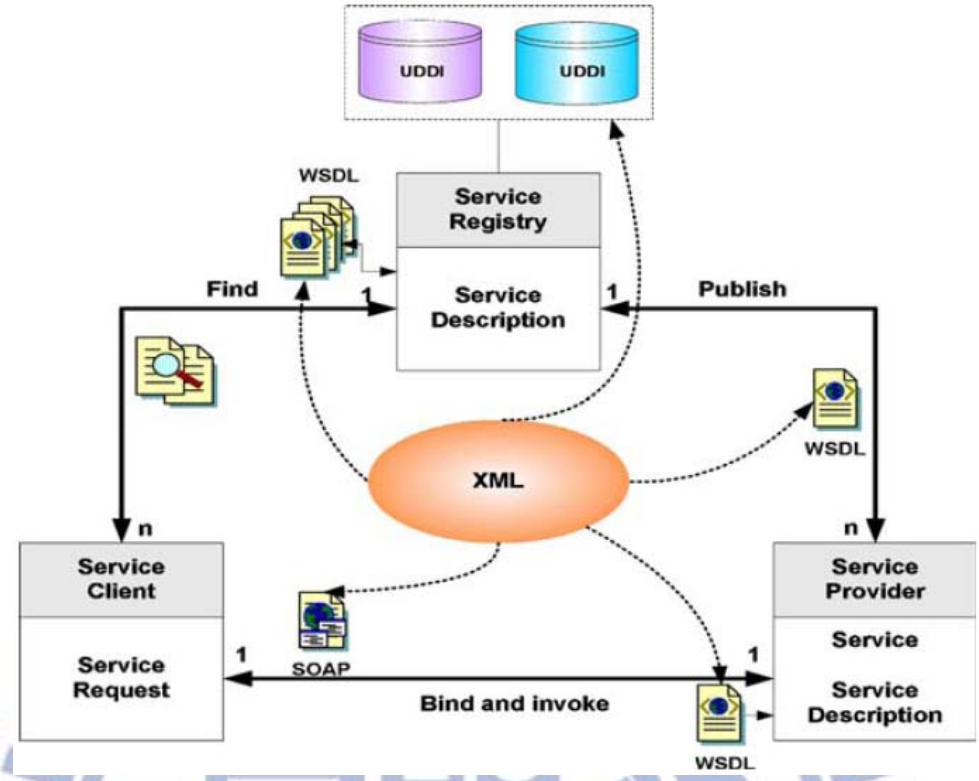


圖 6、Web service reference model[2]

## 2.4.2 SOAP

現今的環境中，常見 Web Service 利用 HTTP(Hypertext Transfer Protocol)通訊協定及 SOAP(Simple Object Access Protocol)來提供服務。

SOAP 是一個專為 Web 上結構化(structured)與典型化(Typed)的資料做交換設計的 XML-based 的協定，除了可以結合各類型現有網際網路通訊協定與格式；SOAP 亦提供 Message System 對 Remote Procedure Calls (RPCs)廣泛範圍的應用。

遵從 SOAP 規格所開發的 Web Service 可以輕易的互相結合使用，因此可以擺脫對於開發平台以及語言的相依性，其主要原因是 SOAP 提供了開放性的存取介面，藉由 SOAP，各家廠商獨立開發的系統皆可以因此相互整合，不會因為異質性的平台問題而需要重新開發。

# 2.5 Windows Communication Foundation(WCF)

## 2.5.1 WCF 基本概念及架構

Windows Communication Foundation (WCF) 是 Microsoft 建置服務導向應用程式的統一程式設計模型，它是.Net Framework 的一部分，由.Net Framework 3.0 開始引入，與 Windows Presentation Foundation 及 Windows Workflow Foundation 為新一代 Windows 作業系統的三個重要應用程序開發類別。開發人員可運用它來建置安全、可靠與可交易性的專案，而這些專案會進行跨平台的整合，建立服務端和客戶端之間的訊息傳遞，相同結構的 API 可在同一個電腦系統或位於其他電腦透過網際網路存取的方式，建立和其它應用程式間的通訊。

WCF 是以訊息通訊的概念為基礎，此模型的區別為『客戶端』(Clinet)和『服務』(Service)，任何可模型化為訊息(EX: Web Service 或 MSMQ)的資料，皆可在此兩端點間傳送，此兩端點是傳送或接收訊息的位置，其會定義訊息交換所需的所有資訊。

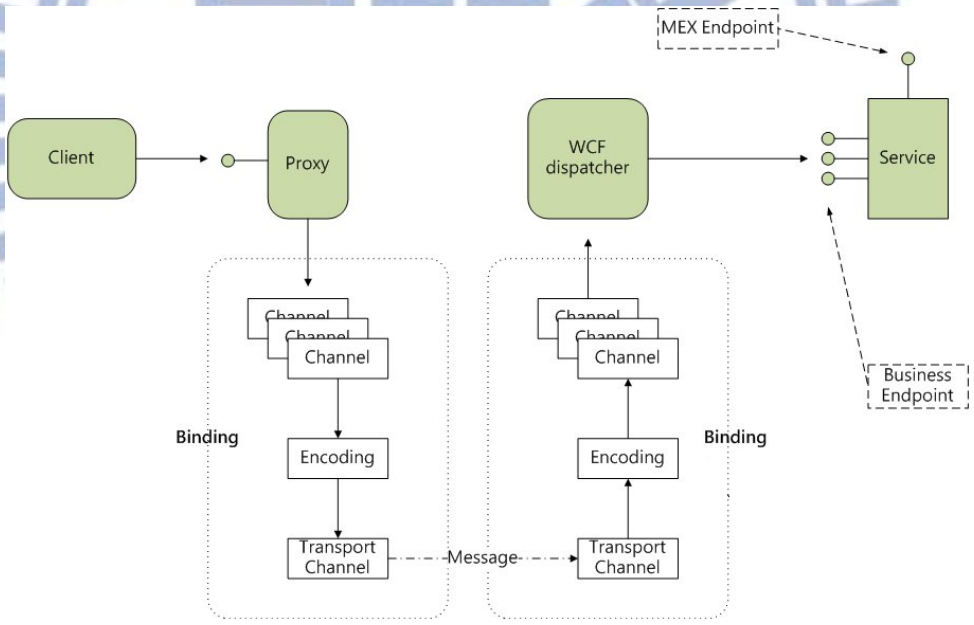


圖 7、WCF通訊示意圖[12]

於 Service 端會有 Endpoint，此為 Service 公佈給其他程式呼叫的介面，而 Service 可自行決定是否需要公開每個 Business Endpoint 的資訊。Client 端可透過 Proxy 呼叫遠端 WCF 所公開的介面(Operation)，將 Meaasge 送給遠端的 Service，而在 Proxy 中隱藏了 WCF 的實作細節，因此可方便 Client 使用。Binding 定義了 client 端如何與 service 端連線和通訊，包含傳輸、編碼、安全性設定、可靠性設定…等。WCF 程式最簡單的情況下

會只包含 Transport Channel 及 Encoding Channel。WCF 會根據 endpoint 中所定義的 binding 來初始化 channel stack 與 channel listener，其中 channel listener 會接聽指定的統一識別資源(Uniform Resource Identifier, URI)來接收訊息，收到訊息後將訊息傳到 transport channel，transport channel 再將訊息傳送給其它 channel。

## 2.5.2 WCF 優點

WCF使用WS的標準，因此使用者可以建立起服務導向架構的應用程式，透過這種方式可以使得程式有了鬆散耦合的優點，而並非Hardcode形式的編碼方式，任何平台上的系統，只要能夠遵守其合約及可互相溝通。在安全性考量上，WCF提供了諸如SSL 或 WS-Secure Conversation 等公認的標準可實作，因此在安全性這一塊上，也有了完善的建置[11]。

WCF 提供多種的訊息交換方式，其中最常見的為要求/回覆的模式，即某端點向另外一個端點要求資料，然後由另一端點予以回覆，其他模式還包含了單向訊息或更為複雜的多工交換模式等，透過這些訊息的交換方式，提供予程式的開發人員更方便的程式開發方式。而在編碼方式上，最常用的通訊協定與編碼方式為傳送文字編碼的 SOAP 訊息，其所使用的是全球資訊網泛用的超文字傳輸通訊協定 (HTTP)。WCF 對於傳送的訊息一率會先存於資料庫中，萬一不幸發生了通訊中斷的情況，也可在重新恢復通訊後，繼續進行訊息的交換。

## 2.5.3 MSMQ

Microsoft Message Queuing (MSMQ)是由 Microsoft 從 Windows NT 和 Windows 95 時開發實作的一種 Message Queue。其允許應用程式執行在同步或不同步的時間和異質網路系統交換訊息。

MSMQ 工作會使用以下的幾種訊息類型之一：檔案、字串、字串訊息變數及其值等方式，透過使用 MSMQ 調整企業內的作業方式，如果目的端目前無法使用或在忙碌中，可將訊息排入佇列中稍後在傳送。

圖 8 顯示了佇列可容納多個發送程式和由多個接收應用程序讀取的消息。

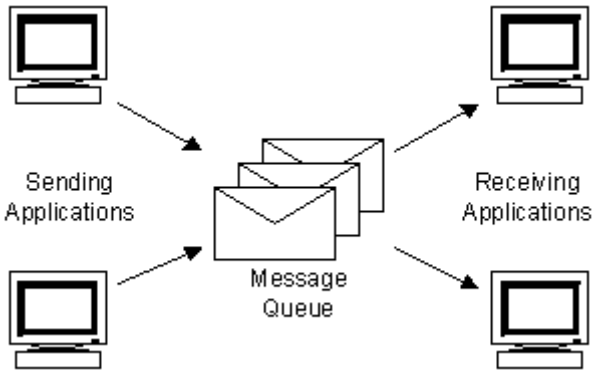


圖 8、MSMQ 示意圖[13]

# 第三章 相關研究

MES 系統在工廠的生產過程中，其可協助各個層級處理各項在生產上的問題或是決策工廠的生產方向，而現場即時資訊系統更是在收集最前端生產的資料上有著重要的地位，因此如何架構出一套良好的資訊系統，方便現場同仁的使用以及呈現各式的資料，就會是一項非常重要的課題。

本章節中，首先從 MEASA 白皮書中定義了何謂 MES 系統，並且其包含了哪些重要的功能；接著分別介紹以三個不同的方式所開發的 MES 系統，透過這三種不同的開發方式中，皆有其可供借鏡的優點，而本研究中則是分別結合了三套系統中部份的優點作為開發時的考量，其分別為三層化的元件設計架構、服務導向精神、以及以網頁化的系統呈現方式。

## 3.1 MES 系統的定義與功能

### 3.1.1 MES 系統定義

在MESA白皮書中，為製造執行系統(Manufacturing Execution Systems, MES)規範以下的定義[3]：

MES 系統傳遞生產資訊，使得產品從下單開始到完成品的產出其生產過程能夠達到最佳化。生產過程在進行時，MES 使用即時、正確的資料，提供適當的導引、回饋，對於生產條件改變時，可立即迅速的反應，如此可減少無附加價值的行為，達到更有效的生產作業及流程。MES 改善了設備的回收率，準時交貨率、庫存周轉率、邊際貢獻、現金流量績效，亦提供企業與供應商之間雙向溝通所需的生產資訊。

MES 系統是工廠管理層級和員工的核心關鍵系統。財務、物資計畫、後勤人員使用此系統以提高工作的準確性和即時性。其側重於現場的生產活動，並提供事件發生時的所有訊息。

MES 為一套由許多系統所結合而成的通稱，其結合了控制系統的規劃、產品設計、生產執行、物品銷售的交付機制、和客戶間的資訊交換。透過此 MES 系統，在企業間，不僅可使工廠的生產資訊系統化，也為製造商帶來了競爭力。

### 3.1.2 MES 系統的主要功能

MESA 白皮書中提及，基本的 MES 系統應包含以下的功能：

1. 資源分配和狀態（Resource Allocation and Status）

管理各項生產資源其中包括機器、工具、人員、原物料、其他設備和文件、物資等，使其在正式生產之前準備齊全。MES 亦提供資源的詳細歷史資料，並確保該流程所需的設備已確切的設置，同時還提供即時的狀態資訊。資源管理包括保留及分派資源以配合作業所需。

2.作業/詳細排程（Operations/Detail Scheduling）

根據產品優先權、屬性、特性以及各種的生產因素來排序，如果排定的順序可達到最佳化的效果的話，可以大幅減少設備調整的次數。

3.分派生產單位（Dispatching Production Units）

依據工單、順序、批量、批次及製程來管理生產單位的流程，根據工廠內所發生的事件即時指示所需進行或變更的作業，且能彈性的更改排程。

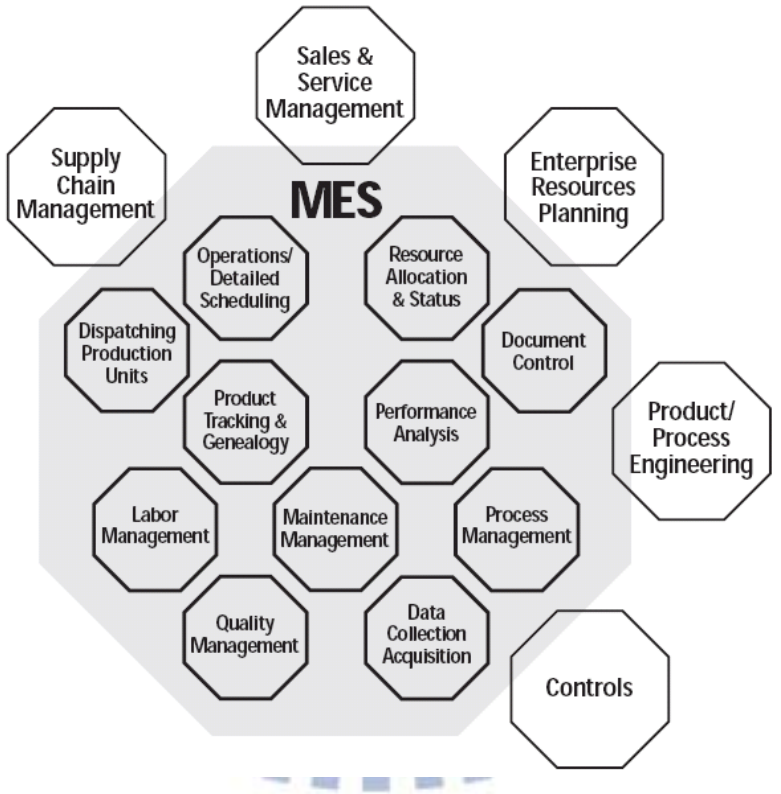


圖 9、MES 功能模組

4.文件管理（Document Control）

維護及保存各項相關表單或記錄，包括工單、配方(Recipe)、標準作業程序、工程變更等記錄，MES 還提供資料給操作人員或是傳送配方給予機台。

5.資料收集/獲得（Data Collection/Acquisition）

提供介面來獲得在製品以及生產參數等資料，而這些資料是即時的從設備上手動或自動收集而來。

6.人事管理（Labor Management）

提供全體人員即時的狀況資料，包括時間、出席狀態及人員的行蹤。

7.品質管理（Quality Management）

提供製造現場即時收集而來的資料進行分析、並同時提供適當的品質管理，最後針對問題提出改善的建議。

8.製程管理（Process Management）

於生產期間或監控特定機器設備，在進行生產的同時監控並自動修正或提供決策支援給予操作人員進行改善作業；之後並追蹤產品加工流程，也包括當產品超出容許誤差時警告現場人員進行處理。

