國立交通大學

電機資訊學院 資訊學程

碩士論文

使用視覺化的軟體建構方法來產生網站應用程式

A Visual Software Construction Approach for Generating Web-based Application System

研究生：張筱楓

指導教授：陳登吉　教授

中華民國九十三年六月

使用視覺化的軟體建構方法來產生網站應用程式
A Visual Software Construction Approach for Generating Web-based Application System

研 究 生：張筱楓　　Student：Shiau-Feng Chang

指導教授：陳登吉　　Advisor：Dr. Deng-Jyi Chen

國 立 交 通 大 學

電機資訊學院 資訊學程

碩 士 論 文

A Thesis

Submitted to Degree Program of Electrical Engineering Computer Science

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

授權書

(博碩士論文)

本授權書所授權之論文為本人在__國立交通__大學(學院)__電機資訊學院__系所

__資訊____組____92____學年度第__二__學期取得__碩__士學位之論文。

論文名稱：____使用視覺化的軟體建構方法來產生網站應用程式__

1.□同意　　□不同意

本人具有著作財產權之論文全文資料，授予行政院國家科學委員會科學技術資料中心、國家圖書館及本人畢業學校圖書館，得不限地域、時間與次數以微縮、光碟或數位化等各種方式重製後散布發行或上載網路。

本論文為本人向經濟部智慧財產局申請專利的附件之一，請將全文資料延後兩年後再公開。(請註明文號:　　)

2.□同意　　□不同意

本人具有著作財產權之論文全文資料，授予教育部指定送繳之圖書館及本人畢業學校圖書館，為學術研究之目的以各種方法重製，或為上述目的再授權他人以各種方法重製，不限地域與時間，惟每人以一份為限。

上述授權內容均無須訂立讓與及授權契約書。依本授權之發行權為非專屬性發行權利。依本授權所為之收錄、重製、發行及學術研發利用均為無償。上述同意與不同意之欄位若未鉤選，本人同意視同授權。

指導教授姓名：

研究生簽名：　　學號：9067581

（親筆正楷）　　（務必填寫）

日期：民國　93 年　7　月　27　日

本授權書請以黑筆撰寫並影印裝訂於書名頁之次頁。

授權第一項者，所繳的論文本將由註冊組彙總寄交國科會科學技術資料中心。

本授權書已於民國 85 年 4 月 10 日送請內政部著作權委員會（現為經濟部智慧財產局）修正定稿。

本案依據教育部國家圖書館 85.4.19 台(85)圖編字第 712 號函辦理。

使用視覺化的軟體建構方法來產生網站應用程式

學生：張筱楓　　指導教授：陳登吉 教授

國立交通大學電機資訊學院 資訊學程（研究所）碩士班

摘　　要

網際網路的普及加快了網站應用的發展，網站應用程式的跨平台特性，更讓其成爲新一代應用程式的開發平台，各種網站應用程式語言以及開發工具也因應而生。

目前視覺化應用程式開發工具已經相當普及，而且確實可以有效的減少開發所需時間，增加開發效率，但多數網站應用程式開發系統，卻沒有完整的視覺化開發系統，可以輔助開發，多數傳統的網站應用程式視覺化開發系統專注於畫面的設計部份，沒有針對網站可重用元件作視覺化開發之應用，更沒有辦法針對網站應用程式流程作完整的規劃控管設計。

本研究設計一視覺化開發網站應用程式之環境，提供根據 MVC 設計樣式設計之伺服器端應用程式框架、符合 XForms 規格之網頁表單，加上網站應用程式產生器，自動產生最後所需之網站應用程式。

市面上此類產品真正完全符合開發需求的很少，對於應用程式開發的生產力提昇有限，本研究透過結合 MVC 設計樣式、XForms、XSLT、Code Generation 等技術實作網站應用程式之視覺化開發系統，不只提供視覺化之網頁元件，更提供視覺化之網站應用元件，讓完整網站應用程式之框架自動產生，讓開發網站應用程式所需之時間縮短，開發效率提高。

A Visual Software Construction Approach for Generating Web-based Application System

student：Shiau-Feng Chang Advisors：Prof. Deng-Jyi Chen

Degree Program of Electrical Engineering Computer Science

National Chiao Tung University

ABSTRACT

The increase on the population of Internet accelerates the development of web-based applications. The ability to cross-platform makes web applications a new generation of developing platform. As such, a variety of web-developing programming language and developing tools are invented.

Nowadays, visualized computer application developing tools are widely used which indeed are able to reduce time and effort cost to the programmers. However, web applications are lacking of such completely visualized tools to assist web developing. Most of the traditional visualized webs developing tools are concentrated on developing web pages rather than web applications. They have neither reusable application modules nor visualized tools to develop these modules. Further, traditional visualized web developing tools are not capable of doing project management and flow control for web-based applications.

Our research design is to implement a completely visualized environment to develop web applications. Its developing tool and code generator are able to generate final web applications automatically by providing MVC server-side application framework, web forms conform to XForms.

There are few products in the market that could meet the requirements of developing web applications. And thus these products do little to increase productivity of developers. Our research consolidates MVC design pattern, XForms, XSLT and Code Generator to build up a completely visualized web application developing system. In addition to visualized web page modules, it provides web application modules as well. Moreover, it can automatically generate the framework of web applications. As a result, our web developing system would be able to reduce time and increase efficiency to develop a web application.

誌　謝

本論文承蒙恩師陳登吉教授耐心的指導及教誨，得以順利完成。在研究生活的三年中，陳教授不僅在學術上給予指導，在研究態度亦受教良多。在此對老師獻上無限的感謝。

感謝家人們在這段日子給予的支持與付出，讓我無後顧之憂地進行研究工作，還有實驗室同學、朋友們的陪伴、打氣與協助，親情與友情是我完成這份論文最大的動力。

目　　錄

表目錄

圖目錄

一、 緒論

1.1網站應用程式簡介

1.1.1 網站應用程式 Web Application

近年網站應用程式的發展、規模、複雜度及成長皆非常的快速。網站應用程式不再只是 HTML 檔案的單向資訊呈現，相反的，許多複雜的功能已經被整合到網站並且普受企業的使用。

應用程式服務提供業者(Application Service Provider / ASP)，變成新興行業，因爲網站應用程式的特性，讓使用者端只需準備網頁瀏覽器，然後連上業者提供應用程式網址即可使用。

服務提供者可以充份集中式管理應用程式版本，服務使用者則不需要另外的安裝程式，這些都是傳統主從架構(Client & Server)應用程式所沒有的優勢。圖 1 爲網站應用程式之主從架構圖：

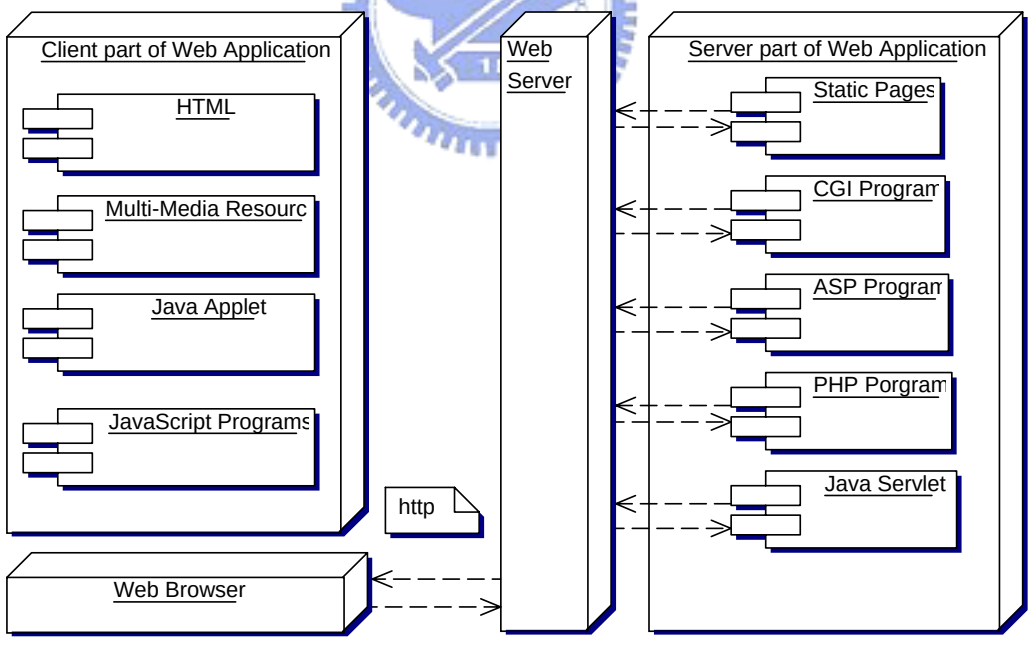

圖 1 網站應用程式之主從架構

資料來源：CoOWA: A Component Oriented Web Application Model, Technology of Object-Oriented Languages and Systems [1]

1.1.2 網站應用程式特性

有網站應用程式特性探討的文獻很多，在[1]中將這些特點大致歸納爲：

(1) Platform independence

如圖 1，瀏覽器在網站應用程式中扮演的角色是架構在作業系統之上，所有網頁也只能透過瀏覽器來存取，因此我們也可以把瀏覽器視爲一個在作業系統之上的虛擬機器(Virtual Machine)。

目前幾乎所有的作業系統都有瀏覽器用來讀取網頁、網站資料，所以等於提供了一個絕佳的標準平台，用來建構應用程式。只要能夠連上網際網路，就可以存取網站應用程式，平台獨立性是應用程式的使用接受度及推廣的基礎。

(2) On the fly，on the need deployment

網頁是網站應用程式的一個部署單元(deployment unit)，當使用者提出需求才即時產生並部署到使用者的瀏覽器，而一個完整的網站應用程式是由這樣一連串的即時產生及部署連接起來的，這是其跟傳統 client-server 應用程式最大的不同點。

在傳統的 client-server 分散式架構，應用程式是在執行前就要在 client 端及 server 端，皆部署完成才能使用。

此種特性讓網站應用程式的分散式架構更能有彈性的集中式管理，因爲省下了 client 端之部署，在版本更新、管理上都具有更大的優勢。

(3) Event-driven client and procedural server

網頁是網站應用程式的使用者介面。Client 端使用 HTML、DHTML、Java applet、Flash 等所構築出來的使用者介面，藉由事件驅動(Event-driven)來觸發伺服器端的應用程式運作。伺服端的應用程式運作則是循序式的、被動式的，由使用者不斷的觸發來執行應用程式。

1.1.3 網路應用程式開發語言

目前網路應用程式開發語言在伺服端以：Java、PHP、ASP 爲主。而在瀏覽器端的程式則以：JavaScript、Java Applet 及微軟之 ActiveX 爲主。

表 1 PHP/Java/ASP 比較表

	PHP	Java	ASP/ASP.Net
免費	是	部份 (Application Server 昂貴)	否
Application Server	無完整方案 (需自行開發)	有(J2EE)	有
物件導向語言	是	是	是
3rd Party支援	普通	多	多
開放原始碼	是	否	否
跨平台	是	是	否
學習難度	易	較難	易

1. Java

Java 在伺服器採用語言主要有 JSP 及 Java Servlet，提供之應用元件多，解決方案完整，但較完整的商業解決方案如應用伺服器(Application Server)相當昂貴(如：IBM Web Sphere、BEA Weblogic …等)。

2. PHP

PHP("PHP: Hypertext Preprocessor")為目前最普遍的網站應用程式語言，採用的語法混合 C、Perl 等，使用方式簡單，可以在 HTML 語法中直接插入所需要執行的程式。

3. ASP

微軟的 IIS 網頁伺服器直接支援之 Server Side Script，可用 VB Script 及跟 Java 類似的 J Script，一樣可以在 HTML 語法中插入撰寫程式。

1.1.4 現有網路應用程式開發系統的問題

在第二章我們會針對市面上網路應用程式之開發系統，作詳細的比較與分析，現在將現有網路應用程式開發系統的主要問題條列如下：

1. 未提供網站元件，僅提供網頁元件
2. 未提供伺服端流程架構，造成網站程式管理、修改彈性差
3. 程式碼與網頁 HTML 混雜，程式與 UI 設計人員必需共同維護檔案
4. 未提供資料庫抽象層，同時連結多種不同資料庫有困難

我們歸納以上困難，理想的視覺化網站應用程式開發工具，應該要

具有下列特點：

1. 視覺化的畫面編輯器
2. 提供商業邏輯元件庫(Business Logic)的視覺化使用
3. 提供統一的資料庫存取抽象層介面

有視覺化畫面編輯可以加快使用者介面的設計，網站架構也採用視覺化呈現流程編輯，再加上方便管理開發者的各式元件，對網站應用程式的開發才有真正增加生產力的效用。

1.2 研究動機、目標

從事網站應用程式開發工作多年，實作應用程式的開發模式也不斷的演進：

1. 程式直接內嵌在一般網頁
2. 使用樣版系統(Template)，將程式與網頁分離
3. 使用 MVC 設計樣式，將商業邏輯(Business Logic)再與 Controller、View 分離。

每次的演進，都讓程式的可重用性(Reusable)、可擴充性(Extendable)、可維護性(Maintainable)向上提昇。

但從數萬行的程式碼當中可以發現，多數的程式碼都有共同的特徵、性質，但我們在下次的開發當中無可避免的又要重複類似的工作，所以最後採用的 MVC 設計樣式，可以把 Model 抽離，例如：討論區、留言版…等，我們第一次將整個討論區的 Model 建構起來，以後有類似的需求時，可能只要作少許的修改就可以重複使用。

此外網站應用程式的開發通常會遇到以下問題：

1. 網站應用程式通常要求能快速上線服務，故開發時間需要縮短
2. 程式設計人員需根據需求使用 UI 設計人員設計之網頁(使用者介面)開發，非常耗費時間

雖然在實作上的架構逐漸的改善，但一般的程式編輯器並無法滿足需求，而部份像 Dreamweaver 等，具有視覺化開發的工具，實際上並不如想像中的好用，因爲這類工具原來就是用來設計網頁爲主，對於設計樣式的支援及提供的彈性都不足夠，也沒有提供 Models(商業邏輯)元件庫的管理。本研究之目標爲：

1. 找尋快速的方法建構網站應用程式
2. 建構常用網站模組元件庫提高網站應用程式開發效率

3. 使用樣版系統讓程式設計人員與 UI 設計人員分別維護個別之檔案

1.3視覺化軟體建構

一般使用者操作軟體時，最先影響其使用的就是使用者介面，程式設計師使用開發系統撰寫程式碼的開發過程中，使用者介面的好用與否對程式開發的效率有決定性的影響。

傳統程式開發工具多以純文字編輯器撰寫程式碼，並且使用編譯器，產生程式，因爲早期電腦程式也多是文字介面，在使用者介面上的發展並不發達。而自微軟發行 Window 系統後，圖形使用者介面(GUI)已成爲軟體介面主流，接著微軟推出 Visual Basic，開發環境也開始邁向視覺化的時代，如微軟之 Visual Studio 系列，Borland Delphi、C++ Builder 等，已成爲程式開發之主流系統。

目前在視覺化開發系統已相當成熟，也的確提升不少應用程式開發的速度和品質，因爲能夠減少使用者介面設計所花費的時間，而且可以馬上看到執行時使用者所看到的介面。

以視覺化方式開發網站應用程式之系統，也逐漸受到重視，但實際使用上的普及程度並不高，多數開發者仍然使用傳統的程式編輯器來設計程式，我們在第二章會逐一分析現有產品的概況。有 Macromedia 的 Dreamweaver，Yes Software 的 Code Charge 及 TurboPHP。

1.3.1 視覺化軟體建構方法論

視覺化程式的定義：

1. 在 Visual Object-Oriented Programming Concepts And Environments 書中有一個說法[2]：

   “Visual programming is distinguished both by what is described and how it is described. What is described are programs, expressed in terms of the syntax and semantics of the programming language.”
   「視覺化程式是被分別在本身如何被描述的和如何描述，被描述的是程式，按照程式語言的句法和語意學表達。」

2. 在世紀末軟體革命 C++，GUI 與物件導向理論[3]書中提出了一個說法：

   “在螢幕上直接設計/畫出程式的結構、流程、或物件的關係、視窗、使用者介面，並且可以將這繪製的結果直接產生可執行的程式、或

產生特定語言的程式碼，這種程式設計和寫作的方式就稱作視覺化程式設計。”