9.維護管理（Maintenance Management）

追蹤設備和工具的維護狀態，安排定期或預防保養，以確定在製造過程中設備、工具是可用的。當問題發生時產生警報或回應，並維護過去發生的事件或問題的資料，以幫助診斷問題發生的成因。

10.產品追蹤及歷史記錄（Product Tracking and Genealogy）

提供產品於加工處理中的各項資訊，包括加工人員、原料供應商、批次、序號、目前生產條件及任何警告、重製或其他相關訊息。此追蹤功能會將所有相關資訊建立歷史記錄並加以維護。

11.績效分析（Performance Analysis）

提供實際製造情形、歷史資料和預期狀況三者比較的即時報告，包括資源利用率、資源可用性、產品週期…等資料，也包括統計製程管制/統計品質管制。此功能可產生定期的報告及即時的績效評估。

## 3.2 元件化應用架構

此節中將介紹以元件化架構的MES系統，這個架構是由操作性高、適應性強且可擴充的元件所組成，如此彈性化的設計可縮短軟體開發的時程；此外使用此元件化的架構開發對於客戶特殊的需求可花費最少的成本效益來達成目標[4]。

此元件式架構主要結合以下兩種方式互補而達成：

1.採用事件以及限制式為基礎的模組化方式來表現機台的狀態以及生產流程，其強調以事件和限制條件替代生產流程站點的管控方式，這種方式以各自獨立的作用屬性來表達其工作流程。

2.程式開發主要是以 C# .Net 語言開發，並以物件導向及元件式基礎作為

程式開發的技術。

本系統使用了三層式的架構，如圖 10 所示

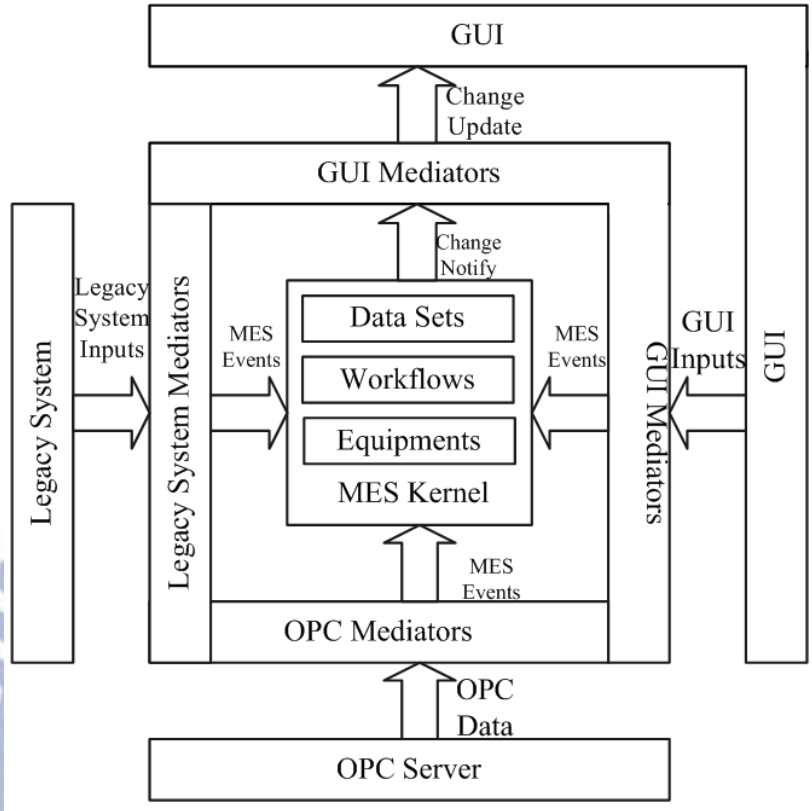


圖 10、Multi-Layer Architecture[4]

MES Kernel：此部分主要實作和 MES 系統有關的主要核心功能，所有和 MES 有關的資訊皆會於此組合並執行，Kernel 需要儲存並維護和操作有關係的資訊，例如：機台、工作流程、媒合資料…等。當 Event 於外部使用者介面發生，經由 Mediator layer 轉發到 Kernel，再第一時間會經由 Equipment 元件處理，接著再轉送到各自的 Workflow 元件處理。

External layer：以元件化及限制化的基礎實作存取介面，透過 OPC(OLE for Process Control) 使用者可控制存取工廠資料並執行符合規範的各項操作。典型的工作內容包含了檢視、編輯 MES 的各項資訊、操控機台行為等。

Mediator layer：介於 External layer 和 MES kernel 之間，其使用預先定義好的介面溝通規範，處理不同的輸入、輸出執行需求，因此可同步 Kernel 的內容資訊和外部的使用者介面的顯示。使用此種架構的方式，可以除去 Kernal 和 External layer 間的耦合程度，使其為一鬆散耦合的程式架構。

此系統將由 External layer 送往 Mediator layer 的事件稱為 UIEvent，而從 Mediator layer 到 MES Kernel 的事件稱為 MESEvent。在 MES Kernel 開發上使用 C#開發，其利用了委派(Delegate)的方式，來針對 MESEvent 事件做處理，當事件發生時會有一 EventHandler 產生，透過 EventSender 將此 Event 送至

EventReceiver，再交由各自對應的 HandlerMethod。而從 EventSender 送至 EventReceiver 時所攜帶的資料則為 EventArgs，如圖 11 所示：

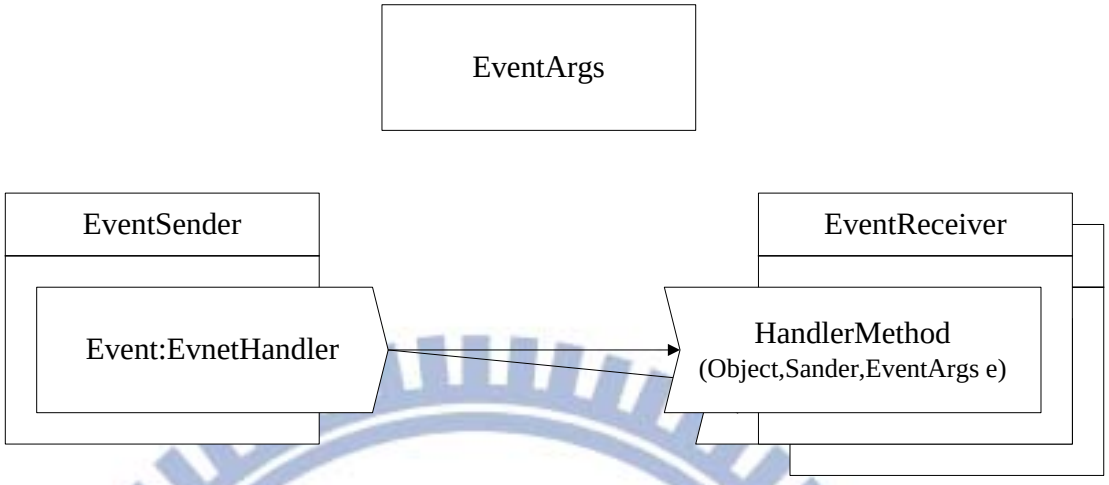


圖 11、Event/Handler範例[4]

於 MES Kernel 中，透過元件化的方式，將機台、工作流程以及兩者的媒合資訊，分成三種不同的元件，各元件中將會記錄了相關的資訊。當事件由 External layer 發生，傳遞予 Kernel 後，在經由 Handler 處理，之後媒合的訊息又再度反饋給 External layer，以此方式可得知現行的生產資訊是否正確。

在各 Layer 的設計上，其希望達成將環境中的資訊嚴謹的擷取和封裝成一 Component 元件，提供高可配置的元件和多樣性的 Form 和使用者互動。為了上述的目標分別對於 External layer 和 Mediator layer 定義出了以下幾項較為重要的介面規範元件：

External layer：

- IObjectInput：基礎的輸入物件，目的是傳送輸入的資料。
- IObjectOutout：基礎的輸出物件，將資訊輸出或呈現在 GUI 上。
- IObjectSelection：當事件發起時，從一堆的物件集合中選擇出所需的物件。
- IObjectListRepresentation：允許從物件集合中代理或操作物件的方法。
- IObjectEditor：當事件發起時，允許此物件對於物件做出編輯行為。
- OPCEventInput：當 OPC server 發出事件後，OPC server 經由此產生 IObjectInput 並設置其中的資訊。
- GUIEventInput：由 GUI Control 所控制、配置並產生 IObjectInput。

- GUIStateControl：實作一 IObjectOutout 物件並用圖型化的介面顯示機台…等狀態。

Mediator layer：

- MEDObjectInput：接收由 IObjectInput 所傳來的物件介面，並將其轉發至 MES Kernel。
- MEDObjectOutput：和 IObjectOutout 互相配合，連接 Kernel 和 External layer 的各項屬性資料，並於 GUI 上顯示。
- MEDOListRepresentation：連接於 Kernel 和 External layer 的物件集並和 IObjectListRepresentatione 共同合作，其作用於 Data table 中的資料變更時。
- MEDEditableListRepr：其依靠 IObjectListRepresentation 和 IObjectEditor 的介面，擁有和 MEDOListRepresentation 相同的功能但額外增加了 IEditedEvent 並更新內部的物件資料。

在此篇論文中，在 Multi-layer 的基礎上建立了以元件化開發方式的 MES 系統，開發者可透過各系統元件，對於生產中的各項變更，修改其元件的內容，以此可達到快速開發以及擴充性強、適應性高的系統。

此系統經由概要(Schema)以及元件庫來建立位於 MES Kernel 機台、流程間的關係，來處理生產流程上複雜的應用。而對於在 External layer 和 Mediator layer 定義出的介面規範可確保在此兩層間的互動是經由定義明確的方式來進行資料的溝通。

因使用 Multi-layer 的方式開發此系統，可達到將 MES Kernel 和 External layer 的 GUI 或是機台間上報的資料降低耦合的作用。另外使用元件化開發方式，使得系統元件在開發上可用性、擴展性、重用性提高。

## 3.3 服務導向架構平台

在 CIM 的資訊系統框架下存在著許多的商業服務或工廠端的服務，這些服務可能建置在異質的系統上，為了能夠整合在各個異質平台上的系統，且更加迅速的將所有的產品流程更動知會到各系統，於是便建立了一個目標為使用服務導向架構整合所有系統的平台，使得不論是外包或自製的系統能夠在異質平台上順利的溝通。

圖 11 為SCWSS[5]的架構圖，其他包含了三個層級，最頂層(Application and Portal Layer)為現有以存在的系統或是終端電腦上的系統。中間層則是該論文SCWSS的核心部份，包含了系統引擎和提供使用者存取操作的系統服務。其包含

了Web service規範和WS-Security或WS-Reliability。對底層則是系統的操作邏輯或者資料存放。最頂層和中間層的溝通則是採用SOAP協定，經由Web service和XML來做為溝通的橋樑。

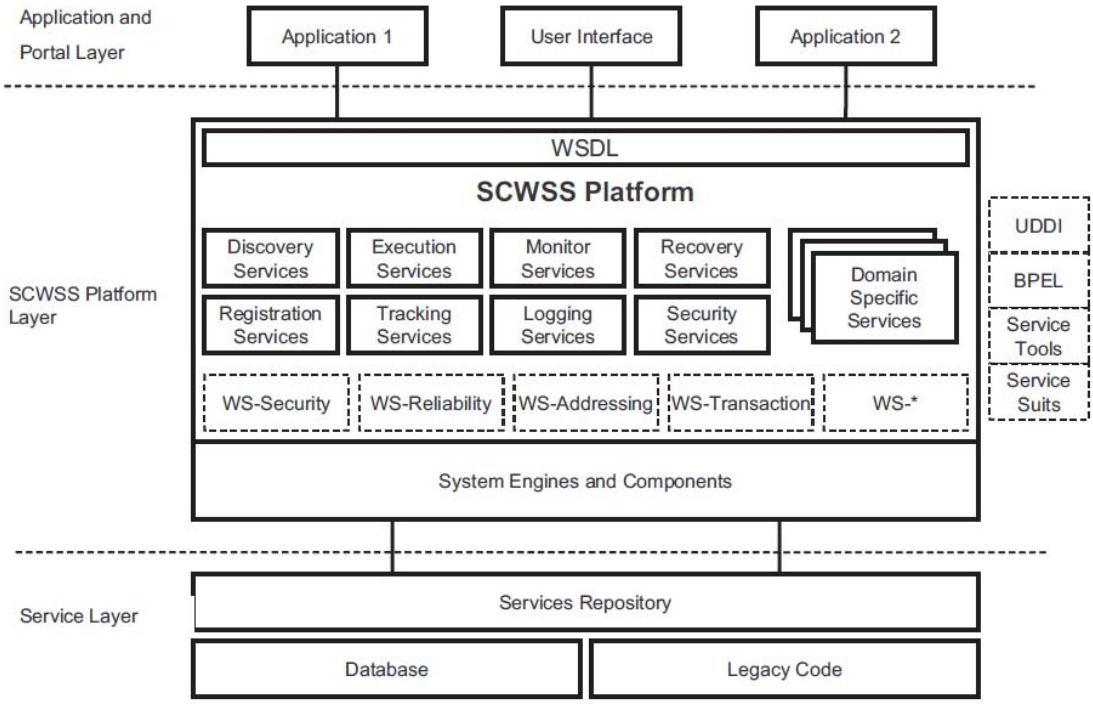


圖 12、CWSS 平台架構[5]