傳統撰寫程式的方法是用文字文式來撰寫，使用者利用鍵盤輸入依照自己的需求輸入符合程式語法的文字。視覺化語言提供圖形爲表達工具，使用桌面、視窗、icon 等比喻來和使用者溝通，使用者則利用滑鼠、鍵盤等設備來編輯所要表達的程式，系統會依使用者編完的內容產生一相對應的程式4。

1.3.2 視覺化軟體建構分析

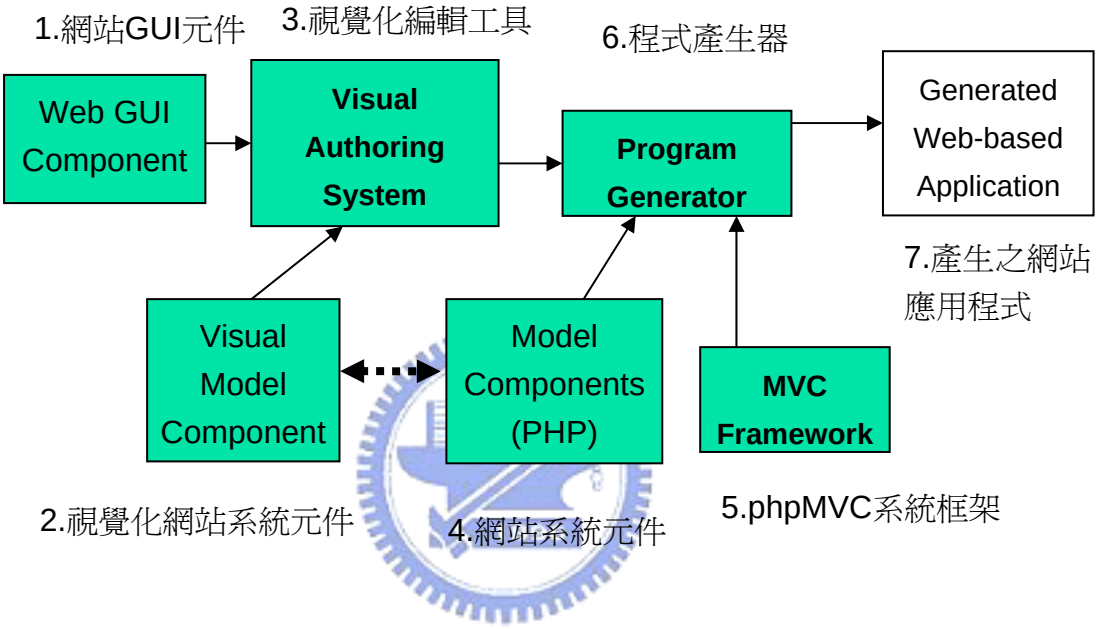

圖 2 視覺化軟體建構方法

如上圖，爲了要解決傳統網站應用程式開發的困難，我們採用了視覺化軟體建構之模型，用來建構整個視覺化網站應用程式的開發系統，其中：

1. Web GUI Component：指的是視覺化的網頁呈現相關元件，如 HTML 表單之按鈕、文字輸入表單等，這部份是多數開發軟體都具有的模組
2. Visual Model Component：是視覺化的網站模組物件，例如在本研究中實作的使用者登入系統中，有一 PHP 撰寫之 User Class 即是，依照同樣的模式，我們可以實作更多的網站模組物件，供視覺化開發系統使用，例如：討論區模組、留言板模組…等
3. Visual Authoring System：是整個視覺化開發之核心，視覺化物件之配置、設定皆在此部份進行
4. Model Components：可重用之網站模組元件，如討論區、留言板等

用伺服端程式語言撰寫之網站應用程式，爲 Visual Model Component 之實體程式碼。

5. MVC Framework：伺服器端程式架構採用 MVC design pattern 設計，在第四章中有詳細說明
6. Program Generator：程式產生器將視覺化元件作分析，加上架構好的系統框架，以及對應的 Model Component 程式碼，產生最後可用的程式碼
7. Generated Web-based Application：最後由 Program Generator 產生的即是完整的網站應用程式，包含有：呈現網頁的樣版以及對應網站運作的各程式。

二、 相關系統研究探討

2.1 Macromedia Dreamweaver MX 2004[5]

2.1.1 軟體簡介

Macromedia Dreamweaver MX 2004 為 Macromedia 所出品之網頁編輯工具，主要提供網頁美編人員，設計編排 HTML 網頁之用，但從 Dreamweaver 4 以後的版本，也開始提供程式設計人員建構網站應用程式的支援，例如：支援 ColdFusion、ASP.NET、JSP 及 PHP 等伺服端應用程式語言，並支援網站資料庫程式之開發。

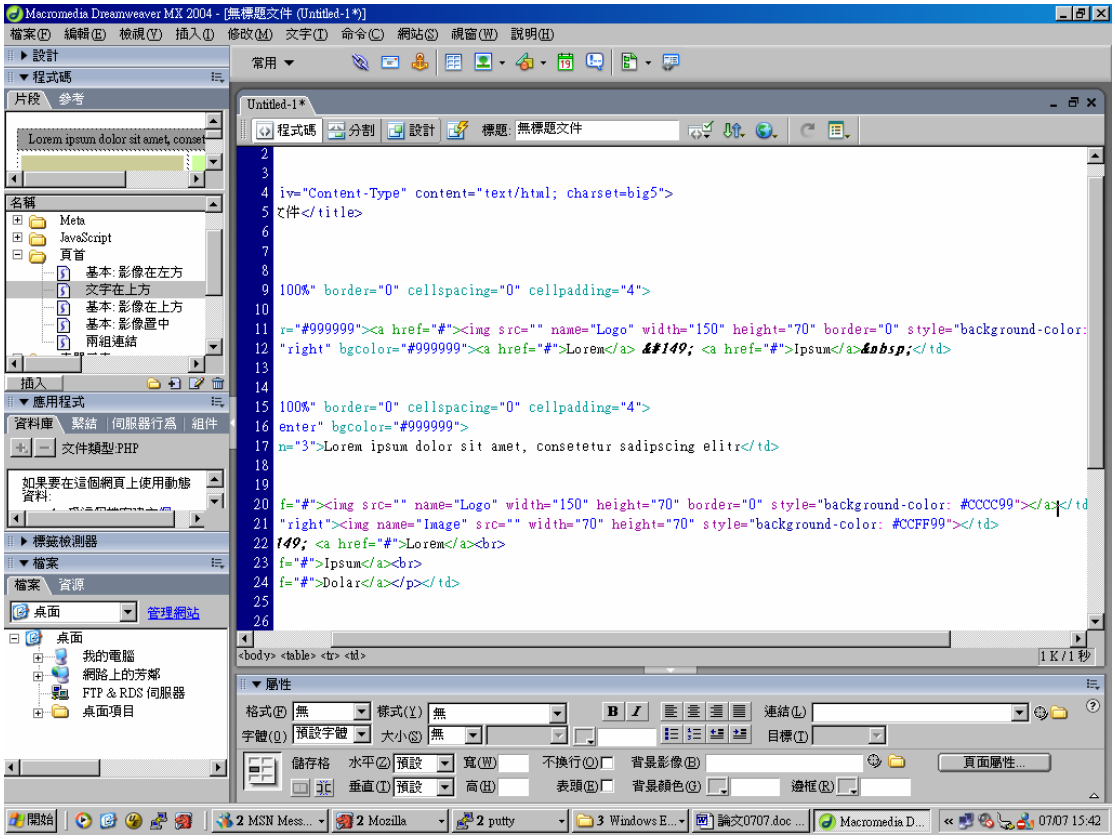

圖 3 Macromedia Dreamweaver MX 2004

2.1.2 軟體功能

Macromedia Dreamweaver MX 2004 提供下列主要功能：

1. 支援 CSS 之 HTML 編輯
2. 程式編輯支援

3. 整合 Flash
4. 外部程式碼、檔案管理
5. 內建圖片編輯器
6. 整合網站應用程式開發(ColdFusion、ASP.NET、JSP、PHP)

2.1.3 優點

Dreamweaver 用於視覺化網站應用程式開發的主要優缺點條列如下：

1. 網頁編排快速
2. 網頁跨瀏覽器相容性佳
3. 支援多種伺服器端應用程式語言

2.1.4 缺點

1. HTML 與程式碼混雜
2. 未提供資料庫抽象層(Database Abstraction Layer)API
3. 無法應用 MVC 設計樣式

2.2 Code Charge Studio[6]

2.2.1 軟體簡介

Code Charge Studio 爲 YesSoftware 之產品，Code Charge Studio 提供一整合式的開發環境(IDE)，主要用來製作以資料庫存取爲主(Database Driven)的網站應用程式 ，能夠快速地開發網站應用程式。

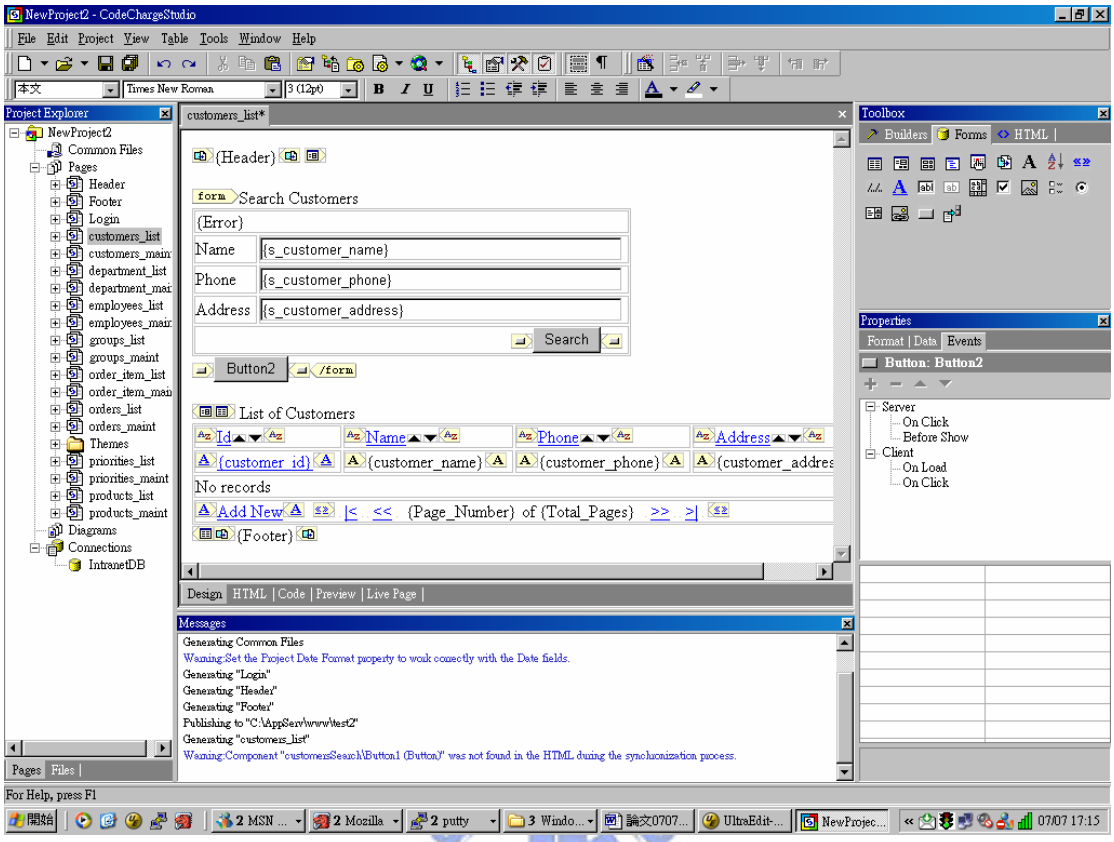

圖 4 Code Charge Studio

2.2.2 軟體功能

Code Charge Studio 爲專門開發網站應用程式之工具，因此設計上較像一般程式編輯器環境，其具有之主要功能如下：

1. 提供具 Drag & Drop 功能之圖形式開發介面
2. 支援 ASP.NET、ASP、JSP、Java Servlet、Coldfusion、PHP、Perl
3. 支援 ODBC 資料庫程式開發

2.2.3 優點

1. 具編輯畫面、編輯程式、預覽結果等工具
2. 支援 ASP.NET、ASP、JSP、Java Servlet、Coldfusion、PHP、Perl，多種伺服端網站應用程式語言
3. 具事件驅動(Event-driven)設計模式
4. 採用樣板(Template)將程式碼與 HTML 碼分離管理

2.2.4 缺點

1. 未支援資料庫抽象層(Database Abstraction Layer)，僅支援 ODBC 資料庫程式設計
2. 缺乏 Controller 部份作完整的網站應用程式流程控管功能。

2.3TurboPHP[7]

2.3.1 軟體簡介

TurboPHP 爲 Least-Resistance Software 公司推出之網站應用程式編輯工具，主要支援網站應用程式語言爲 PHP，採用視覺化元件拖拉之方式建構應用程式版面，並用事件驅動(Event-Driven)的概念，來設計網站元件所需對應之程式碼。

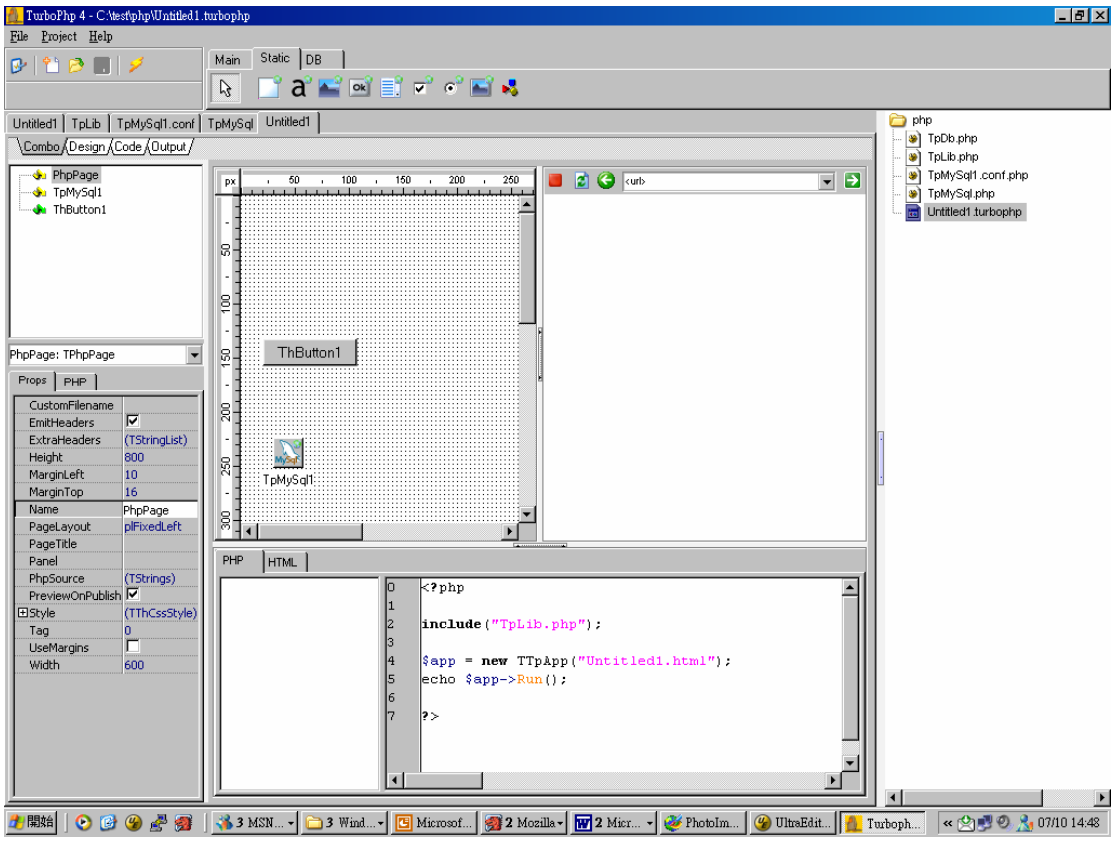

圖 5 TurboPHP

2.3.2 軟體功能

1. 視覺化拖拉元件
2. 與 Style Sheet 結合，修改版面外觀
3. 所有的視覺化元件以物件型態整合進 PHP 程式碼

4. 以物件事件驅動(Event-Driven)設計程式

2.3.3 優點

1. 具視覺化編輯畫面、預覽
2. 具事件驅動(Event-driven)設計模式
3. 採用樣版技術將 HTML 與程式碼分離

2.3.4 缺點

1. 未支援資料庫抽象層(Database Abstraction Layer)，僅支援 ODBC 資料庫程式設計
2. 缺乏 Controller 部份作完整的網站應用程式流程控管功能。

2.4現有系統之比較及限制

將上列 Dreamweaver MX 2004、Code Charge Studio、TurboPHP 加上傳統程式編輯器作比較：

表 2 網站應用程式開發軟體比較表

	Dreamweaver	Code Charge Studio	TurboPHP	PHP Editor
支援語言	PHP,JSP,ASP.NET,ColdFusion	PHP ,ASP.NET/C#, ASP/VBScript, JSP, Java Servlets, ColdFusion,perl	PHP	PHP
MVC	無	無	無	無
Template	無	有	有	無
資料庫抽象層	無	無	無	無
支援資料庫(PHP)	MySQL	MySQL, Oracle, DB2…, (ODBC)	MySQL	無
視覺化網頁元件	有	有	有	無
視覺化網站系統元件	無	無	無	無

由上列比較我們可以看到現有系統之限制：

1. 皆未提供視覺化網站模組元件，僅部份提供視覺化網頁元件
2. 未提供伺服端流程架構(MVC)，造成網站程式管理、修改彈性差

3. 程式碼與網頁 HTML 混雜，程式與 UI 設計人員必需共同維護檔案
4. 未提供資料庫抽象層，同時連結多種不同資料庫有困難

在以下的第三、四章我們提出完整的構想設計與實作，嚐試解決這些限制。

三、 以視覺化軟體開發網站應用程式系統分析

3.1網站應用程式開發流程

在本節我們分別分析網站應用程式之傳統開發方式以及採用視覺化建構方式之開發流程。

3.1.1 傳統開發方式

網站應用程式傳統開發方式如下圖所示：

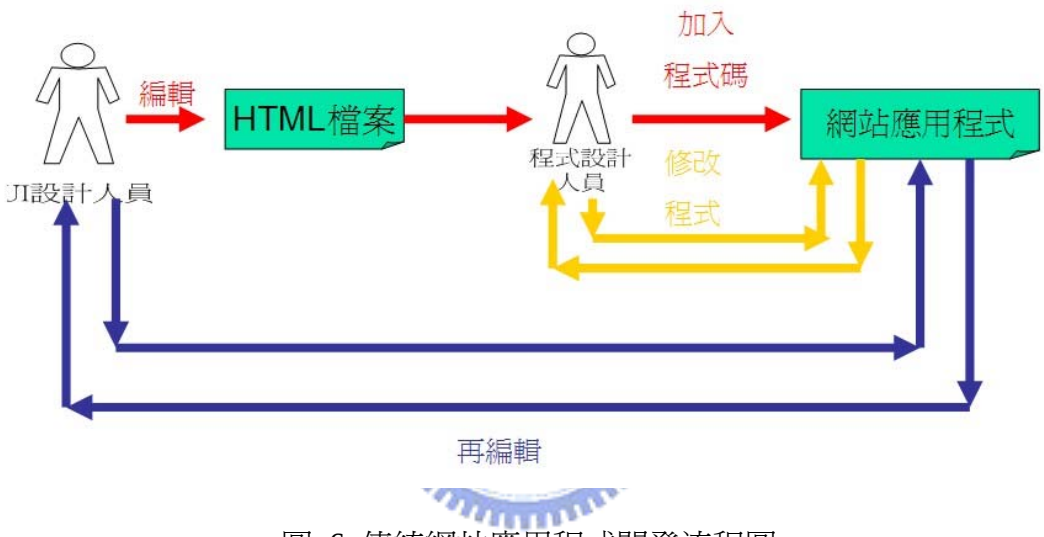

圖 6 傳統網站應用程式開發流程圖

由上圖可以清楚的看到，傳統之網站應用程式開發，多是根據需求由 UI 設計人員先製作 HTML 檔案，即瀏覽器呈現之畫面，再由程式設計人員修改後，產生網站應用程式。

但是一個網站的畫面是需要經常的修改更新的，此時 UI 設計人員直接用網頁編輯軟體修改含有程式碼的檔案，是非常危險的，可能在修改的當中造成網站的運作問題。

3.1.2 使用視覺化軟體建構方法來開發網站應用程式之流程

使用視覺化軟體建構方法來開發網站應用程式之流程，如圖 7 所示：

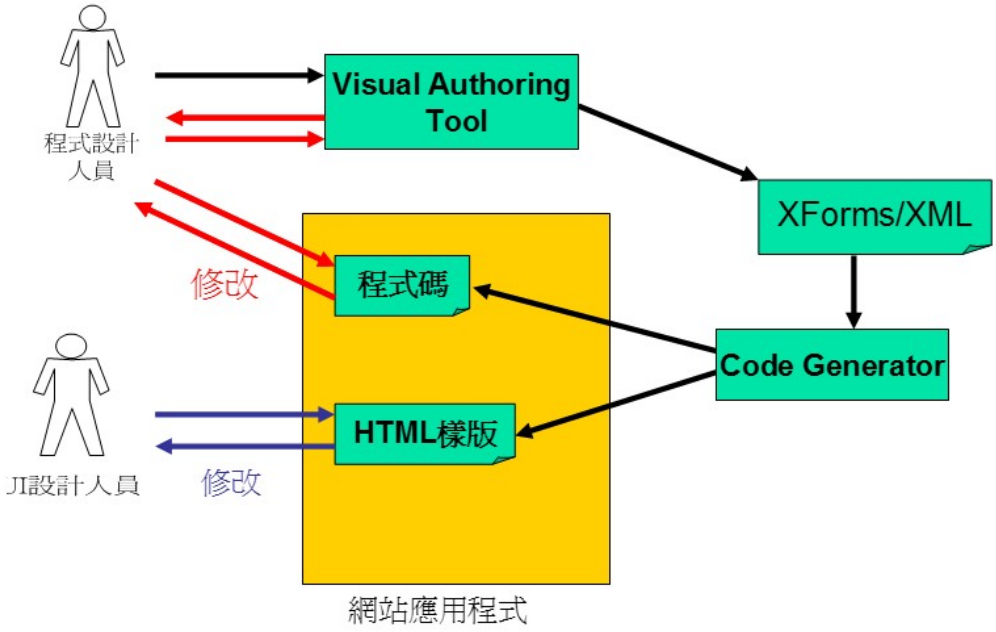

圖 7 使用視覺化軟體建構方法開發網站應用程式之流程

上圖爲使用視覺化網站應用程式產生器之開發流程，程式設計人員先用視覺化編輯工具建構程式以及最基本之軟體介面，再整合程式碼產生器後，產生最後的網站應用程式。

網站應用程式包含兩部份，一爲程式碼，可由程式設計人員修改，二爲 HTML 樣版可由 UI 設計人員處理，二者不會有共同維護相同檔案的困擾。

3.1.3 系統架構說明

如下圖說明整個開發系統之架構，在 Client 端之開發環境有：

Visual Authoring Tool：以視覺化之元件建構網站之介面

Target Code Generator：又分爲產生網頁畫面的 UI Generator 及產生程式碼之 PHP Generator

在伺服端之爲：

1. Server File System：存放開發系統所產生之網頁畫面+程式碼及伺服端流程控制之 XML 設定檔(phpmvc-config.xml)
2. PHP Engine：爲執行 PHP 程式之直譯器
3. Server Module：爲搭配執行 PHP 程式之網頁伺服器，例如：Apache 或微軟之 Internet Information Server

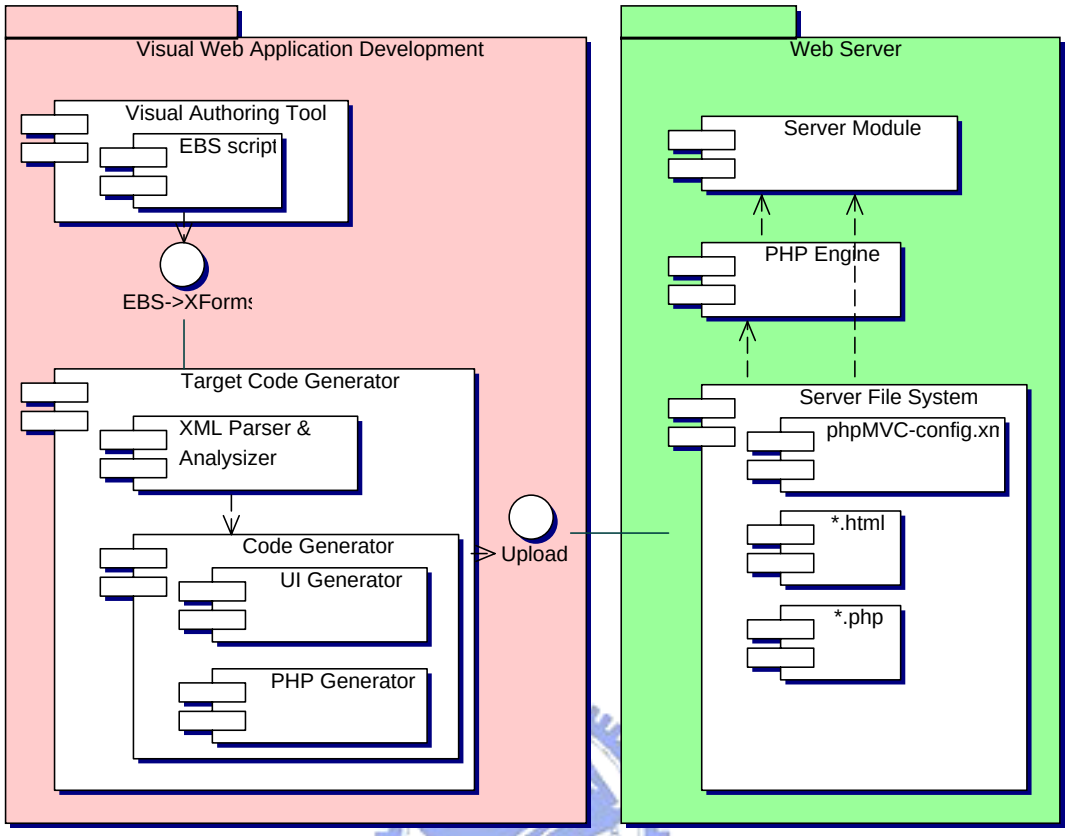

圖 8 視覺化網站應用程式開發系統架構

四個模組的詳細功能，以下分別說明，並舉例之：

1. 視覺化編輯模組(Visual Authoring Tool)：

以編輯手之 Drag&Drop UI 編輯網頁基本畫面並設定各元件對應之事件及屬性，最後以 EBS 儲存，其中元件包含 HTML 畫面元件、網站模組元件等，如圖 9 所示：

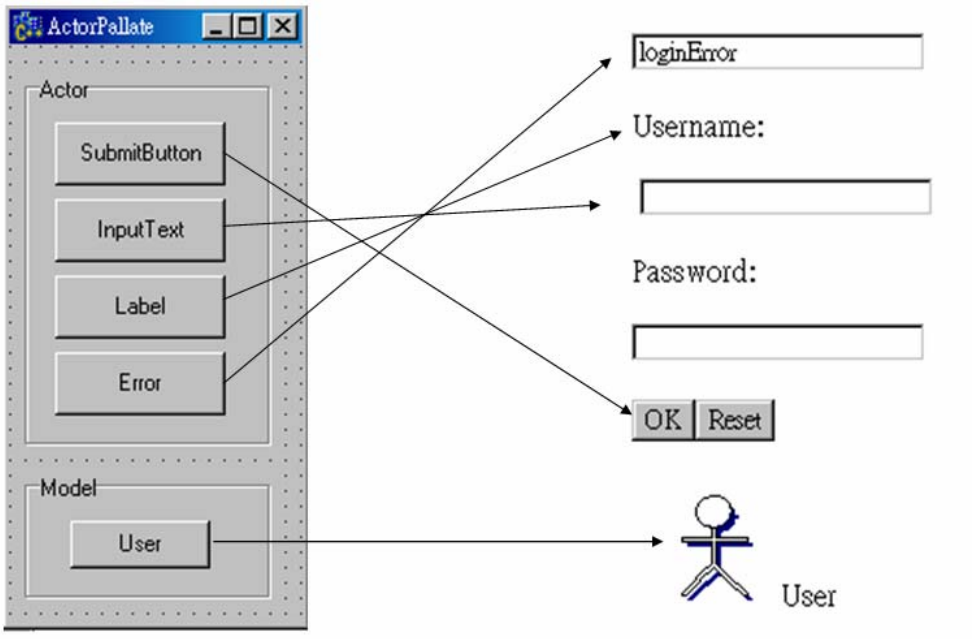

圖 9 視覺化建構程式畫面

下圖爲視覺化建構與視覺化語言之對映，我們用視覺化系統建構網站，各視覺化之元件皆會對映到 EBS 之 Script 描述。

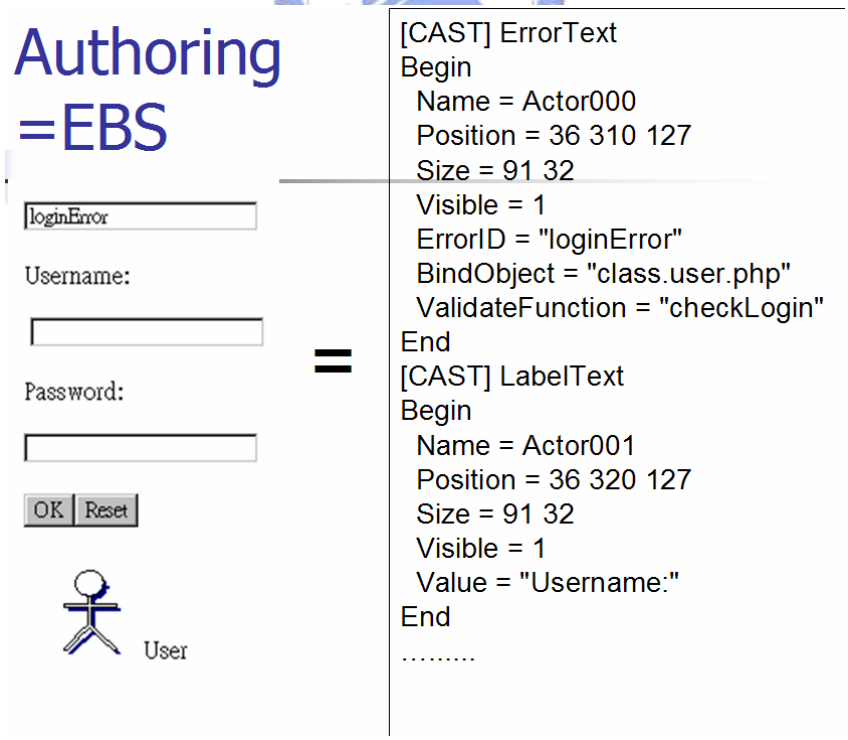

圖 10 視覺化編輯與 EBS 之對映

2. EBS->XForms 轉換器：

將 EBS Script 分析並依據 EBS 之各元件，產生對應之 XML 描述之使用者介面元件(XForms)，提供 UI Generator 模組運用。

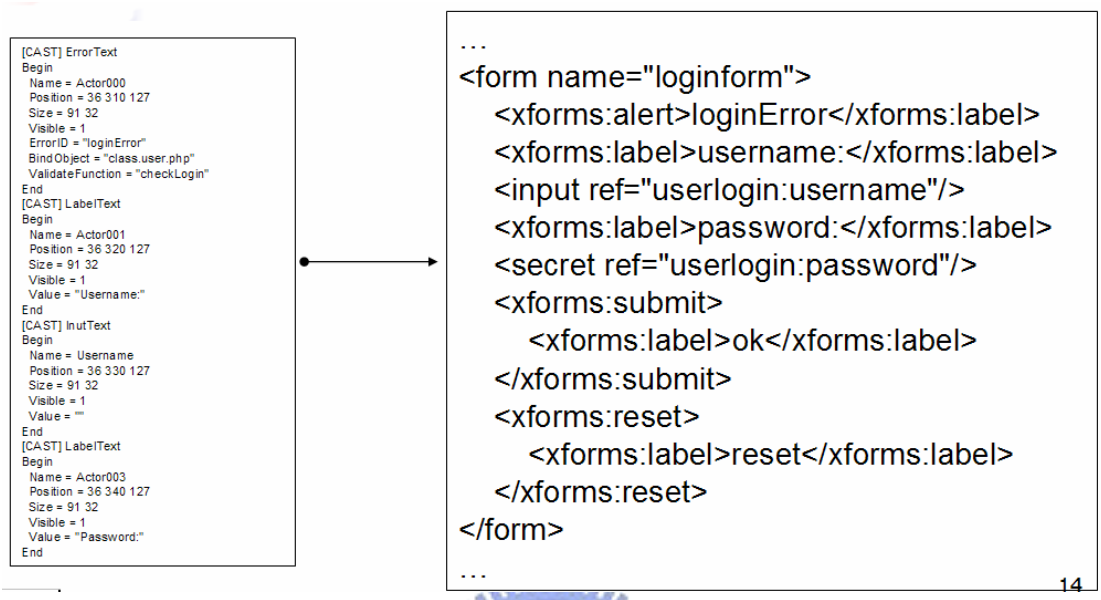

圖 11 EBS 轉換成 XForms 之對映

3. 程式碼產生模組(Code Generator)：

分析各元件之事件、屬性，並轉換成對應之伺服端程式碼(PHP)及 HTML 樣版(Template)，再整合 phpMVC-config.xml 上傳至伺服器執行。