Discovery Service：SCWSS 是一儲存管理網路服務(Web Service)的模組，終端程式或使用者可透過呼叫此模組找到區域服務(Domain service)，為了令終端程式可找到網路服務，此模組收集了各種的網路服務。

Registration Service：在 SWCSS 系統中，此模組可給終端程式或使用者註冊區域服務，註冊區域服務的提供者也須將區域服務所需的各項資訊提供給此模組，而此模組只有經過授權的應用程式或使用者才可呼叫使用。

Execution Service：終端程式或使用者會產生許多的程序或工作，而這些程序可經由提供所需的資訊，透過此模組來呼叫一或多個區域服務。而此模組提供許多不同的方法執行服務，例如：同步執行、非同步執行、強迫執行…等方式。

Tracking Service：其主要功提供 SCWSS 平台可查詢各服務當前執行的狀態；此外 Tracking Service 亦提供終端程式可追蹤網路服務的機能。

Monitor Service：為了維持 SCWSS 系統的穩定和安全，而建立此監控機制，在此服務執行期間，將會呼叫 Logging Service 將各服務的操作狀態記錄下來，並將系統發生錯誤時的資訊儲存於資料庫中。

Logging Service：此服務模組允許 SCWSS 平台或終端程式記錄下系統的狀

態，以方便在系統在失去作用後，可讓開發者快速的了解原因，找出問題的癥結點所在。

Security Service：此模組建立在 WS-Security 規範上，為了確保系統執行上的安全，因此所有網路服務皆需通過此模組的允許及授權後，才會被執行。

Recovery Service：當系統不正常運作或失去作用後，此模組會主動修復錯誤以維持系統的正常。

System Engines and Components：系統引擎是為了改善系統執行效能及能力並提高系統彈性而存在，例如建立依統計引擎計算生產資訊，並回報有意義的數值來分析預測機台的動作，以此來控制製程。而在 SCWSS 系統上也會建立許多的元件(Components)，這些元件為各項網路服務的基礎功能或演算法，其實現了各項網路服務在系統上的行為模式;這些元件同時也可以是系統服務模組中所使用的部份邏輯功能。

此系統提出以服務導向架構的概念方式來達成系統整合目標，在平台主要中包含了系統服務、系統引擎、系統元件三個部份來架構 SWCSS 系統。其存在是為了將 CIM 框架下的異質系統溝通上針對不足處做出補強。使用此種架構的方式，使得各系統間透過使用網路服務，可以互相溝通，而無須針對不同系統皆撰寫各自獨立的溝通介面。

透過使用此平台，系統間溝通更加便利，且對於產品生產流程中變動，相關系統能夠更加快速的做出反應，進而提升工廠效能與效率。

## 3.4 網頁式存取 MES 系統資訊

此篇專利中，主要目的為提供一種透過Internet World wide Web的方式存取MES系統中的各項資訊，且提供對於機密資訊適當的安全存取方式。使用者透過從網路瀏覽器自動下載JAVA程式後，便可經由此的程式存取MES server部份的資料[6]。

圖 13 中，Workstation 12A-12N 所代表者為 MES 中的各項系統，如生產系統、庫存管理系統、品管系統、排程系統、原物料管理系統…等，這些資料透過 Bus 13 儲存在 Database 11 中。各種 MES 的資訊透過 Workstation 將其存入 Database 中。

對於生產中的各項資訊，顧客經常是有興趣去查詢這些訊息，因此可透過 Web Browser 22A-22N 去查詢相關的資訊。Network 21 可能是 Internet 或 Intranet，當 Web Browser 22A-22N 透過 Network 21 連至 Web page server 20 時，Web Browser 變會自動下載並安裝設置一小程式，顧客便可以透過這種方式對生產資訊查詢並提出質問。

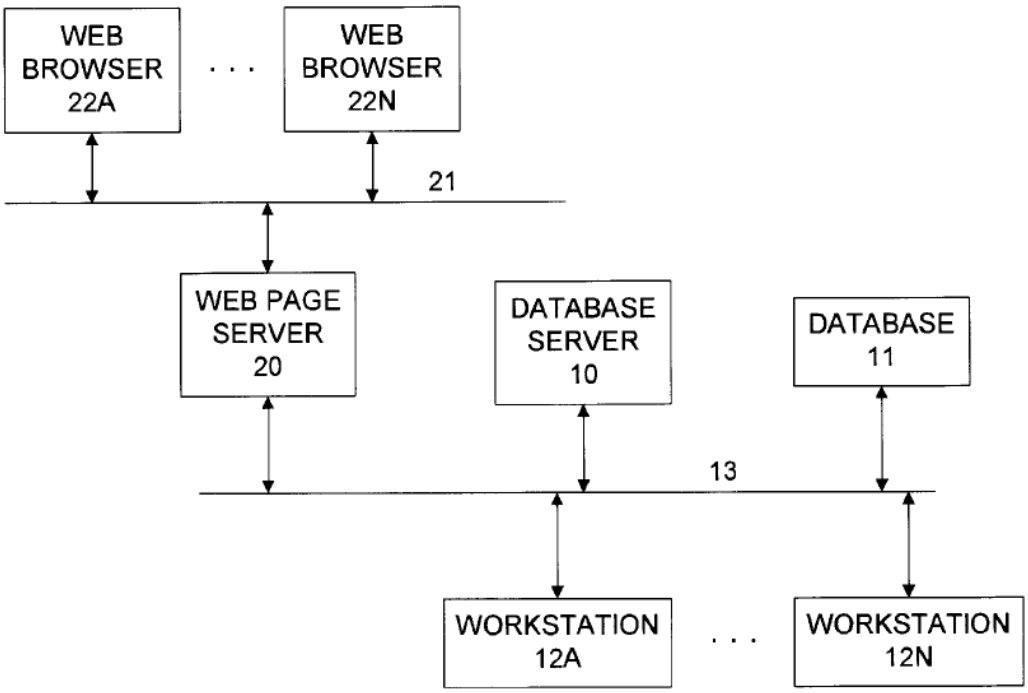


圖 13、Web Access 架構圖

當 Web Browser 下載並安裝程式後，使用者需輸入一組帳號密碼來完成權限的控管，此組帳號密碼則分散於 MES Administrator 或 Web Page Server 20 來管理。使用者在初次使用時，必須先提供註冊的資訊，並在完成註冊且登入後，由系統控管提供其對應的權限，通過此種方式可做到權限控管的目的，而 Web Page Server 20 亦會紀錄下使用者對於資料的查詢紀錄，以方便系統管理者後續的追蹤。

除了權限控管的目的外，使用者可開啟額外的 HTML 網頁，透過 Web Page Server 20 連接 Database server 10，進而得到 MES 的各項生產製程資訊。使用這種方式，Web Page Server 20 就充當了防火牆的腳色，可以避免使用者直接存取 Database 11 並提供有限的資訊給予使用者。

上述的 Web Browser 所下載的小程式則是由 JAVA 來實作，如此可在各種的 Web Browser 平台，例如：Microsoft Internet Explorer、Netscape…等執行。

現行多數的 MES 系統多為套裝軟體，而其內部許多的資訊，並無法從套裝軟體中直接呈現予使用者，透過文中提供以 JAVA 為開發語言的 Web Page Server 連接 MES 系統以及數位網路，再經由網頁瀏覽器，即可呈現給廠商或是公司內部的使用者。

經由透過帳號密碼及權限控管，可依照不同的權限給予對應的資訊。在系統的管理上，也可透過 Web Page Server 來查詢使用者所查詢的紀錄，已此來完成監控的目的。

經由 Web Page Server 連接後端的資料庫系統，在生產資訊的統整上，可加快資料的解析，更加方便使用者判斷目前的生產狀況。

# 第四章 豐富互動式網際網路應用於現場即時資訊系統

即時過帳系統在整個 MES 的架構中，各層級的管理人員皆需仰賴此系統方可得到許多現場的資訊，之後再經由各種方法去分析。然而往往這些收集到的資料都是繁雜、龐大的資訊，因此如何將此部分資訊有效管理、收集，勢必就成為一個很重要的課題。或許可以說，在一個資訊化的工廠中，若沒有良好的即時過帳系統配合，根本就無法有效的進行管理，從而導致許多的浪費、錯誤決策的發生。

在多數 MES 系統中，資料的收集、整合、溝通，往往是一個浩大的工程，針對不同的資料來源或資料接收端，可能會有不同的資訊傳遞格式或通訊協定，例如：從使用者介面上傳的各項指令、製程機台和 MES server 間的通訊、不同系統間的資料交換…等，系統因此需要針對每一種的狀況做出對應的新增或修改。本文中以服務導向的方式，設計一中繼程式”Application Server”可以連接交換不同資料來源，舉凡如最基本的過帳系統所上傳的資訊、機台溝通交握的訊息、各系統間的通訊等。Application Server 將這些資料使用統一的格式，傳遞給最後端的 MES Host Server，如此可使後端的 MES Host Server 在開發的過程中，不必為了許多不同的資料來源而需要開發多種不同的通訊方式且可減少重新編譯的 MES server 的行為模式。

在現場即時資訊系統使用者介面上，現場使用的電腦作業系統版本可能不盡相同，因此造成部分的電腦需額外安裝程式套件或額外更新元件等，如此一來，方可正常引用所需 OCX 或 DLL 元件，為達成不須安裝特定程式且消除各電腦系統平台的差異，本系統採用以網頁瀏覽器配合 Flash 方式來呈現使用者介面；基於 RIA 提供了高可用性、高彈性的開發特性，使得在開發上更佳便利快速，以達成節省開發時間的目標。

## 4.1 系統概述

MES 系統之所以對於製造業而言非常重要的原因即在於，其收集了舉凡從工單開立、製程流程的建立、生產過程中每批在製品的生產狀況、Ship 出貨時，所要給予的下游廠商的資訊…等。這些資料經過整理後，對於公司的每一個生產政策或方針，皆有深遠的影響。管理階層可以了解廠內目前的存貨量以及物料的備存是否足夠，並可和業務單位配合，以此決定生產產品走向。現場生產單位可以透過即時資訊系統，對在製品進行管控，確認其良

率是否符合要求，抑或某條生產線目前是否處於滿載的狀態，動態的分配各生產線的狀態來達到最佳化的生產效率。

多數的 MES 系統架構為 Server 端以及 Client 端兩個部份的設計架構，本文針對此系統的架構進行了改善的同時，將系統主要分為三個部份，分別為後端 MES Host Server、中繼程式 Application Server 以及前端介面 Client User Interface，如圖 14 所示，除了上述的兩個部份外，另外還包含了接收 MES 或後端程式所送出的主動訊息廣播程式，以及同樣執行於 Client 端的主動訊息接收及周邊硬體控制程式。

本研究中所使用的 MES Host Server 目前使用的系統為 NanoTrack，此 Server 程式為本土廠商所開發，且在開發階段的過程中，便已買斷，因此 NanoTrack 部份程式在本系統中修改成符合廠內的過帳方式後，在搭配本研究中的系統使用。在本研究中主要針對 Application Server 和 Client User Interface 進行了實作且搭配另外兩個程式”主動訊息廣播程式”，”主動訊息接收及週邊硬體控制程式”使得使用者介面的功能更加的完善，圖 14 即為系統的示意圖。

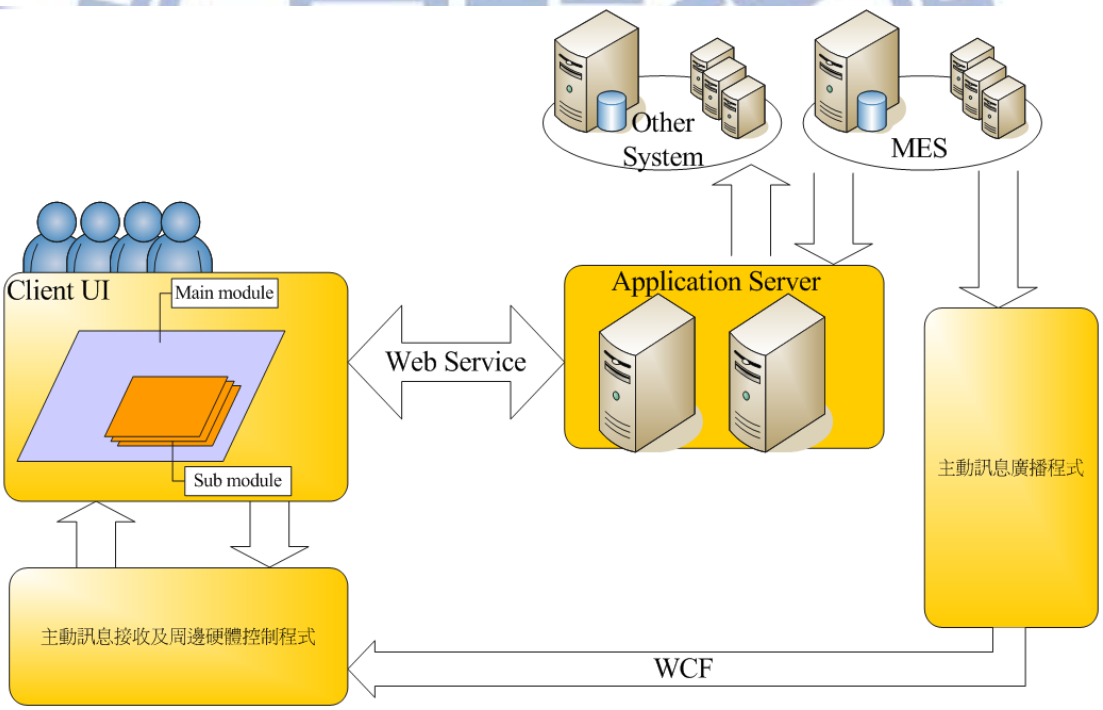


圖 14、系統示意圖

目前廠內的生產於多數情況下，皆為自動化的生產方式，於是乎 MES Host Server 主要的功能便為針對現場在製品的製程、派貨邏輯做出卡關的動作；於在製品完成製程後，儲存、判斷機台上拋在製品生產過程中的資訊正確性與否，並將在製品的製程站點往前推進。在上述的動作中，若檢查到

在製品製程站點、狀態、位置或上傳的資料不正確，Host Server 便會將該在製品於系統上保留其狀態，並回傳訊息給前端的使用者，告知有不合乎正確生產狀態的情況發生，並期許使用者針對該在製品進行檢查。

因為各個資料來源端應用程式的異質性，應如何設計出一個讓外部程式介面模組能更容易使用並交換資訊進而互相溝通，此時 Application Server 就扮演了一個很重要的角色。Application Server 以服務導向的方式為設計理念，使用了可延伸標示語言(Extensible Markup Language, XML)的文件格式作為本系統的基礎用來和前端的 Client UI 以及其他異質平台的外部程式溝通的橋樑，所有和 Application Server 交握的程式以透過網路服務(Web Service)並符合定義的 XML 的規範方式互相溝通；而 Application Server 在連接後端 Server 或其他的子系統部分，則會針對不同的後端模組建立起相對應的工作模組，將訊息轉換成該後端模組可以接收的格式。

以 Flash 為開發語言所設計的 Client UI，在現場的使用者眼中佔了十分重要的位置，對使用者而言，其可依據系統上所顯示的資訊，了解各項在製品、載具、機台，物料的狀態。其可透過網路服務與 Application Server 聯繫，在加上多數的瀏覽器上只須安裝 Flash player 即可執行的特性，提供了高可用性以及便利性。針對程式開發者而言，可以透過 ActionScript 控制程式中的 Component 元件，使得畫面的呈現上更加的豐富，互動性質更高。

## 4.2 系統架構

### 4.2.1 Application server 架構

在 MES 即時資訊系統中，常因為業務量的不斷擴張，系統勢必需要和越來越多的平台接觸，然而各個系統間有各自的通訊方式，這造成了 MES Host server 於後來必須為了上述的理由不斷的因應新的溝通方式，而須追加通訊介面，導致系統維護上困難外，也難以整合，造成系統開發者的困擾。在本文中為解決此一困境，於是提出建構一中介程式，透過此中介程式達到服務導向架構(Service-oriented architecture , SOA)目的，因此 Application Server 就此誕生。其提供了以網路服務的方式，令各種不同的程式間以 XML 電子文本封裝成標準的網路服務描述語言(Web Services Description Language, WSDL)，只要符合規範者皆可透過 Web Service 進行溝通。

Application Server 以.Net Framework 3.5 為程式開發的基礎並以 C# 語言開發，其架構在 Windows 2003 server 平台上，所有前端的需求透過 IIS 中的網路服務，並將訊息轉至 MSMQ，之後再由 Application Server 接收。

圖 15 為 Application Server 的程式設計架構，Application Server 中 Components 元件是構成此程式很重要的一個部份，每一個 Component 元件分別應對於從網路服務而來的不同請求，對於這些請求做出相應的行為模式，這些 Component 元件會依據請求內容檢查資料是否正確，並明確定義出要轉發的後端系統為何，當 Component 元件越多，功能越齊全後，在系統開發上，即可重複使用這些元件，可達到快速開發的目標。

在 Application Server 初始化的過程中，Message Dispatcher 便會載入對應於每一訊息處理的 Component 元件，而當接收到從 Communicator 送來的需求後，執行相對應於需求的 Components 元件並完成其邏輯處理、驗證資料格式正確與否、轉發到哪一個後端系統處理、處理從後端系統回傳的訊息其該如何包裝內容格式並傳回到前端的需求請求程式。

Application Server 透過 Communicator 將 MSMQ 中的需求訊息經由窺視的指令讀取回來後，判斷 Application Server 是否可以處理此需求訊息，之後再將此需求訊息註冊於 Queue Manager 並經由 Message Dispatcher 轉發。而如何判斷 Application Server 是否可以處理這些需求訊息，便仰賴 Message Dispatcher 所載入的 Component 元件，以避免接收了無法處理的需求訊息。

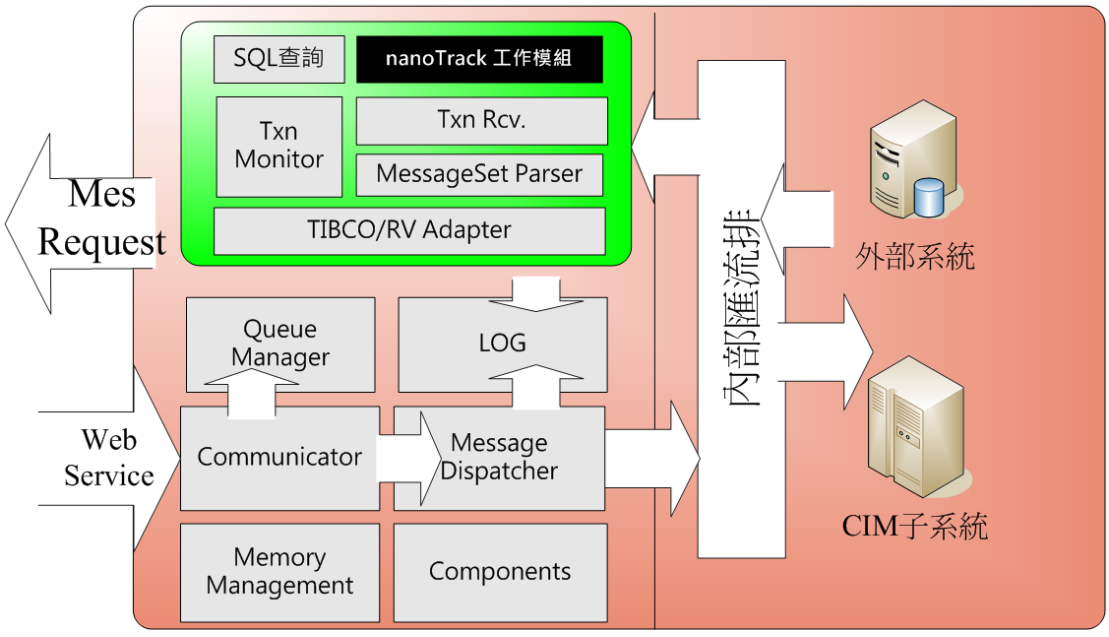


圖 15、Application server 架構

當 Communicator 接收了需求訊息後，首先便會在 Queue Manager 中註冊該需求，此行為是為了當訊息經由後端系統處理後，將會回傳訊息給予前端的程式，而經由此註冊的行為，Application Server 才能確切的知道這個訊息是要傳回予那一個前端的程式。當註冊完需求之後 Communicator 亦會將訊息中的資料經由 Message Dispatcher 中的 Component 元件處理。

當需求經由 Message Dispatcher 送出，透過內部匯流排，被 nanoTrack 工作模組接收後(nanoTrack 為後端的 MES Host Server)，此模組會針對此需求的內容轉換成後端程式可接收的格式以及溝通方式，例如在此模組中，轉換成 TIBCO\RV 的溝通方式送出給予 HOST Server。而如果是送往其他子系統或外部系統的需求，也同樣會建置有相對應的工作模組，進而透過這些模組溝通，圖 15 中右邊所顯示的外部系統和 CIM 子系統即代表著 Application Server 可透過不同的工作模組互相的溝通訊息。

在同一台主機上的不同 Application server 中，可以透過 Memory Management 互相溝通資訊，Memory management 會映射一共享記憶體。在這共享記憶體中，每個 Application server 都可以存取資訊。

Log 模組的部份則是會記錄下所有的 Log 資訊，以供後續問題的查詢及排除，方便在除錯以及追朔資料時找出確切的資訊。

### 4.2.2 Client UI 架構

Client UI 是現場即時資訊系統中和使用者互動最多的介面，本文中使用者介面的設計上，為了達成以下幾點目標：

1. 避免安裝特定程式。
2. 消弭各異質平台間的差異，使系統可以跨平台操作。
3. 模組化新增使用者介面上的子功能，方便程式設計者快速開發及替換。
4. 豐富的使用者介面資訊，可和使用者達成良好互動。

如何完成上述條件，便成為在使用者介面開發時一個很重要的軟體評估因素，而最終決定以網頁瀏覽器的平台內嵌 Flash 的元件方式來進行開發。

Flash 是一種廣泛使用的多媒體編輯程式，其使用了向量圖形(Vector Graphics)的繪圖方式，之後產生出 SWF(Shockwave Flash)影片檔，並透過 ActionScript 來操控 SWF 檔案中的各項元件，以此方式便可達成和使用者良好互動的效果。

在 Flash 的網站設計架構中，許多程式開發者都會將所有的元件以一個 SWF 的方式匯出，用這種方式製作 Flash 網站會遇到每次匯出 SWF 都必須花費許久的時間，且在除錯上也十分的不方便，且最後 SWF 檔案通常都很大，如此一來使用者每次在下載都必須花費許久的時間等待，造成可用性大幅下降。為了改善這項缺點在 Client UI 的設計上，主要是以一個 Main module 作為整個 Flash 網站的大框架，之後再針對每一個子功能做開發，而每一個子功能即為獨立的 Sub module。如此一來使用者在操作此介面的時候，初始化階段只會下載 Main module 框架，之後再針對所需要的功能做出點選，便會自動將子功能程序下載並執行，使用此種方法，下載獨立的子功能程序檔案，而並非全部下載，便可解決等待時間過長的情況發生。

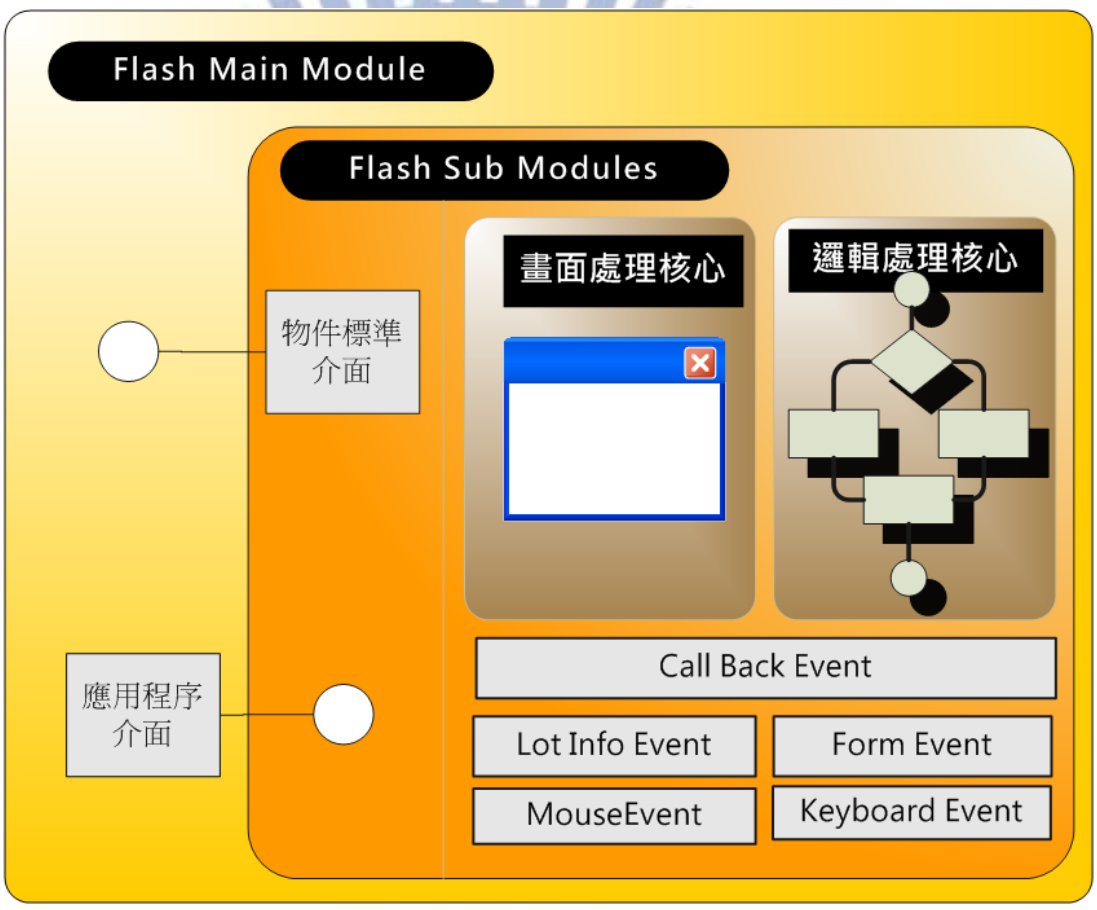


圖 16、Client UI 示意圖

瀏覽器於執行時，首先載入的便是 Main module 的框架，其提供以下幾點和 Sub module 互動的功能：

1. Sub module 可透過 Main module 對 Application Server 提出網路服務的請求及接收回覆的訊息，並將此訊息交由對應的 Sub module 處理顯示。

2. 將”主動訊息接收及周邊硬體控制程式”所接收到的訊息於畫面

上顯示，以方便使用者更加了解廠內的生產狀況。

3. 提供應用程序介面給予 Sub module 使用，對於 Sub module 開發，可提升開發效率。

4. 針對部分特殊的需求，會有需要操控硬體的情況的發生，此時 Main module 便會透過” 主動訊息接收及周邊硬體控制程式”，可控制如印表機，CVR 等的硬體設備。

Sub module 的開發中，透過規範物件標準介面的方式，每一個 Sub module 可被 Main module 所辨識，進而使用各個子功能模組。Sub module 中主要分成了三個部份：畫面處理核心、邏輯處理核心、以及 Event 的管理，如圖 16 所示。

因每個功能所要呈現的內容不盡相同，因此在畫面處理核心中主要是對不同的功能設計了各種的呈現方式，；而邏輯處理核心則會針對使用者的操作步驟，帶出相對應的行為模式或資料，並配合畫面處理核心將必需的資訊顯示或保留。

在各 Event 處理中，Call Back Event 是處理從 Application Server 回覆的訊息；Form Event 則是各個 Sub module 間相互動作時，所產生的行為模式；Mouse Event 和 Keyboard Event 則是分別針對鍵盤滑鼠所對應的行為。