此模組分別產生：

HTML 檔：

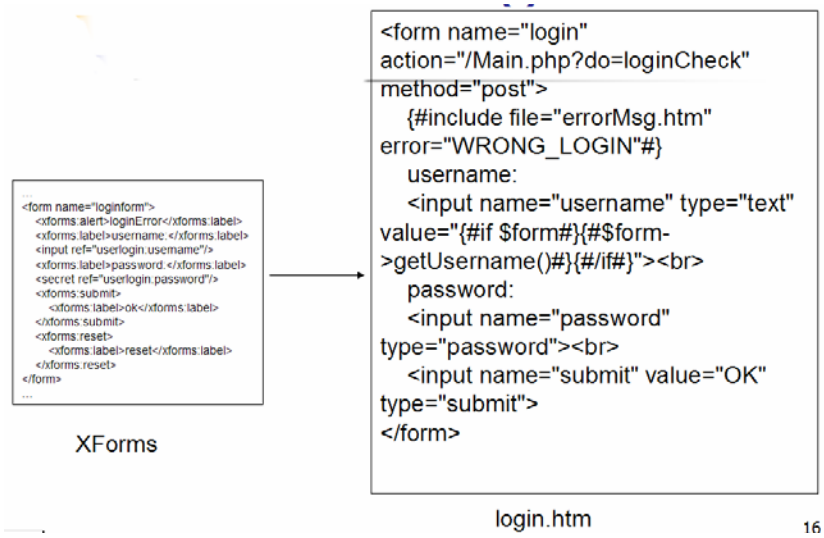

圖 12 程式碼產生器產生對映之 HTML 樣版檔

相關之 PHP 程式檔：

以使用者登入系統爲例，則有顯示登入畫面之 loginAction.php，顯示登入完成之 loginCheckAction.php，以及負責進行驗證使用者帳號、密碼之 loginCheckForm.php

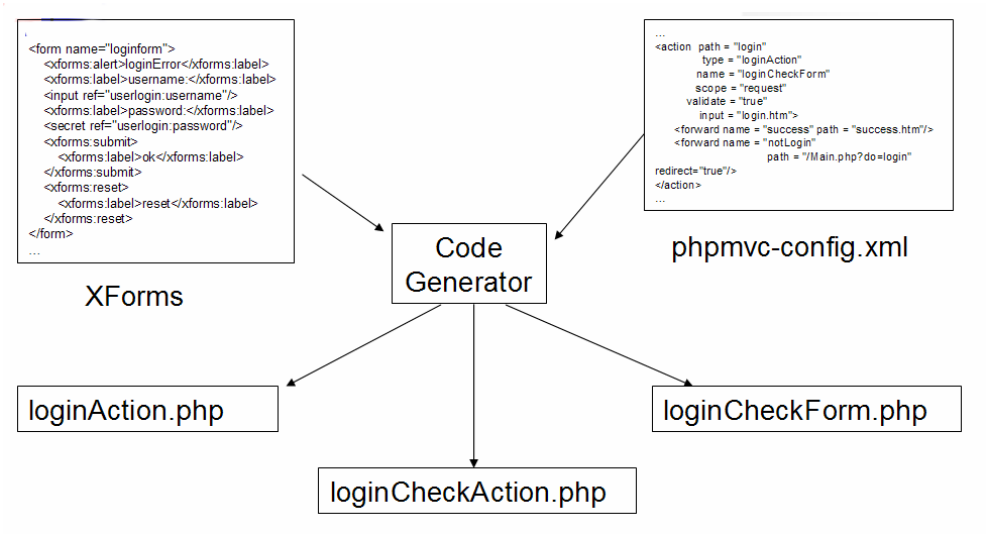

圖 13 程式碼產生器產生對映之 PHP 程式

3.2視覺化軟體開發環境架構分析

3.2.1 以編輯手延伸視覺化開發系統

視覺化開發環境是由”編輯手”這套軟體延伸而來，此軟體是全視覺化之教材設計軟體，使用者完全不需要程式設計的基礎，即可利用其建構多媒體之數位化教材。

編輯手原有相當多種多媒體之動畫演員，用以製作多媒體效果之教材，但目前開發之系統以視覺化建構網站應用程式爲主，故需實作網頁(HTML)之表單之演員，如下圖以使用者登入系統之網頁表單爲例：

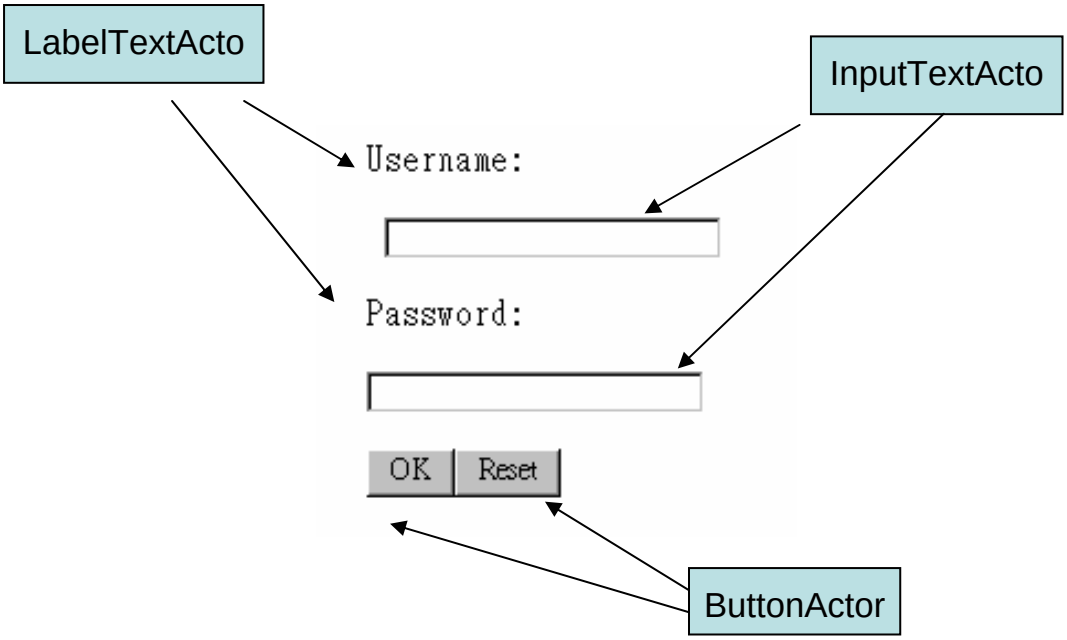

圖 14 網頁表單示意圖

HTML Form 最基本演員元件，如下表：

表 3 HTML 表單元件

演員類別	功能
Form Actor	表示使用 HTML Form
LabelText Actor	顯示純文字
InputText Actor	顯示文字輸入視窗
Button Actor	顯示按鈕

3.2.2 XForms

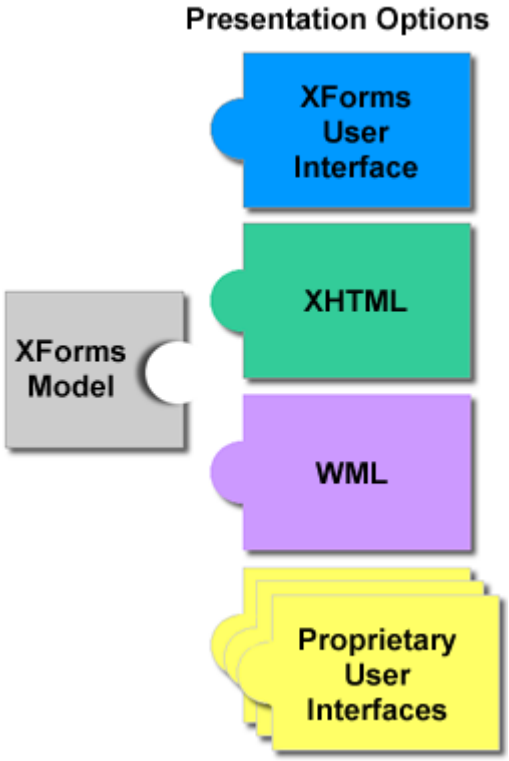

圖 15 XForms Model 之應用

參考資料：W3C XForms 1.0 [8]

由於 1993 年出現的 Web Form 逐漸不能滿足日益增加的需求，於是乎就有很多專家向 W3C 提出新一代 Web Form 的看法。然後由 W3C 的 HTML Working Group 的一個小組在 1999/4/22 ,提出 xForms 需求書(Forms Requirement)，而該小組日後即改組成今天的 XForms Working Group，負責日後有關 xForms 規範的發展[9]，XForms 已在 2003 年 10 月被 W3C 列入 Recommendataion[8]。

誠如 2002/11/12 的 xForms Version 1.0 的摘要中所述，xForms 是下一代的 Web Forms，其主要目的是要解決目前 Web Forms 所不能滿足在 Web Application 和電子商務應用上的需求。

優點：

- 支援各種設備，如手持式設備，電 視，電腦上的各種流覽器，此外尚支援印表機，掃描器等，將來一般使用者將可使用除電腦以外的資訊設備，如資訊家電，進入網際網路的世界。表格的設計也應考慮由紙張列印出來後， 填入資料，再由掃瞄機掃瞄輸入系統。
- 爲因應滿足各式需求，提供更豐富的人機介面

- 將資料、表單邏輯和畫面顯示分開，XForms 的設計將清楚劃分爲表格資料、表格邏輯和表格呈現三個階層，使得同一份表格可以紙張呈現或顯現於掌上電腦如 PDA。
- 改善國際化支援
- 支援結構化的表單資料
- 支援進階的表單邏輯
- 支援一頁多表單和一表單多頁呈現
- 支援暫停（Suspend）與恢復（Resume）的功能，在日常生活中，常有填寫表格需參考其他資訊的情況，以致中斷表格填寫。例如申報所得稅時，需個 別加總醫藥費用或房屋貸款，方便列舉報稅。因此，XForms 必須有一方法，在表格填寫過程中，能夠暫停一段時間後再恢復填寫。
- 能與其他 XML 應用整合，爲達到更豐富呈現效果的目標，XForms 在設計時便考慮到與其他 XML 應用的整合，如 XHTML、SVG（Scalable Vector Graphics）、SMIL（Synchronized Multimedia Integration Language）。同時也可使用樣式表（Style Sheet）語言，如 CSS（CascadingStyle Sheet）或 XSL（eXtensible Stylesheet Language）來調整版面效果。有關 XHTML 和 CSS

XForms 之定位應是作爲所有其他的資料輸入之角色，只要某一規範需從外界接收資料，就需要到 xForms。其與 XML Family 成員的關聯如下：

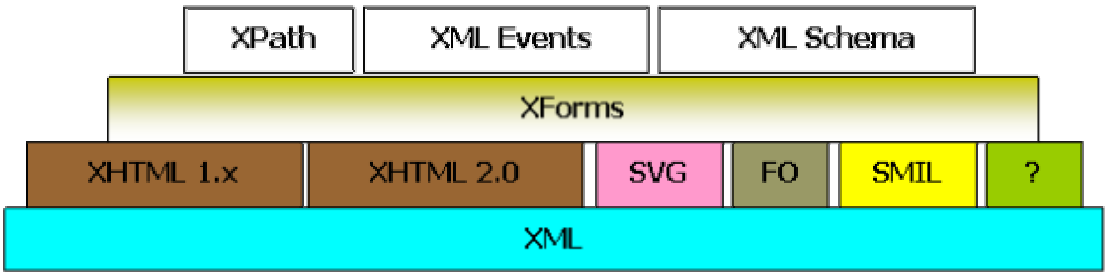

圖 16 Xforms 與 XML Family 成員之關聯

參考資料：W3C Xforms 1.0 [8]

1. XHTML
   XForms 會被整合至 XHTML 中的一個模組
2. Privacy Preferences Project (P3P)
   xForms 可以和 Privacy Preferences Project (P3P)一起使用，而 P3P 主要目的是能讓使用者在流覽網站時對自己個人資料有更佳的主控權。
3. Web Accessibility Initiative (WAI)

XForms 能讓發展 WAI 更加容易

4. CSS & XSL
   可以直接以 CSS & XSL 來格式化
5. VoiceXML
   應可支援在語音流覽器上使用
6. SMIL
   Xforms 需可和同步媒體結合
7. Scaleable Vector Graphics(SVG)
   Xforms 需和可延伸性向量圖形技術結合
8. DOM
   Xforms 需和 W3C 的文件物件模型(DOM)一致
9. XML Sschema
   Xforms 需和 XML Schema 相配

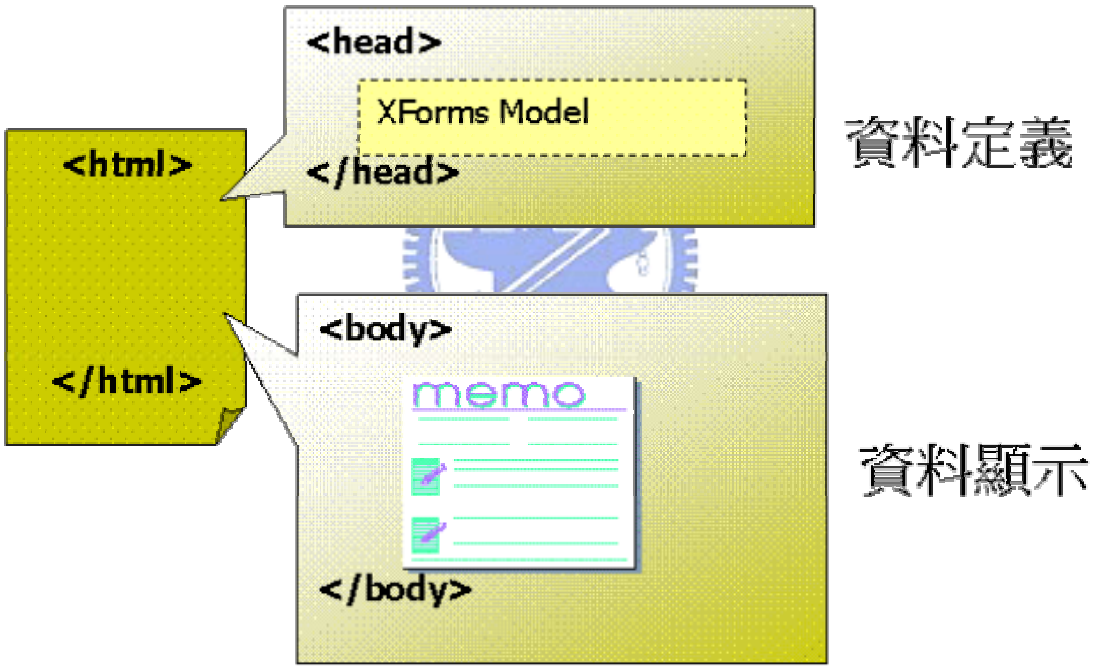

圖 17 XForms 之組成方式

參考資料：W3C XForms 1.0 [8]