Main module 的設計中，除了上述和 Sub module 互動的行為模式外，在另一方面為了能夠和 Application Server 溝通並保證的訊息正確性，於是在 Main module 框架中，設計了一 Queue Manager，而 Queue Manager 的運作方式如圖 16 所示。

在圖 17 中，當 Queue Manager 不斷的從 Sub module 接收訊息時，其會先將訊息置於一訊息佇列中，之後再由訊息提取的模組將佇列中的每一個訊息提取出來後經訊息轉換的模組處理。在訊息轉換模組中，會將訊息包裝成 XML 格式，以及分別在 Call Back Manager 中註冊和經由訊息發送模組將此 XML 訊息送給 Application server 處理。

而當 Application Server 回覆訊息後，將會由 Call Back Manager 所接收，並經由訊息轉換的模組將訊息轉換成 Flash 內容，配合之前在 Call Back Manager 中所註冊的資料，便可以將 Flash 內容回覆給予正確的 Sub module。

而 Timer 的功用主要來計算每個訊息送給 Application server 後，若是超過一定的時間，並沒有收到從 Application Server 的回覆，即自動判斷此訊息已經處理失敗，以免使用者不斷的在等待，亦或程式佔用系統資源。

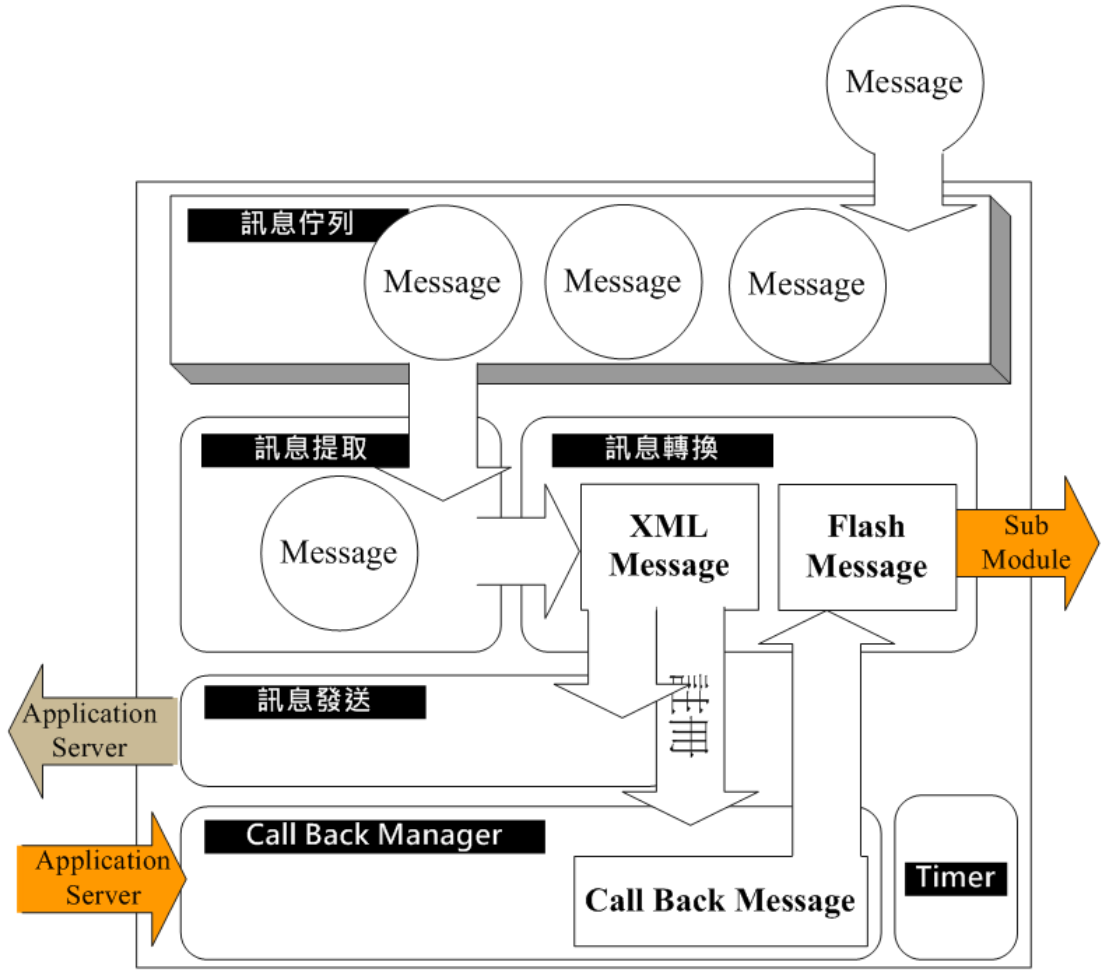


圖 17、Queue Manager

## 4.2.3 主動訊息廣播程式

主動訊息廣播程式的設計目的是為了方便現場的使用者能夠了解到目前廠內可能發生的狀況，例如：機台目前是否正常運作或處於停滯的狀態，各個在製品目前的生產狀況等，Host Server 會不斷的送出上述的內容，方便使用者對於廠內目前的狀態有充分的了解。

因主動式訊息是由 Host Server 發送，因此並不適合使用 Application Server 到 Client UI 的設計方式呈現在使用者介面上。主動訊息廣播程式會不斷的接收從 Host Server 送出的訊息，之後將這些訊息送到曾經註冊過的主動訊息接收及周邊硬體控制程式。

透過這種方式，可以減少 Client UI 為了不斷得到最新的在製品資訊或機台資訊一直送出訊息給 Application Server 的輪詢式方式得到這些資訊，大幅降低 Application Server 的負載。

### 4.2.4 主動訊息接受及周邊硬體控制程式

為了能夠接收主動式訊息，勢必在 Clinet 端會有一接收程式的存在，而主動訊息接收及周邊硬體控制程式的兩個功能中，其中一個就是接收從 Host server 來的訊息，之後再轉給 Client UI 並呈現在畫面上。

主動訊息接收及周邊硬體控制程式在初始化的過程中，首先會在主動訊息廣播程式中註冊，告知主動訊息廣播程式須將主動式訊息傳遞予特定的 Client，當接收到訊息後，以 WCF 方式溝通，將這些訊息給予 Client UI，之後呈現在使用者畫面上。

主動訊息接收及周邊硬體控制程式的第二個功能如字面上所示，便是操控一些週邊的硬體設施，例如：印表機、VCR…等相關產品，當 Client UI 欲使用某硬體時，可經由 WCF 的溝通方式，告知主動訊息接收及周邊硬體控制程式，以此來達到操作的目的。

# 第五章 系統實作與成果

在本章節中，將針對第四章所描述的系統設計想法實作，以期建構一套可快速開發、高可用性的即時過帳系統，首先將針介紹整個程式的執行環境，並且以一個簡單的例子從產品的生產到最後的出貨，中間可能會使用到的幾個比較常用且重要的功能做出描述。

## 5.1 系統環境

在本系統中，主要分為三個部份，分別為 MES Host Server、Application Server、Client UI 三個部份以及另外兩個附屬的程式，分別為主動訊息廣播程式、主動訊息接收及周邊硬體控制程式。因此在系統的建置上將分為三個主要的部份來介紹。

- MES Host Server：

  Host Server 主程式為 NanoTrack，開發語言為 C++，運行於 Red Hat ES3 作業系統上，搭配硬體為 DL-380 Blade 以及 4G 的記憶體。NanoTrack 主程式為在廠商尚在開發階段就已經買斷的產品，因此在本研究中，僅針對 NanoTrack 主程式做出部分的修改，以符合目前的生產模式。

  資料庫系統上，因本系統對於資料庫的查詢、變更非常的頻繁，為了確保系統可以正常使用，因此選用了 Oracle 9i 做為資料庫軟體。

- Application Server：

  程式開發使用 C#並搭配 .Net Framework 3.5 以上的版本。

  程式運行於 Windows 2003 Server 的作業系統，其上需安裝.Net Framework 3.5、IIS Web 伺服器、以及 MSMQ 的服務。

  硬體的配備上，為選用 DL-380 Blade 以及 4G 的記憶體。

  主動訊息廣播程式亦同時運行於 Application Server 的執行環境上。

- Client UI：

  Client UI 在開發上使用 Flash CS4，運行於 Windows XP 的作業系統，配合 IE 6 以上版本的網頁瀏覽器並安裝 Flash Player9.0 以上版

本便可正常執行。

硬體方面則是使用 Acer Aspire Revo 主機，其上搭配 Intel Atom 230 CPU 以及 2G 記憶體。

主動訊息接收及周邊硬體控制程式同時運行於此環境上，為了主動訊息接收及周邊硬體控制程式能夠正常運作，此時尚需安裝.Net Framework 3.5。

## 5.2 程式開發實作

### 5.2.1 過帳流程簡介

說明 Application Server 和 Client UI 的開發過程後實作畫面前，在本節中將會簡介在製品(WIP)過帳的概念，對於現場即時系統的過帳方式有基本的了解，而有了這些基本概念後，更有助於說明後續 Client UI 的功能說明，以及 Application Server 開發的 Components 元件。

在製品於工廠生產時，其會經過多道的製程，且在製品的數量也非常的龐大，因此為了有效的管理在製品的生產，便定義了部分的名稱，以下先介紹較常使用的幾個名稱：

- Component：

  此為在製品玻璃在系統上最基本的單位，其會對應出一獨立的 Component ID，而現場可經由 Component ID 來管理實體帳以及系統帳是否相同。。

- Lot：

  為複數 Components 的集合，每個集合會給予一個個別 Lot ID，通常 Lot 中的每片 Component 皆為已完成相同製程的產品。

- Cassette：

  每個 Lot 在搬送的過程中，都會裝載於一載具中，此載具使用 Cassette ID 稱呼，而每個 Cassette 也都是獨立的 ID，Lot 中的在製品等待進入某個站點製程的過程中，大部分時間，因為 Lot 中的玻璃都是裝在 CST 中，所以一個 Lot 便會對應一 Cassette ID。

- Product：

廠內所生產的產品，將因應客戶需求不同以及生產製程的改進，會將產品分為非常多的種類，而每一種產品及對應一個 Product ID。

◆ Flow：

每種產品會針對不同的需求，而微調生產所經過的製程站點，在 Flow 所規劃的流程中，會包含從製程起始到結束的站點名稱。而系統在過帳時，便會依據 Flow 中的站點，帶出現行資料或下一站點的資料，方便現場使用者了解現況。圖 18 為一簡易 Flow 流程圖。

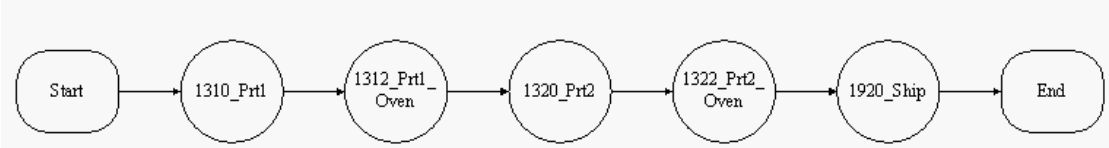


圖 18、簡易 Flow 流程圖

於開始在廠內生產時，首先會先從 SAP 的系統中，下載工單資訊並從中得到要生產的產品名稱、數量…等的資訊，MES 系統得到了這些資訊後，才會了解到目前這一批在製品是要使用何 Product、Flow 來生產，下載回來的工單會先經過啟用的動作後，代表廠內確定有排程生產此工單。

之後投產的過程中，便有一個圖 18 中 Start 的動作，其對應到系統上的一個 Lot start 功能，完成 Lot Start 的動作後代表此批 Lot 已正式在廠內生產，但不代表其已經投入某個製程，可能尚在某個倉儲中。

由圖 18 的簡易 Flow 中，可知道第一個站點為 1310_Prt1，在 Lot start 動作後，Lot 所帶出的站點名稱即為 1310_Prt1，之後當產品在 1310_Prt1 正式經過此製程時，須先將 Lot 作 TrackIn 的動作，如圖 19 所示，


圖 19、Lot 生產過程中系統行為模式

當完成 1310_Prt1 製程後，便會將該批 Lot 的系統帳做 TrackOut 的動作，代表此批 Lot 以徹底完成 1310_Prt1 製程。

該批生產 Lot 以 TrackIn、TrackOut 基本方式經過多個圖 18 中的站點，最後到達 1920_Ship 站點時，在此站點中同時會進行打包作業，因此在系統上，便會圖 19 使用 Packing and TrackOut 功能過帳，此時裝載 Lot 的載具，就會從 Cassette 變為 BOX(裝載出貨的紙箱)。之後再經由 Ship 功

能產生 Ship file 並列印出相對的標籤出貨。上述就是基本的在生產過程中，在系統上會產生的部份行為模式。

## 5.2.2 Application Server

對於過帳的基本流程有了些許了解後，本研究所實作的系統最主要目的便是建立一套透過 Application Server 可連接各系統的即時過帳系統，因此本節中首先介紹的便是 Application Server。

Application Server 設計的目標其中一點是透過建置大量的 Components 元件，因在 Client UI 上會使用到的 Components 元件有很高的重複性，因此透過詳細規劃 Components 元件並重複利用，可大量節省開發時間。使用 Application Server 程式時，必須先將對應於各個需求的 Components 元件撰寫完成，在 Server 初始化的過程中，便會載入大量的 Components 元件，這些元件中便會對於傳入的各項參數進行確認、邏輯判斷、並定義出不同的需求可能對應的後端程式為何。圖 20 為開發 Component 元件之畫面，於每個 Component 元件上皆會定義出其名稱、版本資料、邏輯行為等。