XForms 主要將 Form 的 Model 定義在<head>、</head>間，資料顯示的部份則置於<body>、</body>間，以下爲採用 XForms 設計之表單：

```
<?xml version="1.0" encoding="iso-8859-1"?>
<html
 xmlns="http://www.w3.org/1999/xhtml"
 xmlns:xforms="http://www.w3.org/2002/xforms/cr" >
```

```
       <head>
              <title>Schema Driven Forms</title>
              <link rel="stylesheet" href="../main.css"
type="text/css" />
<!－開始定義資料模型 -->
      <xforms:model id="mdlDate">
         <xforms:instance>
            <instanceData
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            >
        <theDate xsi:type="xsd:date">2001-06-04 </theDate>
        <theNumber xsi:type="xsd:int">976</theNumber>
            </instanceData>
         </xforms:instance>
      </xforms:model>
<!－定義資料模型結束 -->
       </head>

       <body>
              <h1>xForms 介紹</h1>
              <p class="example">
              控制項的顯示之一
                     <br/>
                     <br/>
<!－xForm 資料控制項 -->
         <xforms:input ref="theNumber">
         <xforms:label>請輸入數字:</xforms:label>
         <xforms:hint>Enter an integer  here</xforms:hint>
         <xforms:alert>不能輸入非數字!</xforms:alert>
         </xforms:input>
              </p>
              <p class="example">
              控制項的顯示之二
                     <br/>
                     <br/>
<!－xForm 資料控制項 -->
       <xforms:input ref="theDate">
```

```
      <xforms:label>Please enter your date of birth:<br/>
               </xforms:label>
      <xforms:hint>Enter date of birth </xforms:hint>
      <xforms:alert>日期格式錯誤!</xforms:alert>
      </xforms:input>
                    Your birthday is: <b><xforms:output ref="theDate"
/></b>
             </p>
      </body>
</html>
```

XForms 分割爲三個層次：資料層（data layer）、邏輯層（logic layer）和呈現層（presentationlayer）。資料層讓使用者定義表格的資料模式（data model），也就是表格的抽象結構（abstract structure）。使用者可以利用已有的或自行定義資料型態（datatype）來描述表格資料。

邏輯層可讓使用者敘述表格欄位間的關係，例如欄位值的加總。XForms 將支援簡單的運算語法（expression syntax），且這些語法是建立在廣泛使用且熟悉的試算表功能上。當然，若要更彈性地使用 XForms，也可呼叫程序語言（script language）。

呈現層包含了表格控制（form controls）的標誌（markups）和 HTML 標誌，而每一表格控制限於資料模式中的一個欄位。使用者可依自己喜好，利用 getter 和 setter 功能(methods)呈現資料，如日期(dates)、貨幣(currencies)等。如此一來，資料內在的表現方式仍然不變，卻簡化表格的處理。

有了 XForm 我們就可以用來定義完整的 HTML Form 元件，未來可以直接適用在新一代的瀏覽器上，現在則可以根據 XForms 來實作 HTML Form 的 Render，用來產生網站應用程式的使用者元件的畫面。

3.3伺服端網站應用程式架構

一般網站伺服端應用程式的架構演進，我們採 JavaServer Pages[10]二階段來分析，可概略分爲 JSP Model1[10]及 JSP Model2[10].

3.3.1 JSP Model1

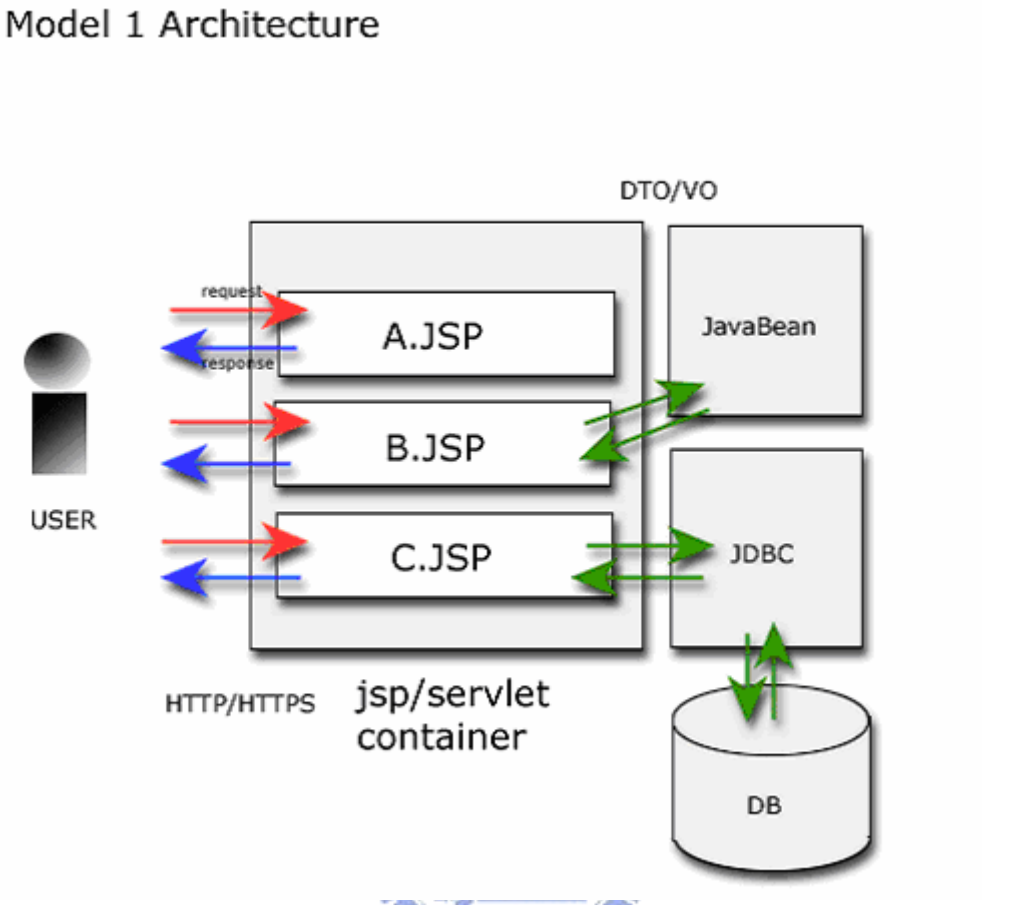

圖 18 JSP Model1 架構

Model1 架構，以程式+HTML 的方式製作程式，較進階的作法爲使用程式建構商業邏輯的物件，畫面的呈現仍以程式、HTML 建構，此種系統開發最爲直覺，而且是多數初學的開發者採用，市面上的書籍的範例程式也多採用此種架構。

此種架構之缺點爲：

1. 畫面呈現跟程式邏輯混雜在一起
2. 程式設計師與美工設計人員共同編輯同一檔案，易造成程式錯誤。
3. 移轉至不同用戶端相當困難，例如：手機、PDA 等。
4. 多國語言化困難，因程式、畫面混雜。
5. 維護程式成本高，較不適用於大型開發專案。

3.3.2 Model View Controller Design Pattern

在 1970 年代末期，當 GUI 發明出來時，軟體設計師看見應用程式有三個主要部份：處理資料的部份、建立畫面和報表的部份、以及處理用戶和其

他子系統之間互動的部份。在 1980 年到初期，ObjectWorks/Smalltalk 程式設計環境引入了這三位一體的觀念作爲開發架構。以 Smalltalk 80 的說法，資料系統稱爲模型(Model)，版面配置系統稱爲外觀(View)，而互動系統就稱爲控制元(Controller)[11]。

Smalltalk MVC 架構在一本相當著名的書 ”Design Pattern: Elements of Resuable Object-oriented Software”，其中有通知/訂閱協定(notify/subscribe protocol)以及觀察者設計模式(Observer pattern)的範例，此範例的本質是系統需要替相同資料顯示不同的外觀，像是條狀圖、扇形圖、以及試算表。這是劃分應用程式的絕佳基由，而且這個範例也常在許多地方被引用。

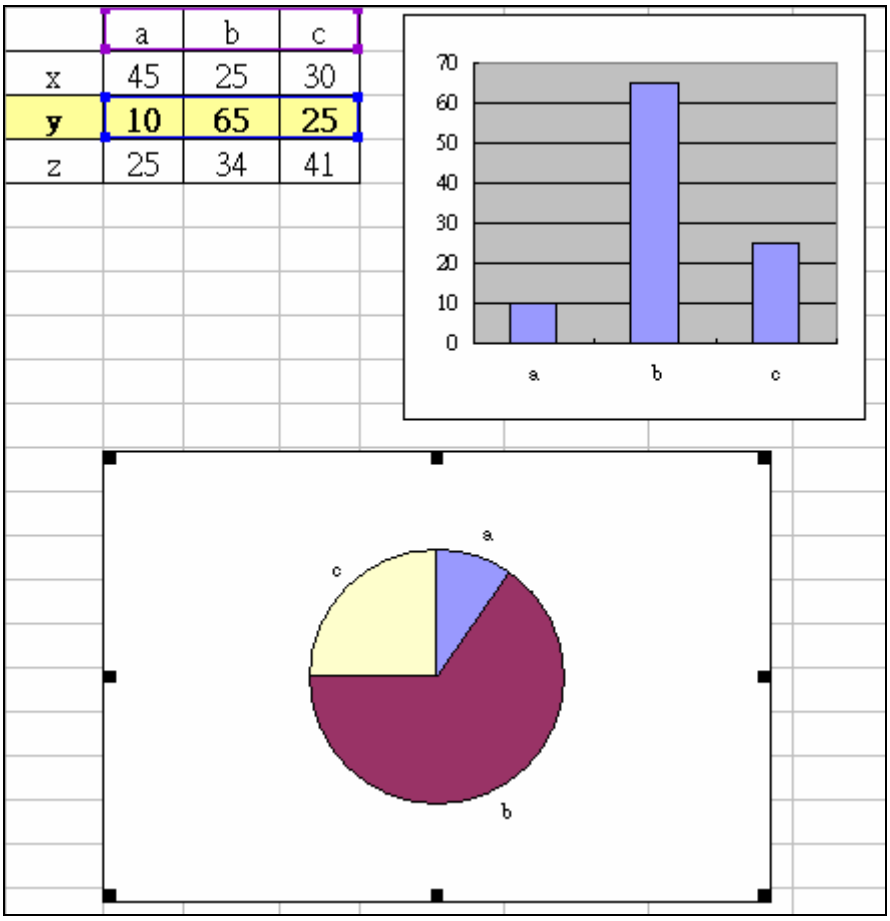

圖 19 模型資料可以用在不同的外觀內

如上圖，每一種外觀可以在同一時間顯示給不同的用戶看。應用程式必須在底層資料或是模型(Model)變更時，隨時保持外觀能隨之更新。爲了更新模型，用戶交付一道請求給控制元，再由控制元來協調模型的變更。然後，顯示資料的外觀必須跟著更新，來反應模型的最新狀態。

Smalltalk MVC 手段是使用一個觀察者通知設計模式(Observer notification pattern)。在這個設計模式中，每種外觀都註冊成模型資料的觀察

者。接著模型可以送出一道訊息通知所有已註冊的觀察者以通知外觀做更新。後來 Smallalk MVC 架構被修正成通用的 MVC 架構標準，使其能適用任何平台的程式建造工程。

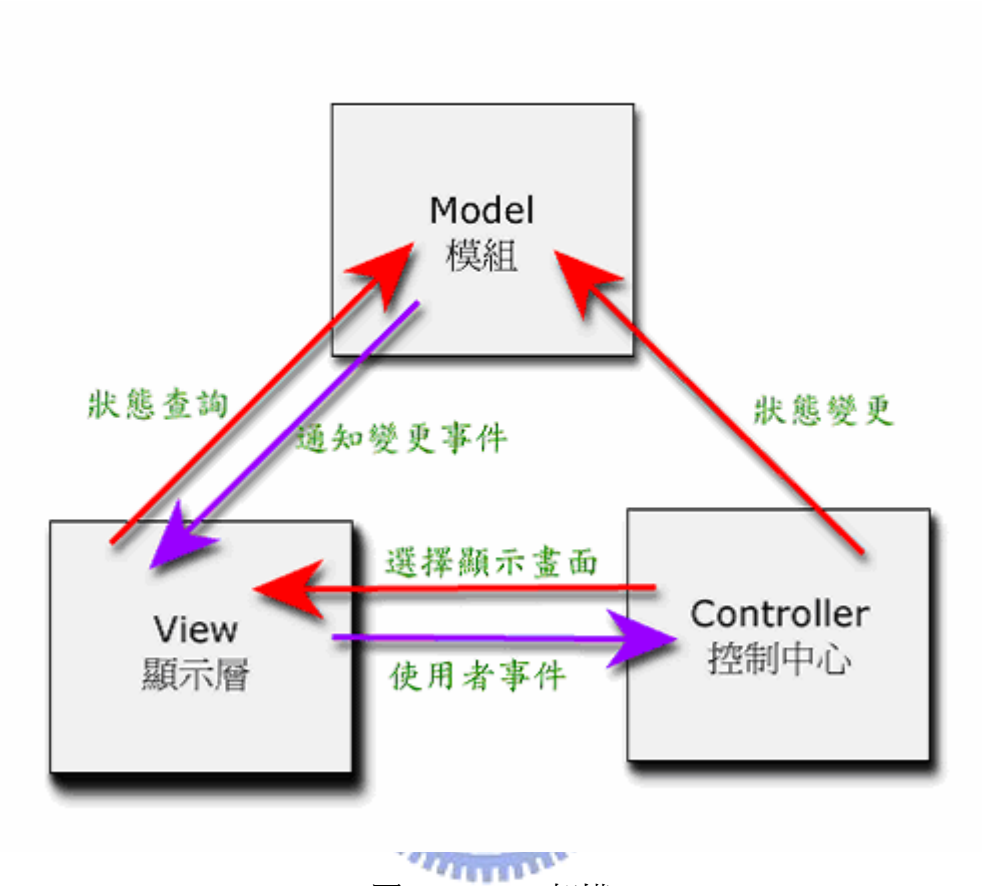

圖 20 MVC 架構

上圖爲 MVC 最常採用的標準模式圖：一個三角形，三角上有互相連接的元件。網站應用程式要維護圖中「變更通知」的部份是相當困難的。當所有的資源都位在相同伺服器上，而且客戶端和伺服器之間的連線維持不斷時，這種情形當然可以處理得很好。但是當資源是分散在好幾台伺服器上，而且客戶端和伺服器上的應用程式之間並沒有維持不斷的連線時，處理的困難度便提高很多。

很多分散系統的建構者，包括網站應用程式在內，都會避開讓外觀做狀態查詢的想法。最常見的是，遠端應用程式會以分層設計模式(Layers pattern)來設計，也就是說必免同層或相鄰層的類別此互動，對複雜的應用程式而言，這樣可以避免新增元件時，複雜度呈指數成長。分層在設計遠端應用程式上是核心的設計模式。

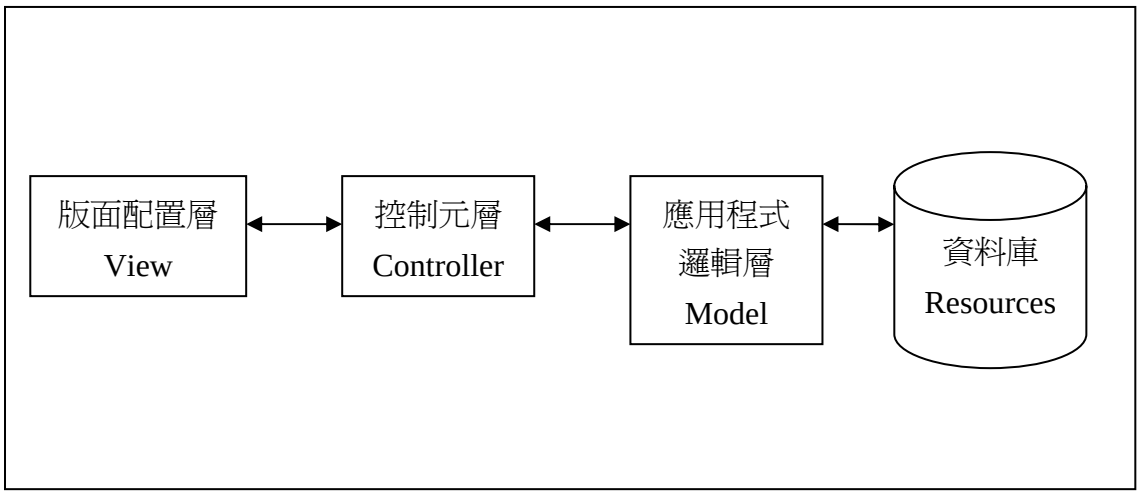

圖 21 網站應用程式分層架構

若以 MVC 爲背景，引入分層設計模式時，當通知變更時，變更狀態和查詢狀態的責任就會變成控制元的責任。分層後的網站應用程式使用的設計模式比傳統的 MVC 要好。控制元是位在版面配置層(View)以及應用程式邏輯層(Model)之間的夾心。