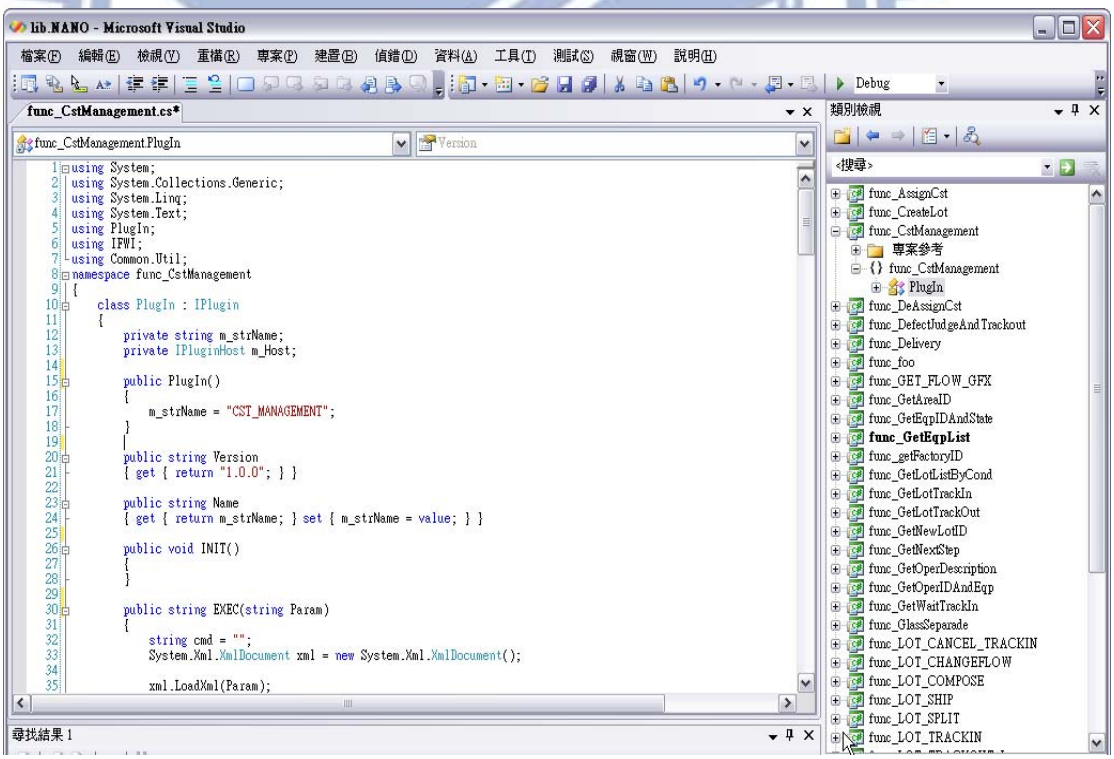


圖 20、Component 元件開發畫面

而圖 21 即為 Application Server 的畫面，其中可檢視此 Server 目前所載入的各項變數以及 Components 元件，畫面左側為執行階段時的各項參

數，而右邊的紅色框內容即為目前此所載入的 Components 元件及其版號。


圖 21、Application Server 介面

透過使用網路服務(Web Service)，將包裝好的 XML 電子文本傳遞給此 Server 後，接著 Server 便會依據 XML 中所定義的各種不同需求，決定套用哪一個 Component 元件，之後再將需求包裝成後端程式可接收的格式送出。下圖為一 TrackIn 動作之 XML 內容。

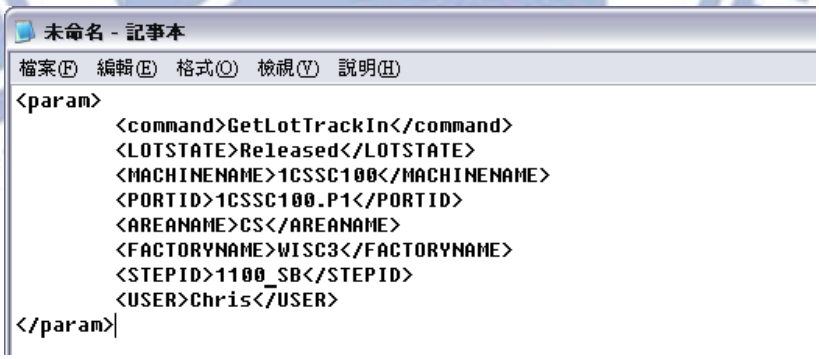


圖 22、Application Server 需求範例

其中<Command>所代表即為欲使用的 Components 元件，其餘的資料則為使用該元件時所必須附上的各項參數資料。

## 5.2.3 Client UI

本研究中，對於一般使用者而言，其唯一會接觸到的介面即為 Client UI

的過帳系統，而此介面的呈現方式則是選用以網頁瀏覽器的方式內嵌 Flash 程式。在此 Flash 程式中，其是由一 Main.swf 的框架下，之後再依據不同的功能而分別載入不同的子功能檔案。Main.swf 提供了大量的 API 元件提供予子功能呼叫引用，其中包括了畫面顯示效果以及和 Application Server 及周邊硬體通訊的應用程式介面供呼叫使用。

在個別子功能的開發上，首先透過 Flash 的元件編輯製作子功能畫面，如圖 23 中所表示為 Track Out 功能的開發畫面。

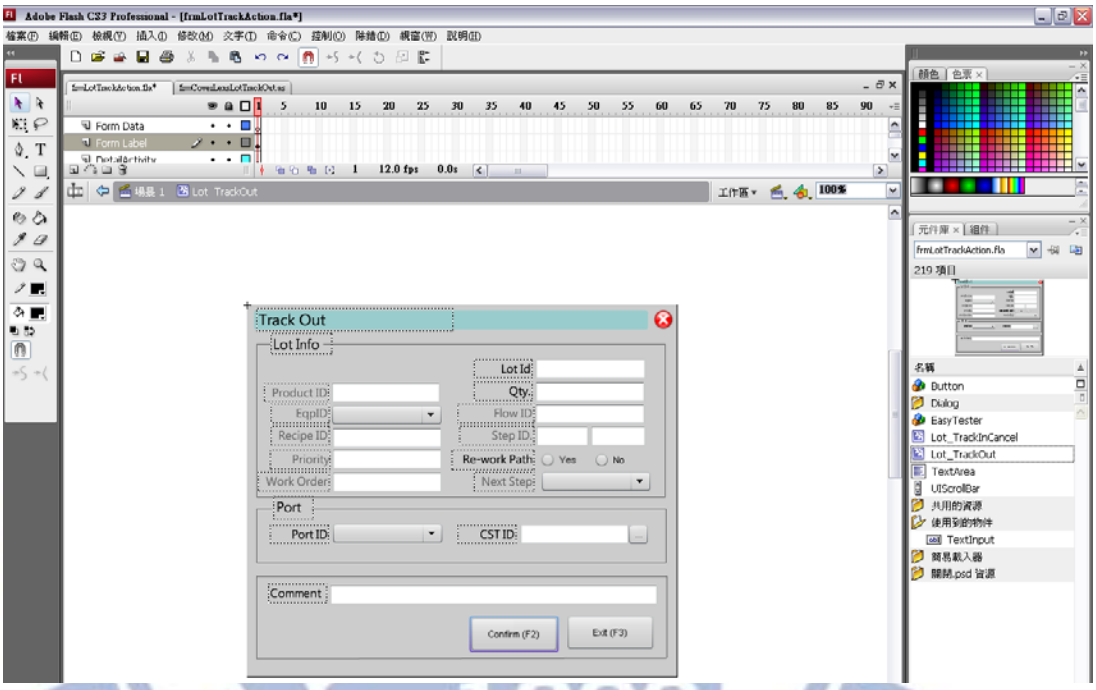


圖 23、Client UI 子功能開發畫面(1)

之後再配合 ActionScript 定義出子功能名稱，以方便 Main.swf 辨識，而子功能中相關的運作邏輯、事件觸發相關動作…等行為，亦是由 ActionScript 撰寫完成，之後再編譯成獨立的.swf 檔案，圖 24 即為 ActionScript 的開發介面。

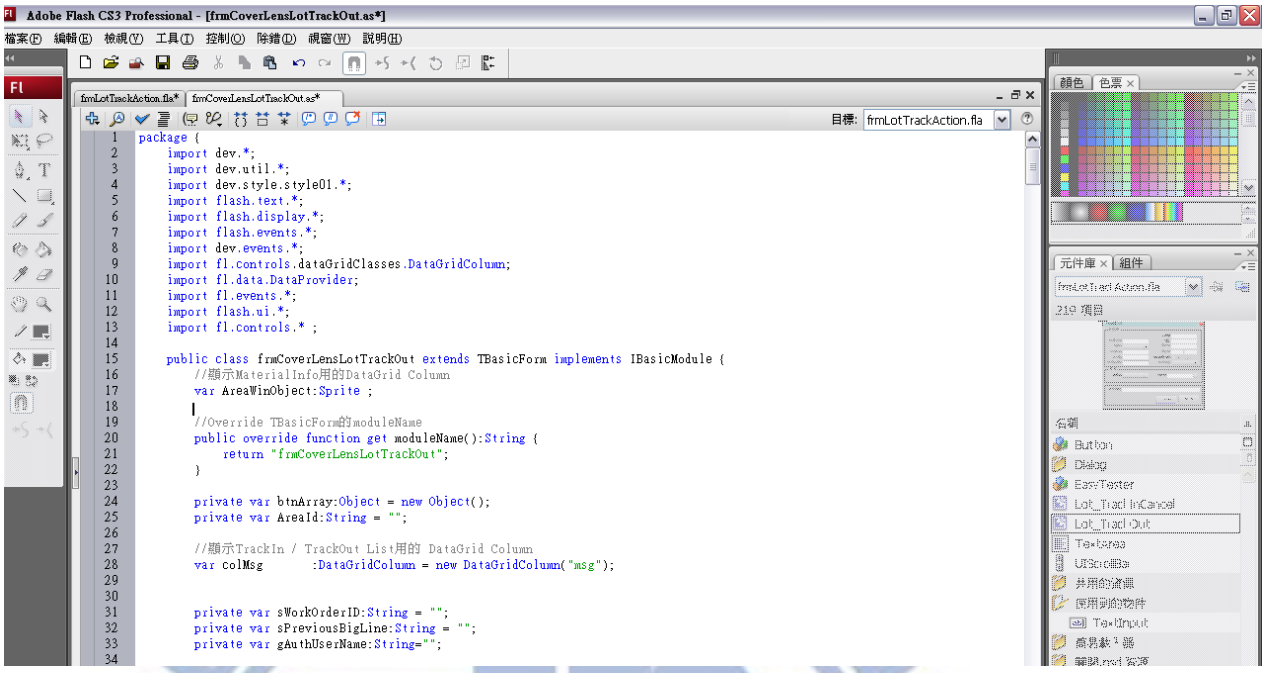


圖 24、Client UI 子功能開發畫面(2)

了解完程式開發的部份後，接著將以一批產品製作生產製作的過程中，其會在系統帳面上所使用到的主要幾個基本功能以及使用的時機做出介紹。

當中央經管在 SAP 上開立即將要生產的工單後，本系統將會透過”工單建立”的功能，從 SAP 上下載生產該工單所需的各種資訊，其中包含了產品名稱、數量、生產期限、生產物料、生產流程…等相關資訊。