每一元件的主要責任未變，流程只有些許的改變，任何狀態查詢或變更通知，必須經過控制元。另一個不同的地方是，當外觀或版面配置層在設計動態內容時，會使用控制元傳遞來的資料，而不是由模型直接傳遞的資料。這種改變把外觀和模型拆開來，使得控制元可以選擇資料，以及用以顯示資料的外觀，以下整理 MVC 分層模式：

1. Model：

Model 爲程式商業邏輯所在，是整個應用程式之核心，對資料庫、檔案系統之實際讀寫，皆在此進行，Model 從從 Controller 取得資料，並實際對 Model 的 Instance 進行設值動作，並提供 Interface 供 View 及 Controller 使用。

2. View

負責畫面之呈現，由 Controller 控制選擇 View，使用者輸入的事件也從 View 觸發並通知 Controller，另外，此層不應有商業邏輯存在。

3. Controller

Controller 更改 Model 之資料，接受所有 Client 透過 View 傳來之 request，當 Model 有所改變時通知 View 作畫面的調整。

3.3.3 php.MVC 架構

下圖爲網站系統在伺服端執行時之流程：

1. 使用者從瀏覽器觀看網頁後，點選連結或按鈕後送出 HTTP 請求，此時在伺服器端由 ActionServlet 接受請求，並至 1-1 查看此請求之 URL 在 phpmvc-config.xml 定義爲何，如果有需要驗證資料則進行步驟 2 否則進行步驟 3
2. ActionServlet 將使用者傳過來的資料(如：輸入之帳號、密碼)封裝放入 FormBean
3. 執行 Action
4. 需要使用者輸入之資料時則到 FormBean 取得
5. 呼叫相關 Model 的 Method，執行商業邏輯
6. 傳回執行完後之 TO，即執行完 Action 後要再將程式導向到何處
7. 根據執行狀況 return ActionForward 到 View 層，顯示成功、或錯誤之畫面

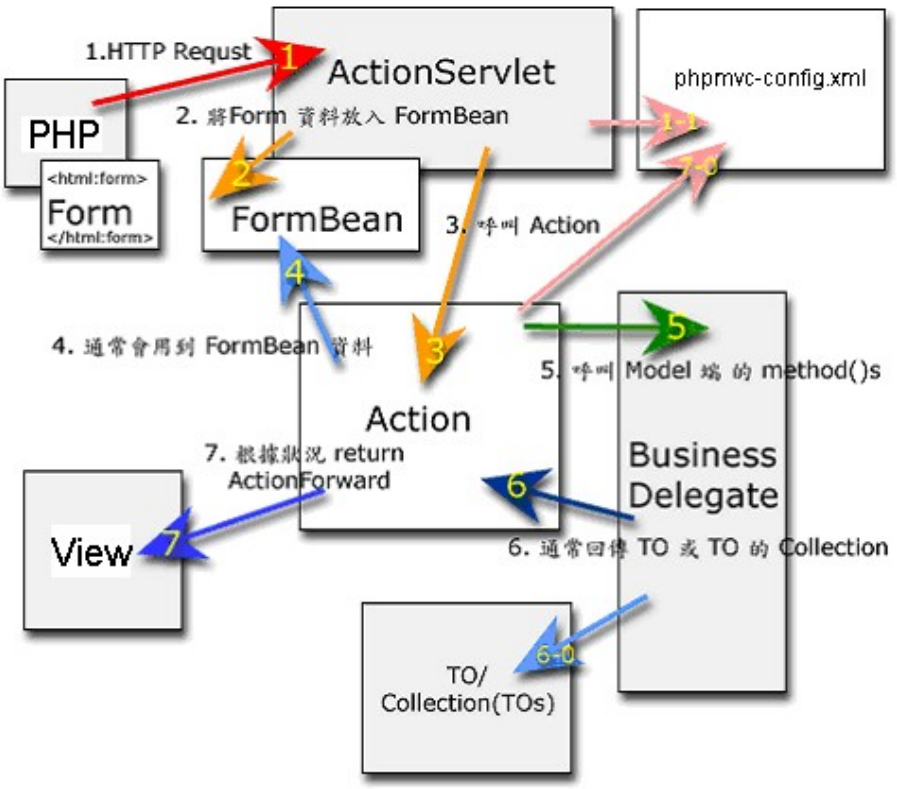

圖 22 php.MVC Framework 運作流程

本開發系統在伺服端架構採用 JSP Model2 架構，使用 php.MVC[712]此 framework 爲基礎，並加上自行開發之 driver 來建構伺服端程式的所有框架。

以下爲 php.MVC 的主要功能與優點：

- 開放原始碼：使用者可以完全掌控整個系統之原始碼，並且可以視需求去修改。
- 安全性：每個 php.MVC 應用程式，只有一個進入點，一般使用者使用應用程式時，只能接觸到這唯一個程式，因此資料的安全更容易達成。
- 安裝彈性：每個 php.MVC 應用程式的安裝，不需要放在 php.MVC 的 Library 目錄下，我們可以將 php.MVC Library 安裝在/usr/local/lib 路徑對，其他的應用程式可以自行安排安裝位置，而不受限制。
- 無應用程式數上限：一套 php.MVC Library 可以提供無限個應用程式使用，無數量上之限制。
- 物件導向設計(OOD)：php.MVC 是以物件導向設計方式來建構的，因而具有可擴充、易維護之好處。
- 整合資料庫：php.MVC 提供 Pear::DB 資料庫抽像層的整合，並提供 MySQL 資料庫系統之驅動程式。
- XML 設定：每個 php.MVC 的設定皆使用一個 XML 檔案。
- MVC Model 2 設計：php.MVC 實作 Model-View-Controller(MVC) Model 2 設計樣式，此設計樣式可以讓應用程式之表現層(Presentation layer)跟商業邏輯(Business logic)分離，設計者與程式設計師可以分別專注於各自專業的工作上。
- 訊息資源：php.MVC 提供 PropertyMessageResources class，此功能可以控制、置換文字訊息，提供多國語言化應用程式之用。
- Struts：php.MVC 是根據 Jakarta Struts[13]的設計實作，Struts 已經被證明是可靠的設計。

php.MVC 的基本架構與，Jakarta 的 Struts 類似，主要差別在於使用的語言，Struts 以 Java 開發，主要搭配 JSP/Java Servlet 系統使用，而 php.MVC 模仿 Struts 之架構，以 php 實作出 MVC 這個設計樣式，以下分別說明 php.MVC 的各主要元件：

1. Action Servlet

   爲 Controller 之主體，接收所有的 request，讀取 xml 格式之設定檔

後，決定是否要對表單(Form)作檢查(Validate)，並將包裝好之表單物件(Form Object)傳給 Action 物件供進一步之使用。

2. FormBean

表單元件，主要功能爲包裝、驗證使用者使用 HTML 表單所傳過來之值，並將值包裝進 FormBean，供 Action 使用，驗證如果發生錯誤，則傳回 ActionError 給 ActionServlet 以進行錯誤畫面之處理。如果驗證通過則傳回 NULL 給 Servlet。

3. Action

Action 實際與商業邏輯部份進行互動，並根據 Model 處理的結果，決定使用者的下一個畫面(ActionForward)。

4. phpmvc-config.xml

此 XML 設定檔爲，Controller 運作最重要的核心，所有 Action、ActionForward、FormBean、Plugin、Datasources，皆在此設定檔作設定。ActionServlet(Controller)根據此設定檔，執行網站程式之運作，從此設定檔也可以很清楚的看到網站運作的流程。

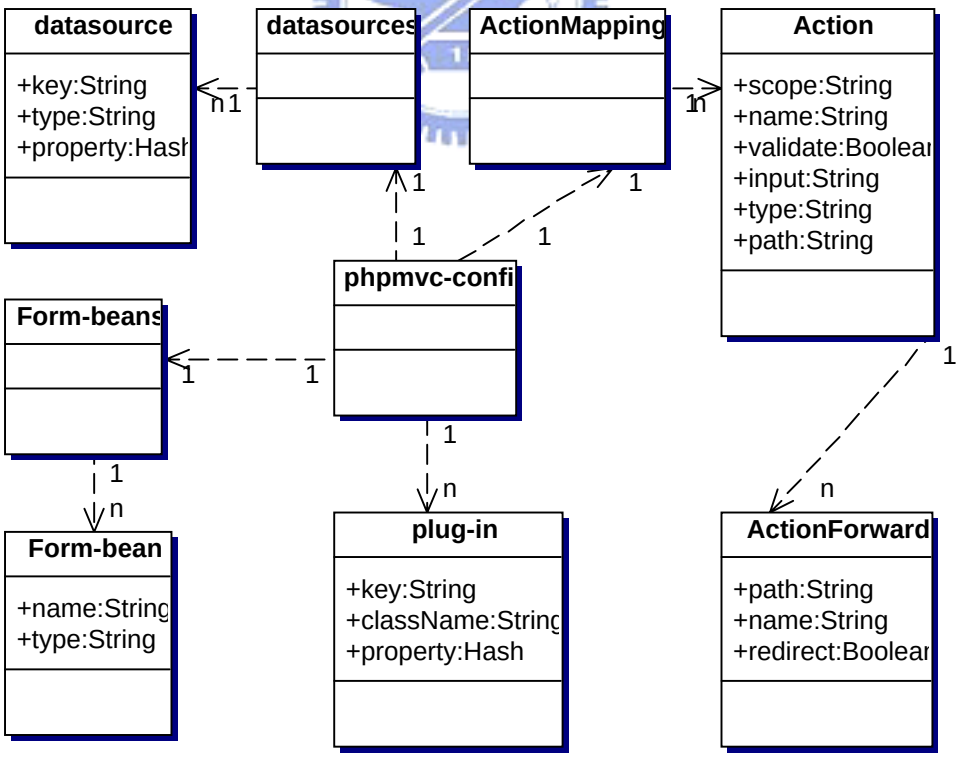

圖 23 phpmvc-config.xml Class Diagram

5. Model

主要商業邏輯所在，實際與資料庫、檔案系統等各種系統資源作存取或設定等動作。

6. View

負責網站應用程式之畫面呈現，主要由 Controller 控制，從 Action 傳至 ActionForward 物件執行。

php.MVC 提供的 MVC Framework 雖不如 Jakarta Struts 完整，但已足以用來建構大型 php 網頁應用程式，加快開發速度，也可增加軟體可重用性。

3.3.4 網站應用程式 Action 使用實例

接著以一個網站使用者登入系統程式爲例，說明 controller 如何透過 phpmvc-config.xml 的設定來控制整個網站應用程式運作的流程：

```
<form-beans>
    <form-bean name="loginCheckForm" type="loginCheckForm"/>
</form-beans>
<action-mappings>
    <action    path = "login"
               type = "loginAction"
               name = ""
              scope = "request"
           validate = "false"
              input = "">
     <forward name = "success" path = "login.htm"/>
   </action>
   <action    path = "loginCheck"
              type = "loginCheckAction"
              name = "loginCheckForm"
             scope = "request"
          validate = "true"
             input = "login.htm">
     <forward name = "success" path = "/Main.php?do=userHome"
redirect="true"/>
     <forward name = "exception" path = "exception.htm"/>
   </action>
</action-mappings>
```

1. 建構 HTML 樣版 login.htm 用來顯示登入畫面，其中的 form 是 submit 到 loginCheck。
2. 建構程式 loginAction 用來顯示 login.htm(即登入畫面)
3. 建構程式 loginCheckForm，用來包裝 loginAction 傳過來之 username、password 等，並加以驗證，如驗證失敗，則 Controller 會返回 login.htm 並顯示登入錯誤之訊息。
4. 如果成功則 Controller 會 forward 到 userHome。

圖 24 及圖 25 即爲此登入系統流程式意圖：

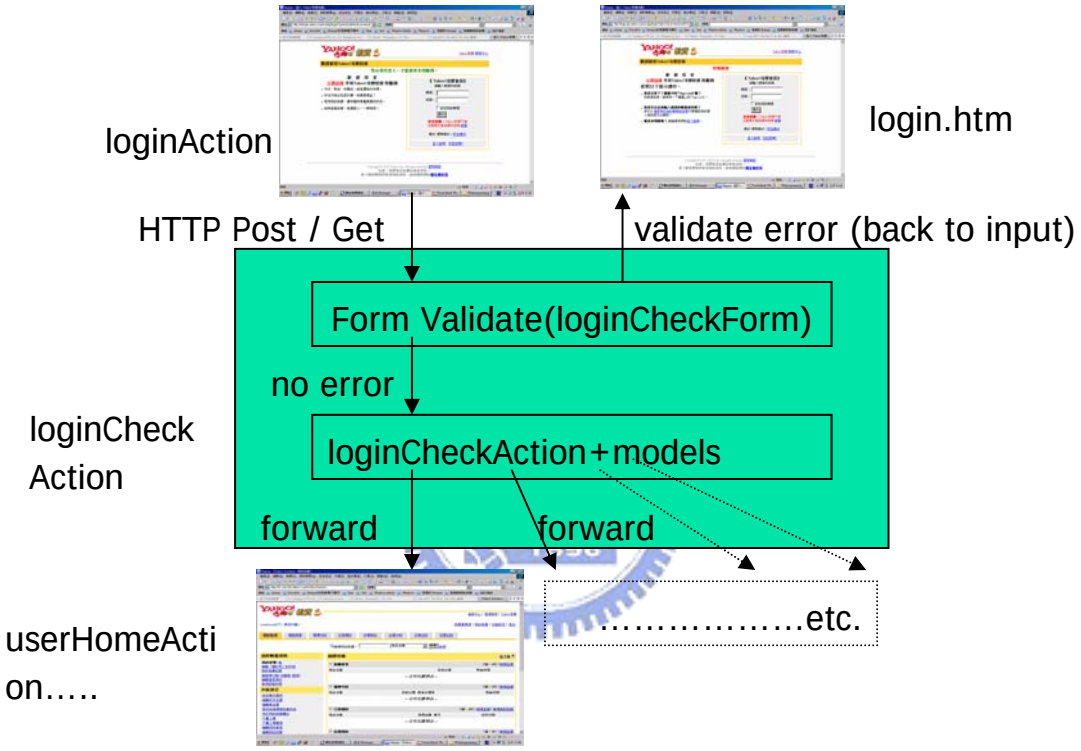

圖 24 網站登入系統流程示意圖 1

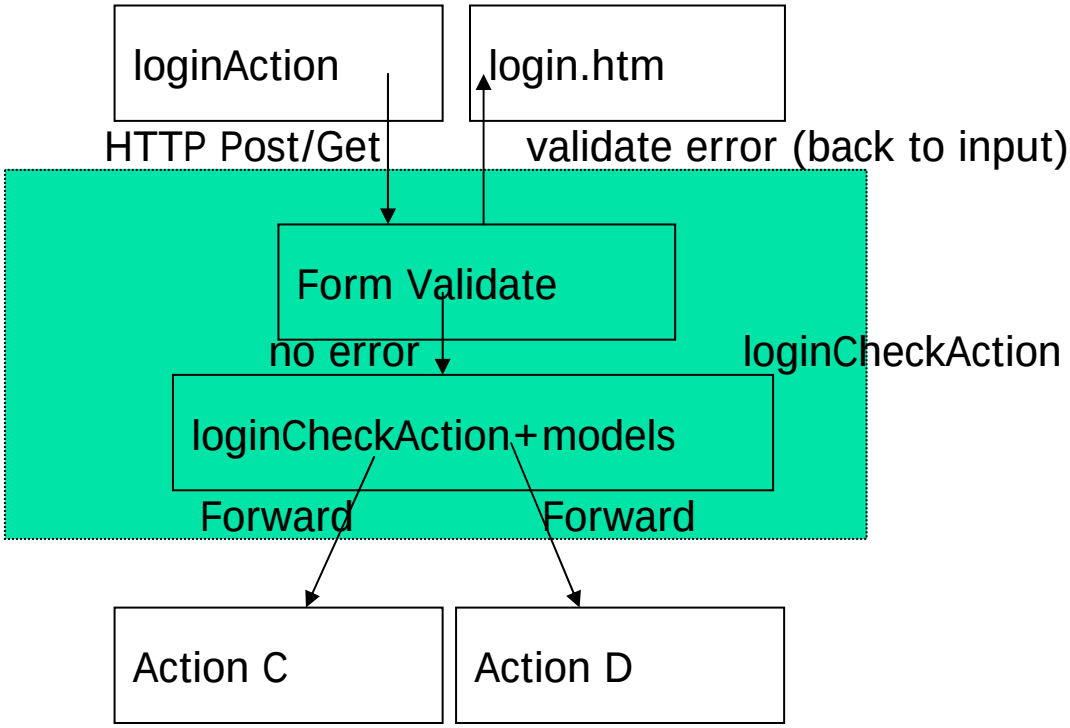

圖 25 網站登入系統流程示意圖 2

3.3.5 php.MVC plugin

php.MVC 提供一簡易的 plugin 介面，可以將常用的網站程式模組以 driver 的型式跟整個 php.MVC framework 整合。本系統實作二個(Template、Datasource) plugin driver，如下圖所示：