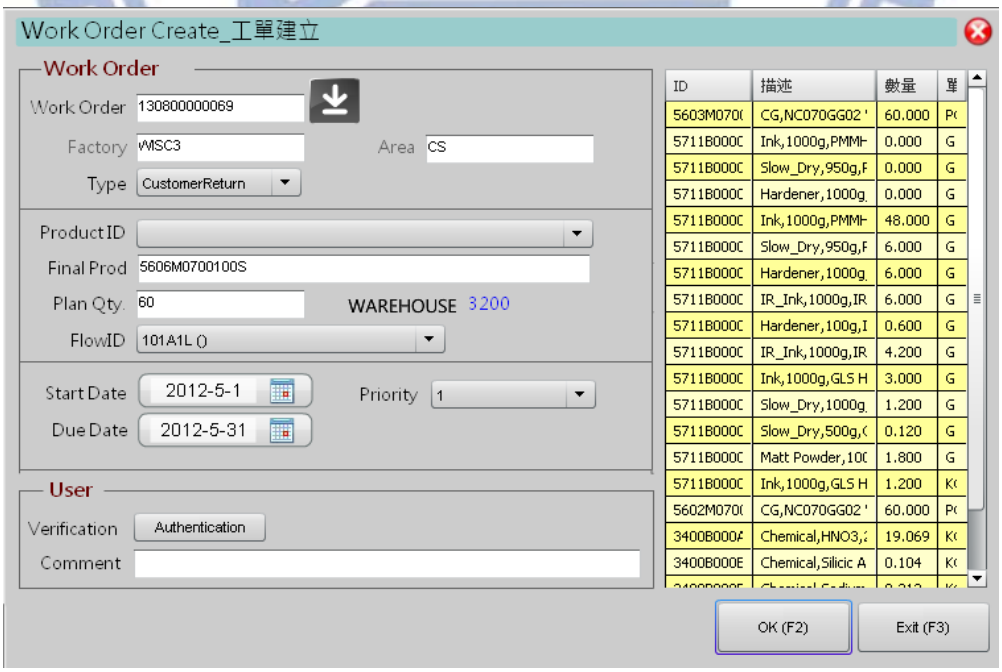


圖 25、工單建立

在工單建立以及啟用之後，待現場同仁領完原物料後，之後再將原物料

分成多個 Lot，此時在系統上會經歷一個”Lot Start”的動作，其代表著每一批分出來的 Lot 已經可以正式投產。

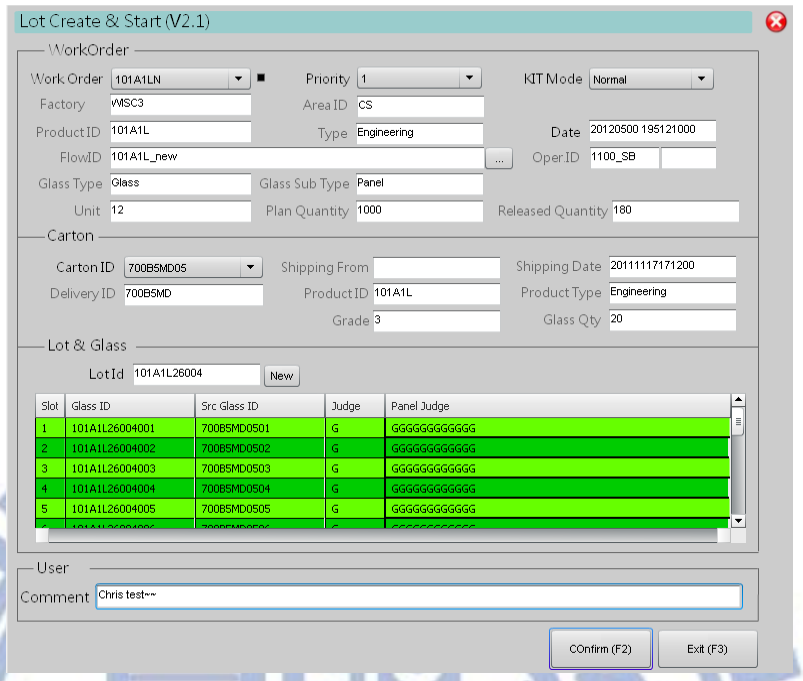


圖 26、Lot Start

在生產的過程中，若對製程的順序有疑慮，可隨時點出 Flow Viewer 資訊，了解所生產的產品其流程狀態正確與否。

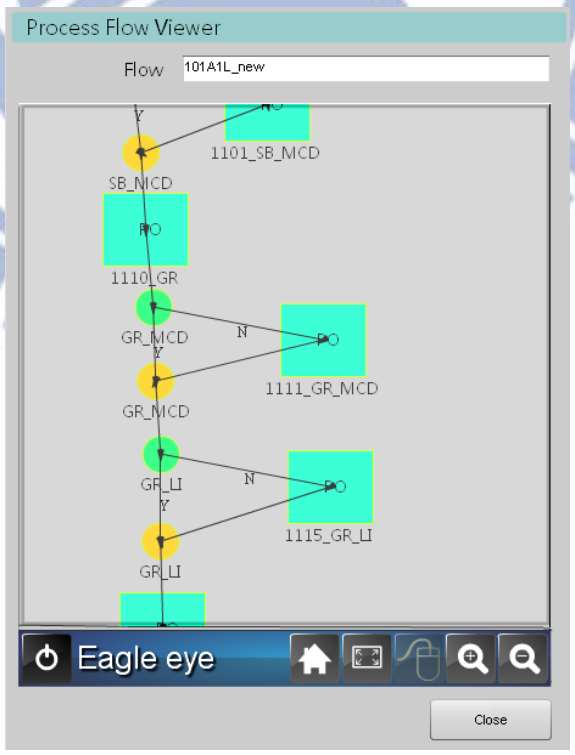


圖 27、Flow 站點流程圖

當 Lot 經由 Lot Start 確認生產後，之後透過 TrackIn 功能，其代表已將此批在製品送到了該站點，並開始該站點的製程。

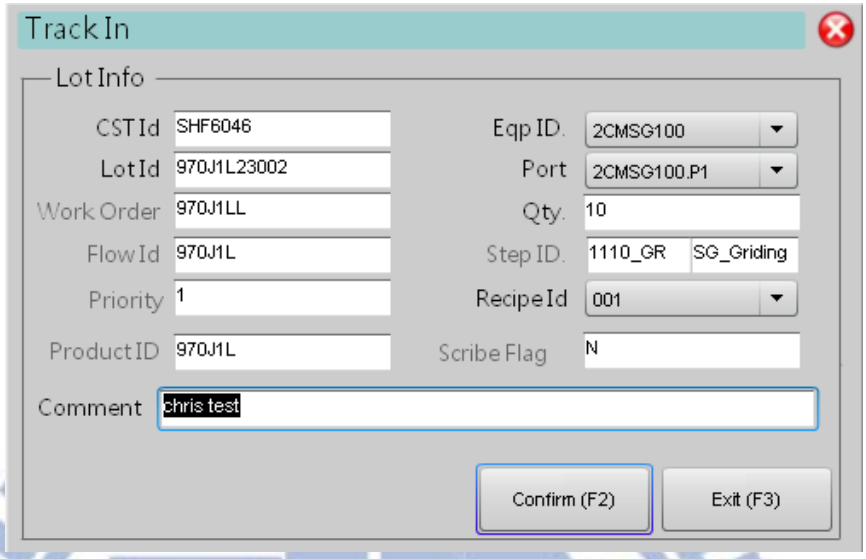


圖 28、Lot TrackIn

於該站點完成製程後，經由 TrackOut 功能，上傳生產過程中的部份參數並告知系統，目前以完成該製程，並將在製品送到下一個站點準備下一道製程。

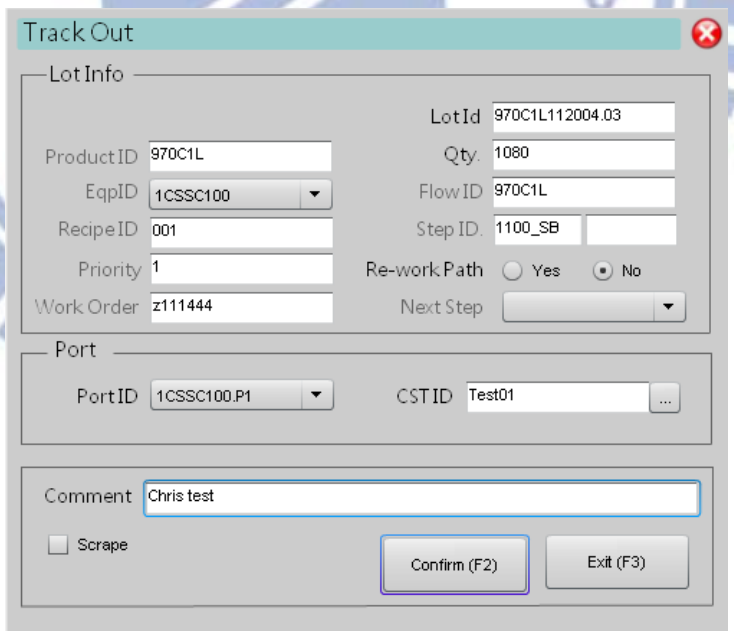


圖 29、Lot TrackOut

在生產製造的過程當中，勢必也會遇到須要將在製品分併批的狀況發生，在此時便可使用 Lot Split 或 Lot Compose 兩個功能來協助完成帳籍上的資料調整。

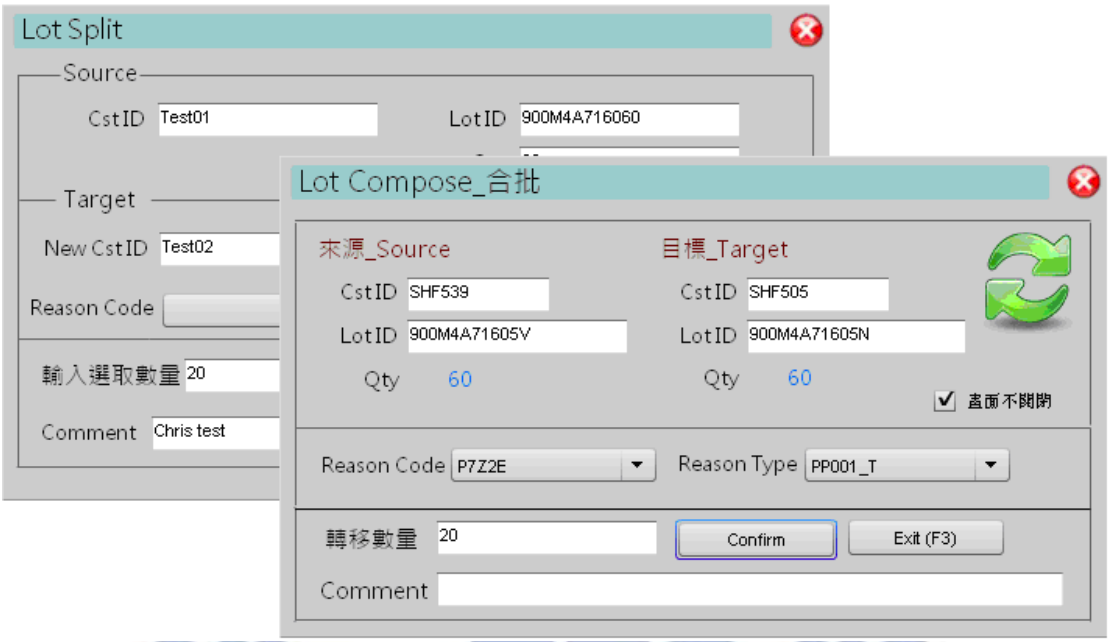


圖 30、Lot Split & Compose

在製品經過了數道製程後，會於最後一個站點裝箱打包，此時則由 Packing & Trackout 功能來進行帳籍上的調整以及將數量相當的在製品帳籍和箱號互相連結。

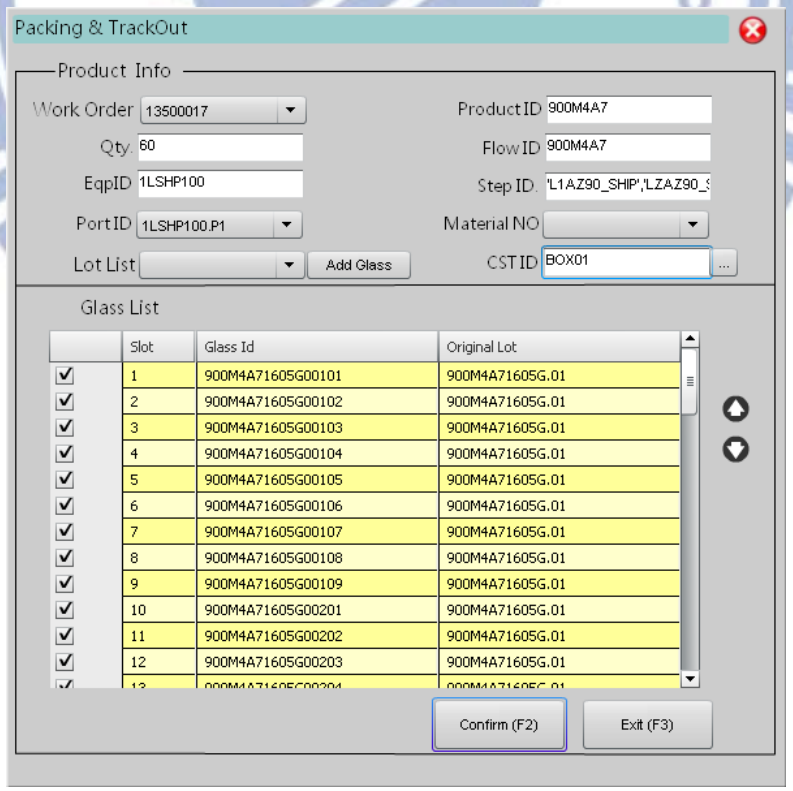


圖 31、Packing & TrackOut

於裝箱打包後，統一經由 Ship 功能，計算相關的帳籍資料，並將這些帳籍資料，透過 Application Server，轉發至後端的倉儲管理系統 (Warehouse Management System , WMS)。

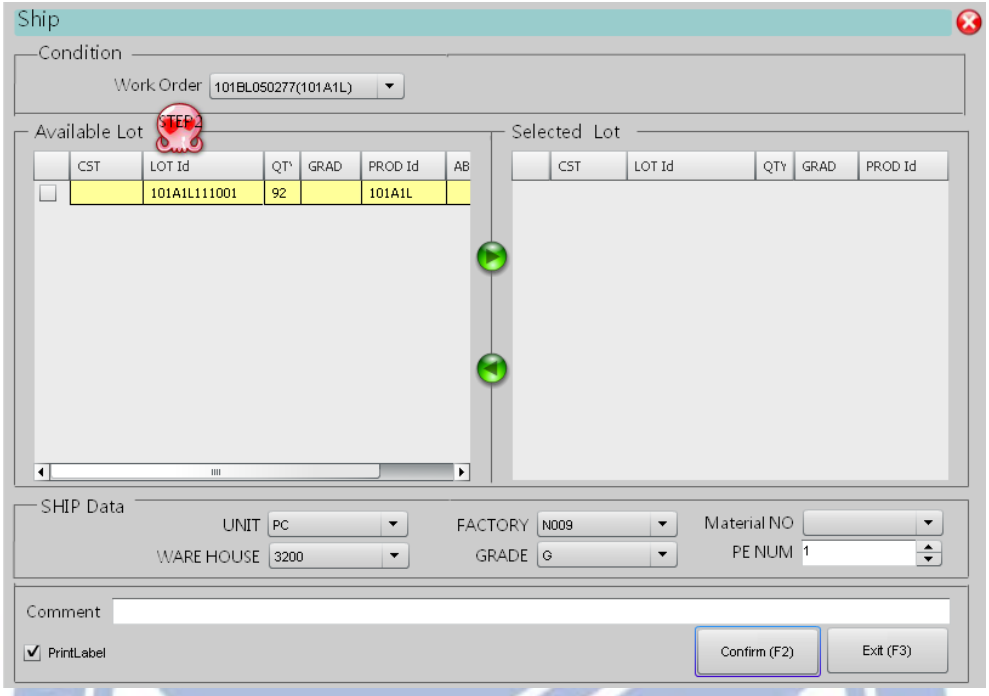


圖 32、Ship

在生產的過程中，一定會面臨到需要查詢各項在製品目前狀態的情形發生，此時系統上提供了 Lot information 功能查詢，可瞭解目前的產品各項參數資訊。系統亦同時提供了歷史資料查詢的功能，如此可詳細了解到特定在製品的各項生產狀況，另一方面也方便找尋問題。