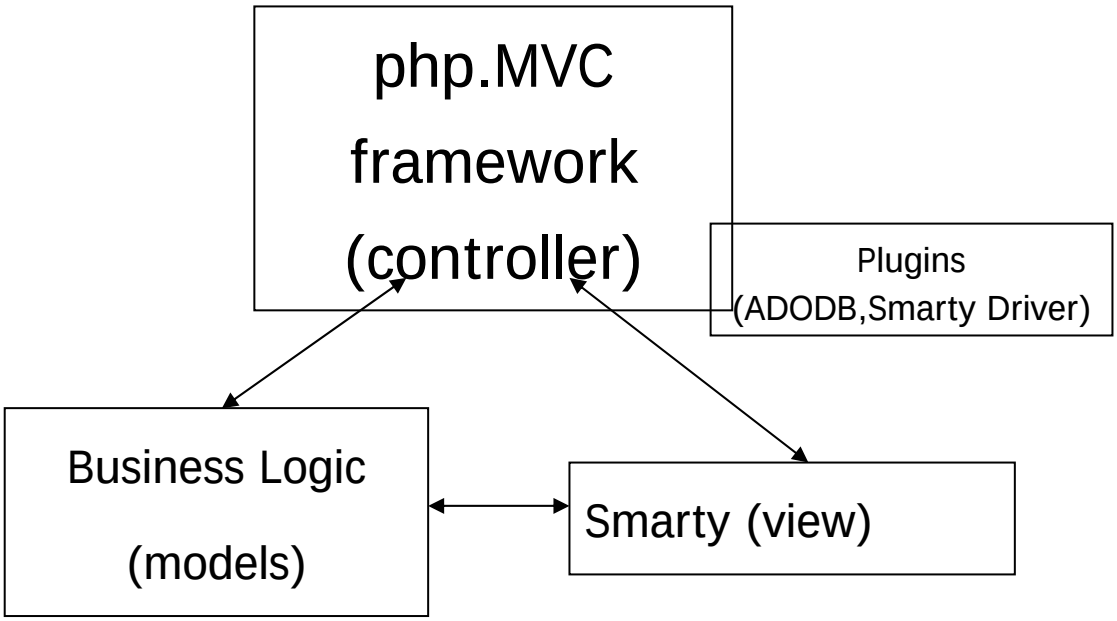

圖 26 Smarty Driver 、ADODB Driver 在 php.MVC

1. php.MVC 在 View 的處理並未提供樣版引擎(Template Engine)，我們採用 php 中最被廣泛採用的 Smarty(http://smarty.php.net)來加強 php.MVC framework 對於 view 的處理。以 plugin driver 的型式在 phpmvc-config.xml 作設定，並在 Action 中以 getPlugin 的方式取得 Object 的 handler。

2. php語言對於各種不同資料庫的存取，如(MySQL、Oracle…)，皆以不同的function實作，並未提供一個完成的Database Abstraction Layer來整合所有資料庫的存取動作。ADODB(http://php.weblog.com)是johnlim以php撰寫的Database Abstraction Layer，提供很方便的彈性，讓開發者可以類似微軟ADO的方式去作資料庫的連結操作，也讓系統轉換資料庫系統變成可能，如(MySQL->Oracle)。我們實作一個Adodb Datasource的plugin整合進php.MVC的Framework，用來作爲存取資料庫的統一handler。

3.3.6 應用程式檔案配置架構

圖 27 應用程式檔案配置架構圖

網站應用程式所運用到的資源、檔案相當多，故有並要規劃一方便管理、維護之檔案配置架構，以下爲伺服器端安排相關檔案之配置：

1. WEBROOT 資料夾：爲整個應用程式之資料夾，應用程式進入點爲 Main.php 即位於此資料夾。
2. WEB-INF 資料夾：位於 WEBROOT 資料夾下，此權限設定爲只有系統可以讀取，主要放置 phpmvc-config.xml 設定檔。
3. actions 資料夾：位於 WEB-INF 下，主要放置所有網站應用程式 Action 的 class。
4. forms 資料夾：位於 WEB-INF 下，主要放置所有網站應用程式 Form 的 class。
5. include 資料夾：位於 WEB-INF 下，保留使用者自訂程式或函式庫之用。
6. models 資料夾：位於 WEB-INF 下，商業邏輯(Business Logic)之 Model 等 class 皆置於此。
7. tpl 資料夾：位於 WEB-INF 下，主要放置樣板網頁(Template)之用。

四、 以視覺化軟體開發網站應用程式系統設計

整個開發系統主要分爲以下幾個模組：

1. EBS->XForms Converter
2. XForms UI Render
3. PHP Parser
4. phpMVC-config.xml editor

4.1 EBS->XForms Converter

此模組主要將 EBS Script 轉成 XForms，供 XForms UI Render 產生畫面，及系統最後產生完整應用程式碼之用。

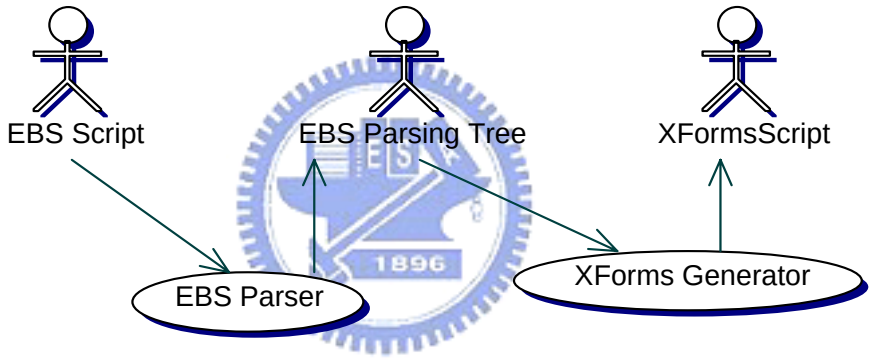

圖 28 EBS to XForms Use Case Diagram

4.1.1 Ebook Script(EBS)

Ebook Script[14]，爲一以視覺化描述多媒體電子書之語言，爲配合系統需求產生 HTML Form，新定義許多 HTML Form 演員，分別對應之 EBS Class Diagram 如下：

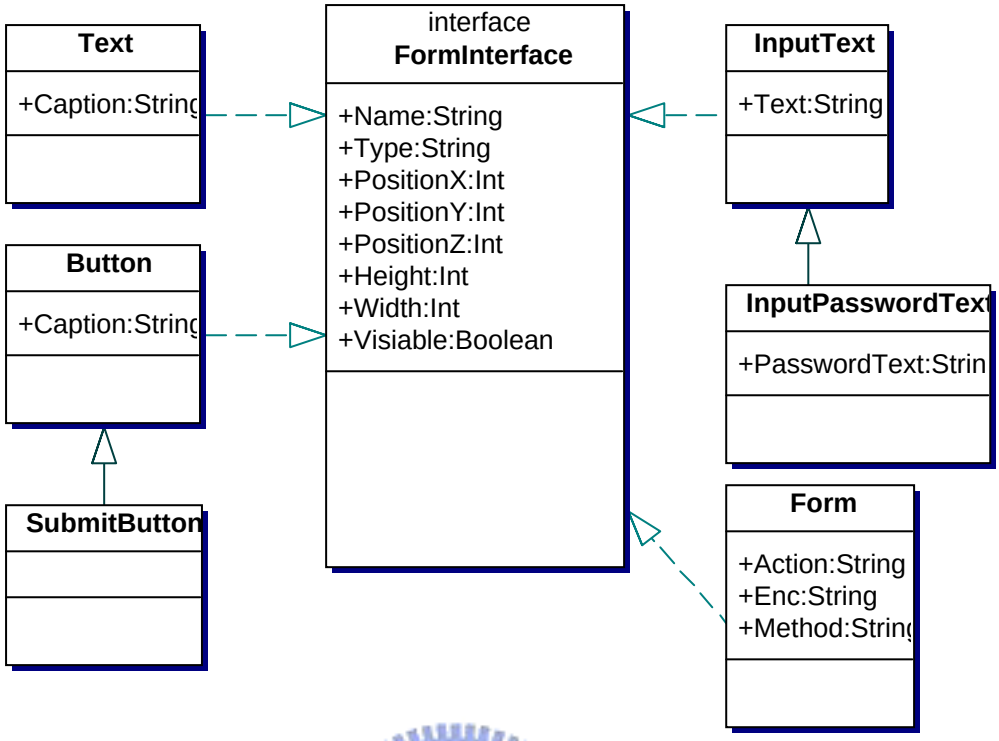

圖 29 新增 Form Actor 之 Class Diagram

使用 EBS 語言，設計上表之 Form Actor、Text Actor、InputText Actor、InputPassword Actor、Button Actor、SubmitButton Actor 等元件

4.1.2 EBS Parser、XForms Generator

使用 EBS Parser 將各 HTML Form 元件以 XForm 之規格重新輸出如 EBS Script：

```
[CAST] InputText
Begin
  Name = Actor000
  Position = 36 310 127
  Size = 91 32
  Visible = 1
  Text = "測試"
End
```

轉爲 XForms:

```
<input ref="InputText">
    <value>測試</value>
```

```
</input>
```

各元件在 XForms 上的排列順序，以各元件的 y 座標爲準，由上向下排列，最後輸出完整的 XForms 文件，供程式碼產生器使用。

4.2 XForms UI Render

實作將使用者元件，由 XUL 轉成對應之 HTML Template 檔案

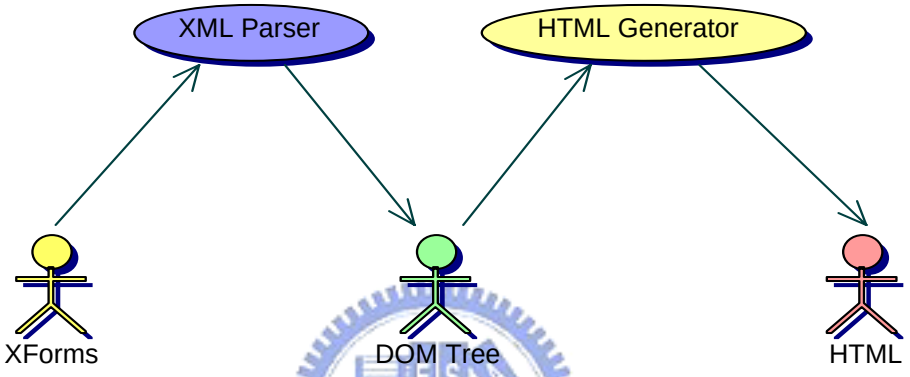

圖 30 XForms UI Render

此模組之功能，再將 XForms 之檔案，以 XML Parser 讀入成 DOM Tree，再轉換爲一般 HTML 使用之表單

XForms:

```
<input ref="InputText">
    <value>測試</value>
</input>
```

轉爲 HTML：

```
<input type=”text” name=”InputText” value=”測試”>
```

仿以上之作法，我們可以將最常使用之 HTML 表單，由 EBS 到 XForms 再到 HTML 作對映。

4.2.1 XML Parser

實作之 XML Parser 採用 Document Object Model(DOM)，是採用將整份 XML 文件讀入記憶體，並以 Tree 的形式來處理，好處在於可以自由的處理

整份 XML 文件，移動其中的節點，更改節點之值等，缺點是如果 XML 檔案較大，處理所佔用的記憶體較多，效能也會變差。

但因網站應用程式每頁之使用者介面之元件不大，以 XML Parser 以 DOM(Document Object Model)讀入 XForms，解析成 DOM Tree。

4.2.2 HTML Generator

實作將 DOM Tree 作 Tree Traversal ，並加以轉換成 HTML 輸出。以下爲使用者登入系統爲例之原始 XForms:

```
<head>
<xform xmlns=”http://www.w3.org/2000/xforms”
     action="/Main.php?do=loginCheck" method="post"
     id=”loginform”>
     <model>
        <group name=”userlogin”>
           <string name=”username”/>
           <string name=”password”/>
         </group>
     </model>
</xform>
</head>
<form name="loginform">
    <xforms:label>使用者名稱:</xforms:label>
    <input ref="userlogin:username"/>
    <xforms:label>密碼:</xforms:label>
    <input ref="userlogin:password"/>
    <xforms:submit>
       <xforms:label>OK</xforms:label>
    </xforms:submit>
</form>
```

輸出 HTML:

```
<form name="login" action="/Main.php?do=loginCheck" method="post">
    {#include file="errorMsg.htm" error="WRONG_LOGIN"#}
    使用者名稱:
    <input name="username" type="text" value="{#if
$form#}{#$form->getUsername()#}{#/if#}"><br>
    密碼:
    <input name="password" type="password"><br>
    <input name="submit" value="OK" type="submit">
</form>
```

其中有兩段 Smarty 之 tag：

{#if $form#}{#$errorMsg#}{#/if#}

用來顯示登入錯誤時顯示之錯誤訊息，當$form 這個物件存在時，表示登入處理有誤，才顯示錯誤訊息。

{#include file="errorMsg.htm" error="WRONG_LOGIN"#}

這段用來當使用者輸入使用者名稱後，但後續登入處理有誤時，回到輸入畫面，系統自動填入使用者之前輸入之值，WRONG_LOGIN 是錯誤訊息之 key 用來取得某錯誤訊息之用。

errorMsg.htm 內容：

```
{#if $errors#}
{#assign var='eMsg' value=$errors->get($error)#}
        {#if is_array($eMsg)#}
            <font color="red">{#$eMsg.0#}</font>
        {#/if#}
{#/if#}
```

如果$errors 物件存在，則根據使用者所給之鍵值(WRONG_LOGIN)去$errors 物件中取得正確之錯誤訊息。從 XForms->HTML 產生器中，系統加入了 Smarty Tag 用來實作錯誤訊息處理。使用者不需要如傳統網站程式開發，手動增加：

1. 錯誤訊息顯示處理
2. 使用者之前輸入或選擇之值

3. 這些複雜運作都由系統的 Controller 配合 View(Smarty)處理掉，減少程式設計師之負擔。

4.3 PHP Parser

下圖爲 PHP Parser 之主要組成部份與流程：

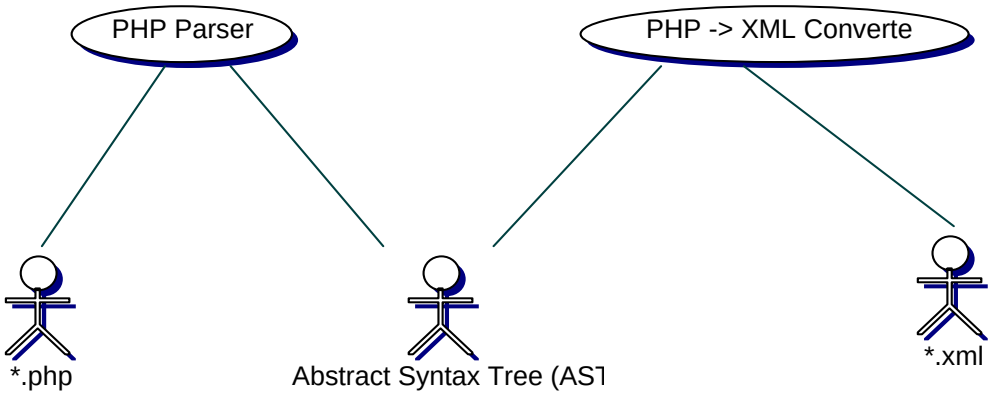

圖 31 PHP Parser 架構

1. *.php 爲輸入之 PHP 程式檔
2. PHP Parser 爲將 PHP 程式轉爲 Abstract Syntax Tree 之模組
3. Abstract Syntax Tree 爲 PHP 語法之抽象樹
4. PHP->XML Converter 將 AST 轉成 XML
5. *.xml 則爲產生之 PHP XML 檔