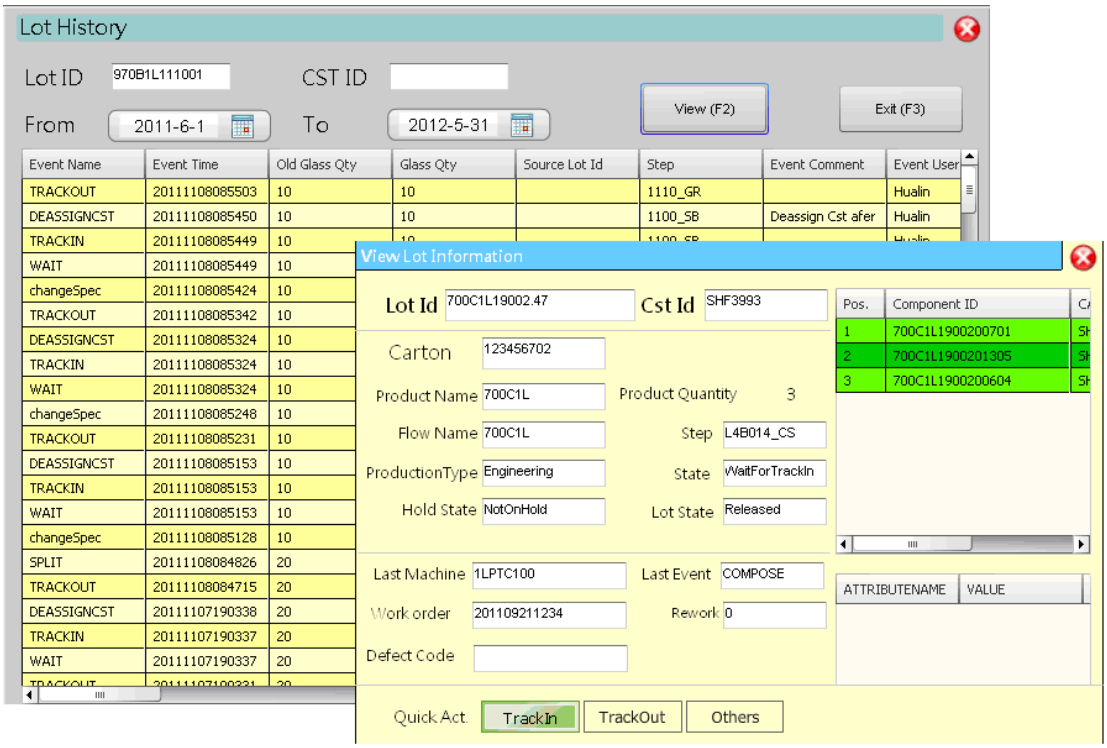


圖 33、Lot Information & Lot History

## 5.2.4 主動訊息廣播程式 & 主動訊息接收及周邊硬體控制程式

主動訊息廣播程式，其程式的執行環境和 Application server 是相同的，此程式主要由 C#來開發，並以 TIBCO/RV 通訊方式接收來自 Host server 的資訊後，接著再以轉送給有在程式中註冊過的接收程式。

而主動訊息接收及周邊硬體控制程式則是執行在 Client 端的電腦上，其亦為 C#為開發語言經由 WCF 的通訊方式，接收來自主動訊息廣播程式的訊息，並和 Client UI 溝通。

在圖 34 中，畫面下方的資訊則為 Host Server 所送出來的機台資訊，右方則是 Lot 相關的資訊。

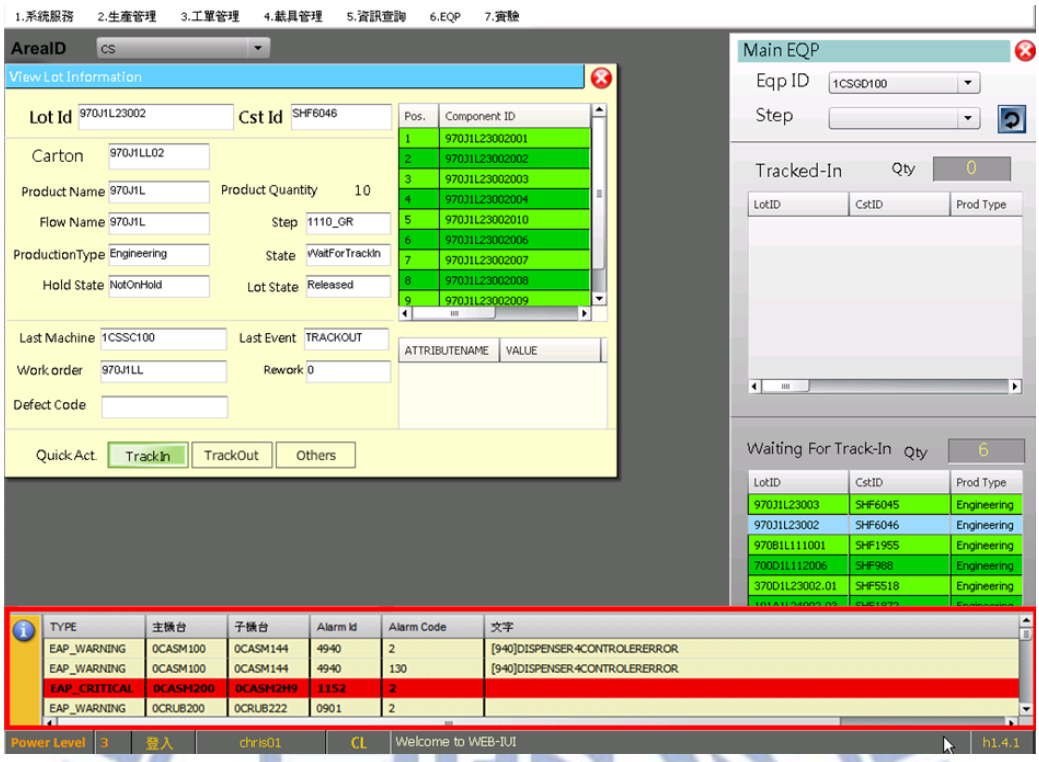


圖 34、主動式通知訊息

而在控制硬體方面，最基本的功能則是列印出貨時的 Ship Label，因此圖 35 中所顯示即為透過周邊硬體控制程式來列印相關的資料。

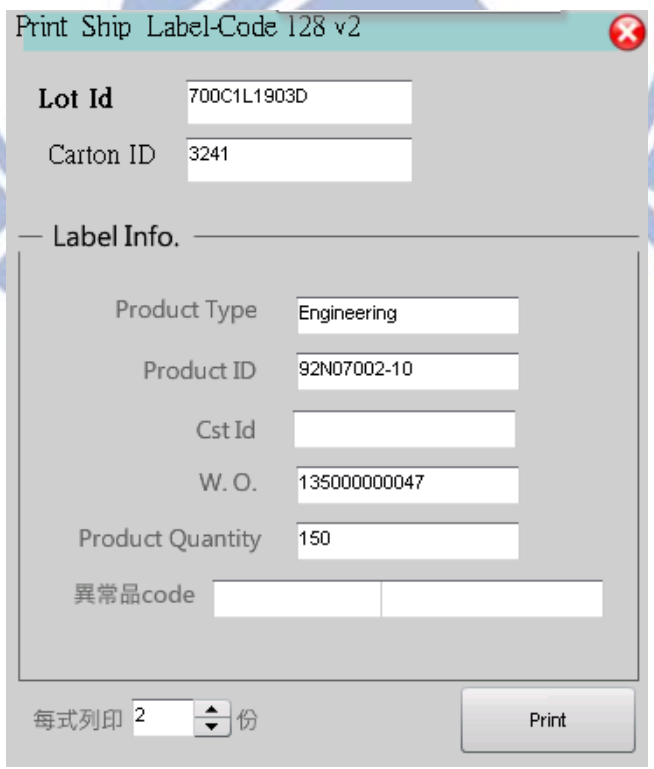


圖 35、列印標籤

## 5.3 系統評估

一個良好的現場即時過帳系統，是可以令製造商中的各個層級的了解產品的各種狀況，隨時可依據各項資訊做出決策，因此提供一個操作流程完整、穩定運行的系統，勢必更加能夠獲得使用者的喜愛。

對於本研究的現場即時過帳系統方面，主要針對以下三點進行本系統的評估。

- 動態調整系統資源

  本研究中，Application Server 的設計目的之一，就是為了當 MES Host Server 無法正常運作，藉由 Application server 的運作機制，可自行切換備援的 Host Server；亦或當 Application server 效能不彰的情況發生時，可隨時增加 Application server 程式的運行數量，以此可對於系統的資源做出有效的調整。

- 標準化介面的通訊方式

  透過使用 Web Service 的通訊方式，Application server 可以接收來自各系統間的請求，透過此種方式，CIM 中的每一個系統間，可以減少開發各自交握訊息的方式及時間，降低開發程式資源上的人力浪費。

- 元件化的子功能開發方式

  因應不同的產品有著許多不同的生產方式，自然而然的情況下，一定會面臨到需要使用不同邏輯的作帳系統功能，因此不斷的開發或修改功能在即時過帳系統上是不可避免的狀況。

  本研究中 Application Server 和 Client UI 接使用元件化的開發特性，皆可以使得開發人員透過開發修改不同的元件，可快速的針對需求做出回應，大幅減少開發時程。

- 互動性高的使用者介面

  使用 Flash 所開發出來的 Client UI，因 Flash 本身提供了大量的元件可使用，因此可呈現許多舊有系統所無法帶出的資訊；和使用者的互動上，也可設計許多的提醒互動功能，以方便使用者能夠正確的操作本系統。

## 5.4 系統比較

本研究的目的，主要是為了設計出一套於建廠時，可取代目前廠內所運行的系統，因此在本節中，將針對本研究以及廠內目前系統做出比較。

表 1、系統比較表

| | Factory works | 本研究 |
|---|---|---|
| 軟硬體建置維運成本 | 高 | 低 |
| 軟體版本更新 | 須完整編譯程式，開發效率慢 | 只須針對特定需求更新 |
| 備援切換 | 須完全停下系統、再行切換備援 | 透過Application Server備援切換機制，可快速切換 |
| 各系統整合 | 須依照不同系統，開發溝通方式 | 經由Web Service，以Application Server爲溝通的橋樑 |
| 使用者介面 | 無法完整呈現所有資訊 | 互動性強<br>圖型化呈現生產資訊 |

針對表格中的內容，以下將說明所代表意義。

- 軟硬體建置維運成本

  Factory works 為向廠商所購買的產品，其在每次建廠的過程中，必需重複購買，且除了在建置的過程中，需要大量的費用，每年亦須簽訂維護合約，長期下來是一筆不小的金額開銷。而本研究中，除了建置過程中的硬體成本不需要使用到非常高階的機器外，亦可節省下大量的購買維護合約。

- 軟體版本更新

  Factory works 在更新程式的時，需要將整個應用程式重新編譯，且 Client 端亦需完整更新程式；而本研究只需針對特定的功能做出編譯，Client 端亦只需要下載特定功能即可。

- 備援切換

  本研究中的 Application Server，其會在後端 Host Server 無法運作的同時，快速的切換到備援上，而透過同時執行複數支的 Application Server，就算任何一支 Application Server 程式失效，前端送上來的需求也可被其他 Application Server 所執行。

- 各系統整合

本研究中可經由 Application server 做為溝通各系統間的橋樑，而 Factory works 則需配合不同系統開發特定的通訊方式。

◆ 使用者介面

Factory works 所提供的使用者介面較為傳統，無法有效呈現許多的生產資訊；本研究除了可順利呈現外，亦可和使用者產生良好互動。

# 第六章 結論與未來研究方向

現場即時資訊系統對於製造業而言，是一項重要的資訊掌控系統，其可以有效的協助工廠生產。本研究中為了使得此系統能夠在開發及修改需求的過程上，更加的便利，而使用者在使用的過程中，能夠有良好的互動、忠實的呈現所有的生產資訊，而在新建廠的過程當中，期望透過這套系統可以大幅的減少初期投入成本以及後期的維運費用。

## 6.1 結論

現場即時過帳系統，在工廠的生產過程中，可以比喻成是工廠的眼睛，透過正確的使用本系統，可以令工廠的管理輕鬆不少，也可以有效的了解目前的產品狀態。以下將簡述本研究中，所達成的幾項目標：

- 大幅縮減系統建置維運成本，除了可使用較為低階的硬體配備外，亦可減少維護的合約經費。
- 三層式架構的邏輯，可使得系統彈性大幅度提高。在備援的切換上，可使得使用者不會感受到系統有發生切換的停滯時間。
- 透過 Application Server 可有效溝通其他 CIM 子系統，如此一來對於系統的整合提供強大的便利性。
- 元件化的開發方式，可以使得系統在開發的過程中，減少許多開發時間，同時許多元件可重複利用。
- 友善的使用者介面使得系統在使用上更為簡易、方便。

## 6.2 未來研究

本研究中，對於廠內的生產使用狀況，目前尚有許多不足之處可加以補強，以下將針對其中的幾項做出闡述。

- 整合報表系統相關功能

  目前本系統和報表系統是互相獨立的兩套系統，因此在使用上，便須同時開啟兩套的程式，造成使用上的不便，因此在未來整合報表系統將成為必然的趨勢。

- 結合行動裝置

在廠內的生產過程中，現場的同仁勢必會到處走動，而目前使用本系統的Client UI，只能在定點的電腦上執行，造成使用者可能需要來回奔波於特定位置間。為了解決此問題，配合行動裝置所使用的 Client UI 程式將可有效降低使用者的不便。

# 參考文獻


[1].Keynote, “Rich Internet Applications: Design,Measurement, and Management Challenges”, White Paper,2006

[2].Qi Yu , Xumin Liu , Athman Bouguettaya , Brahim Medjahed,” Deploying and managing Web services issues, solutions, and directions”, THE VLDB JOURNAL,2005

[3].MESA International, “MES Explained: A High Level Vision”September, - White Paper Number 6 ,1997

[4].Reinhard Füricht, Herbert Prähofer, Thomas Hofinger,and Josef Altmann , “A component-based application framework for manufacturing execution systems in C# and .NET “, CRPIT ’02 Proceedings of the Fortieth International Conference on Tools Pacific: Objects for internet, mobile and embedded applications,2002

[5].Huang-yu Lai, "A Service-Oriented Architecture Based Platform to Integrate Information System for Semiconductor Manufacturing", Master degree thesis,2006

[6].Harvey I. Cohen, James V. McCusker, "Web access for a manufacturing execution system", United States Patent number: 5847957 ,1998

[7].關宇，” 整合 HL7/XML 與 XSL 對電子病歷之轉換”，國立成功大學工程科學研究所碩士論文，民國 91 年

[8].郭乃榮，” 支援資料描述及可攜性文件集合之 XML 資料儲存系統”，國立成功大學電機工程學系碩士論文，民國 91 年

[9].張亞飛,” Flash Flex ActionScript 3.0 開發權威手冊”,2011

[10]. 網頁新視覺享受-RIA, Available at http://www.cc.ntu.edu.tw/chinese/epaper/0002/20070920_2008.htm

[11]. WCF概觀說明, Available at http://msdn.microsoft.com/zh-tw/library/ms731082.aspx

[12]. WCF概要, Available at https://sites.google.com/site/stevenattw/dot-net/wcf/wcf-essentials

[13]. Message Queue, Available at http://msdn.microsoft.com/en-us/library/ms711472%28VS.85%29.aspx

[14]. 將JAVA應用程式移至.Net, Available at http://60.251.1.52/taiwan/msdn/library/2002/Aug-2002/article/dotnet_MovingJavaApps.htm