本節實作一 PHP Parser 可將 PHP 程式轉爲 XML，供未來可以載入可重用之 PHP 元件，並進而供使用者管理這些屬於商業邏輯之元件。

4.3.1 Parser Generator

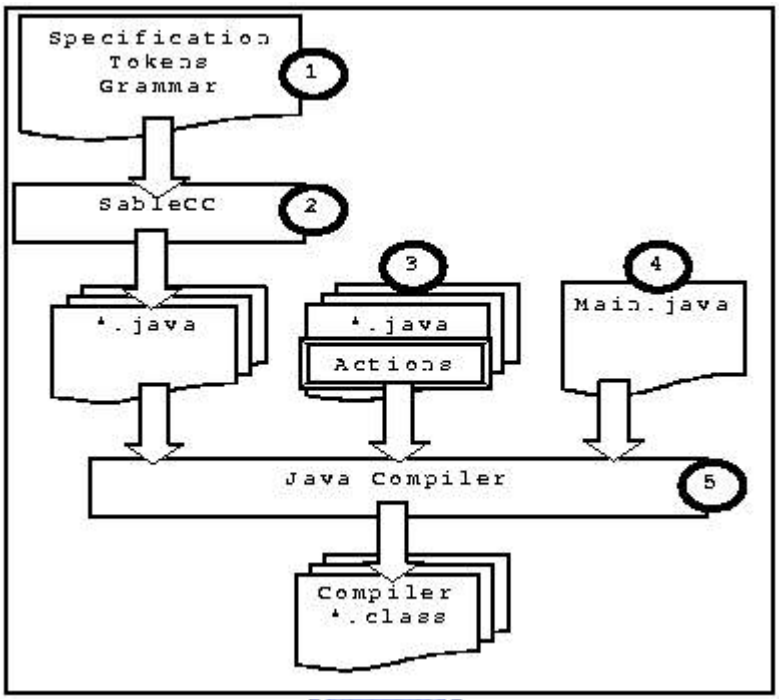

圖 32 使用 SableCC 建構 Parser 的步驟[15]

使用 SableCC[15](建構 PHP Parser 的步驟：

1. 建立 SableCC 的 specification 檔案，其中要有語言 lexical 的定義，以及語言之 grammar，我們實作 php4.sablecc3。
2. 針對定義之檔案，執行 SableCC，SableCC 會產生 org.sablecc.php4 這個 Java Package 以及 lexer package、parser package、node package、analysis package

本 PHP Parser 實作採用 SableCCJava Parser Generator)，使用 Parser Generator 的好處在於，我們只需依據 SableCC 語法定義出語言的規格檔，SableCC 就會幫我們產生完整的語言 Parser。

然後，我們實作兩個 Java Class 用來顯示 Abstract Syntax Tree(AST)及最後的 XML 檔案。

1. phpAST
   此 package 是 AST Tree Walker，用來產生圖形顯示的 AST，方便我們進行除錯之用。
2. phpXML
   此 package 用來 Walk AST，然後產生對應之 XML node，並作最後輸出。

圖 33 PHP->AST->XML 示意圖

4.1.1 XML->PHP Generator

上節我們將 PHP 的程式碼轉爲 XML，本節我們要實作由 XML 轉回 PHP 程式碼，主要採用的技術爲 XSL/XSLT[16]。

XSL 主是用來定義 XML 文件的呈現及轉換，其中主要包含三個部份：

1. XSL Transformations (XSLT)：用來轉換 XML 的語言
2. XML Path Language (XPath)：XSLT 使用 XPath 這個語言來作讀取 XML 文件。
3. XSL Formatting Objects (XSL-FO)：XML 的字彙表，用來作語意格式之用。

如下圖即爲使用 XSL/XSLT 轉換 XML 文件之流程：

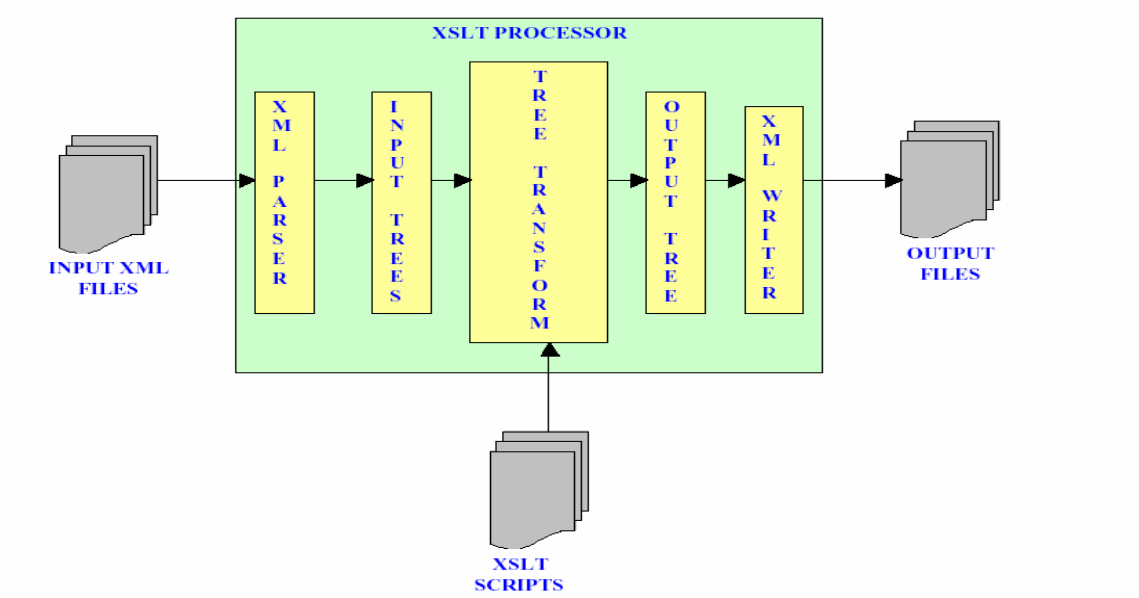

圖 34 使用 XSLT 製作 Code Generator

以下分別解釋上圖之流程：

1. Input XML Files：即爲所輸入之 XML 文件
2. XML Parser：將 XML 文件作 Parsing 並解析成 Tree
3. Input Trees：爲輸入 XML 文件之樹狀表示
4. Tree Transform：將來源 XML Tree 依照 XSLT Scripts 轉換成 Output Tree
5. XSLT Scripts：定義來源 XML 文件格式到目的文件格式之轉換方式
6. Output Tree：輸出文件之樹狀格式
7. XML Writer：輸出 XML 文件至檔案系統之模組
8. Output Files：轉換完成之文件

由上例可以知道，我們可以使用 XSL/XSLT 之技術進行 XML 與 XML 文件之轉換，其實不僅如此，XSL/XSLT 也可以用來將 XML 文件轉換成非 XML 之件，例如轉換成 HTML 是目前在應用上最普遍的作法，在管理網站的文件皆以 XML 格式儲存，但是在伺服端加入 XSL/XSLT，當使用瀏覽器觀看時，網頁伺服器會即時將 XML 文件轉換成 HTML，如此具有文件以 XML 建構的優點，同時又可以最方便使用的瀏覽器來觀看。

使用 XSL/XSLT 也可以讓我們把 PHP Parser 讀入所產生的 XML 文件，再轉成 PHP 程式碼，雖然此種作法的效率比不上傳統編譯器，但仍有開發速度快之好處，我們以下表作一比較：

表 4 使用 XSL/XSLT 與一般 Code Generator 製作的比較

	一般 Code Generator	XML/XSLT Based
優點	■輸入的語言可以用簡潔、易懂的語法 ■程式碼產生器使用如 C++或 Java 等，功能較強大的語言製作	■在程式碼產生規格的部份，很容易修改 ■採用 Script 語法，可以適用快速開發的環境 ■在程式碼產生的架構方面，不需花費心力，可以專注在程式碼之語法或邏輯上
缺點	■語法若改變，需要更動的部份相當多 ■不是使用 Script，也就是說程式碼產生器，必需要先編譯出來，較花時間 ■在建構整個程式碼產生架構上需要花費較多程間 ，例如 tree walking 等邏輯。 ■學習曲線較陡	■輸入語法必需爲正確 XML 格式 ■XSLT 是一特定用途的語言，無法像 C++、Java 般擁有強大的語法及功能，實作上的限制較大

以下爲我們實作 XSL/XSLT 之 php 程式碼產生器所產生的範例：

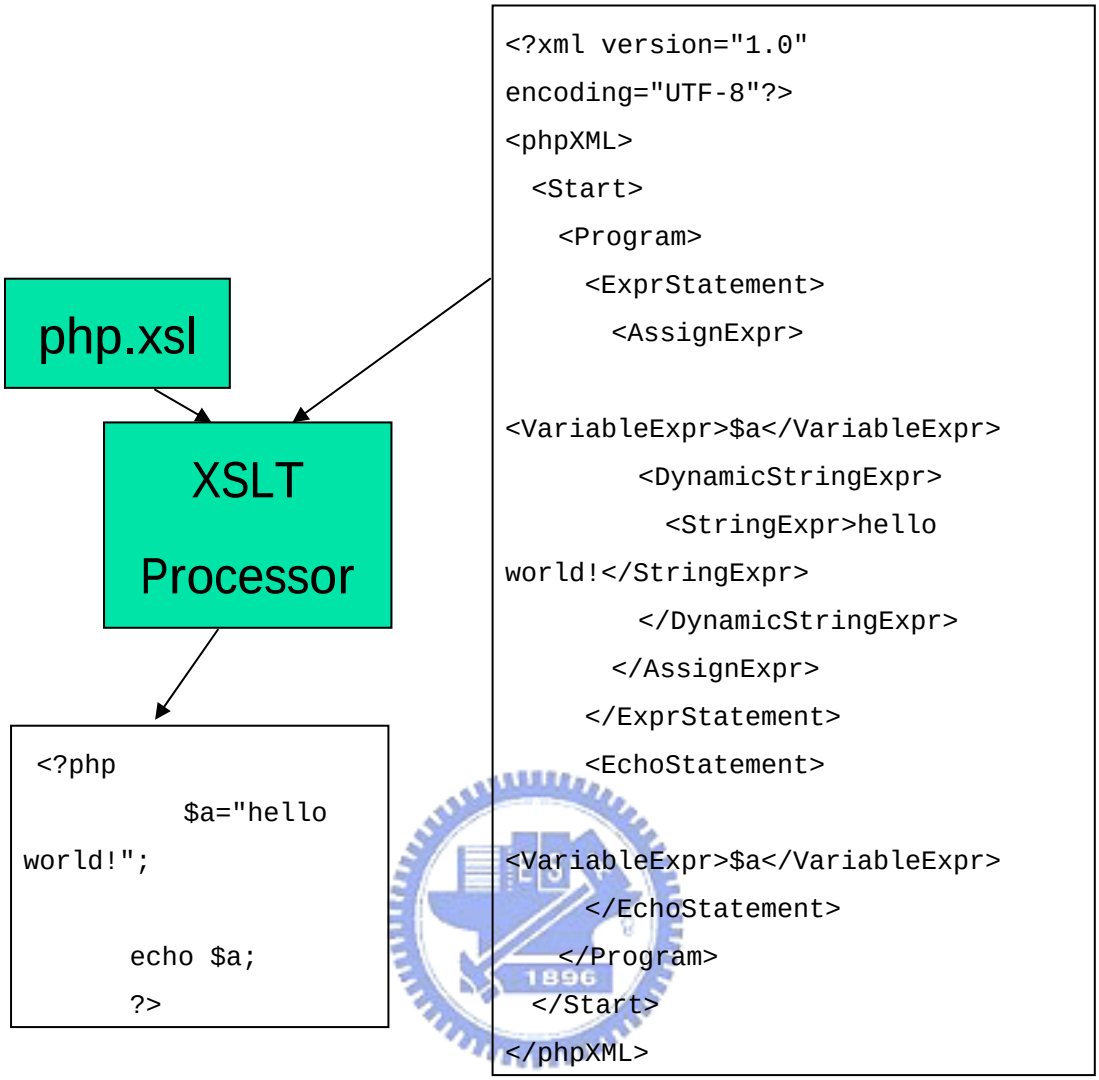

圖 35 XML->PHP 程式碼產生器實作例圖

我們使用 XSL/XSLT 語法建立出 PHP 的 XSL 檔，接著就可以直接採用 XSLT 的處理器，透過 php.xsl 將 PHP 的程式碼產生出來，這種開發方式最大的優點就是快速，因爲只需專注在語法的對應上，不需處理傳統編譯器實作上之各種困難，語法如果有所改變，也僅需修改相關的 XSL/XSLT 定義檔，實作上的時間可以因此而大爲縮短。

五、 應用實例

5.1網站開發應用實例

以下以開發一英語教學網站之使用者登入程式步驟爲例，示範如何使用視覺化建構工具自動產生出我們所需之網站應用程式，下圖爲使用視用視覺化環境編輯並設定使用者介面：

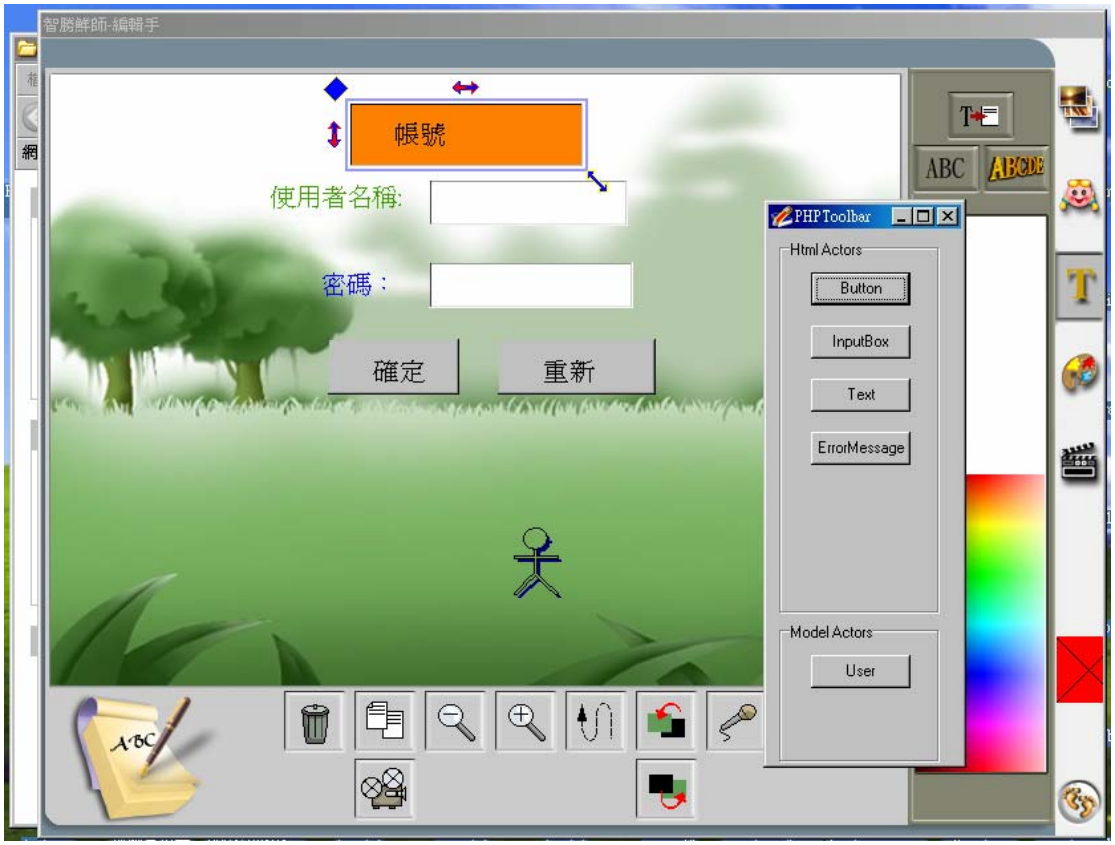

圖 36 使用視用視覺化環境編輯並設定使用者介面

上圖即爲視覺化環境編輯之畫面，其中 OK 及 Reset 兩個按鈕，User 即爲使用者之 model，可以用來作使用者相關之操作，此部份即爲 3.1.3 之第一步驟。

完成編輯後，產生目的碼其中的 HTML 畫面如下圖：

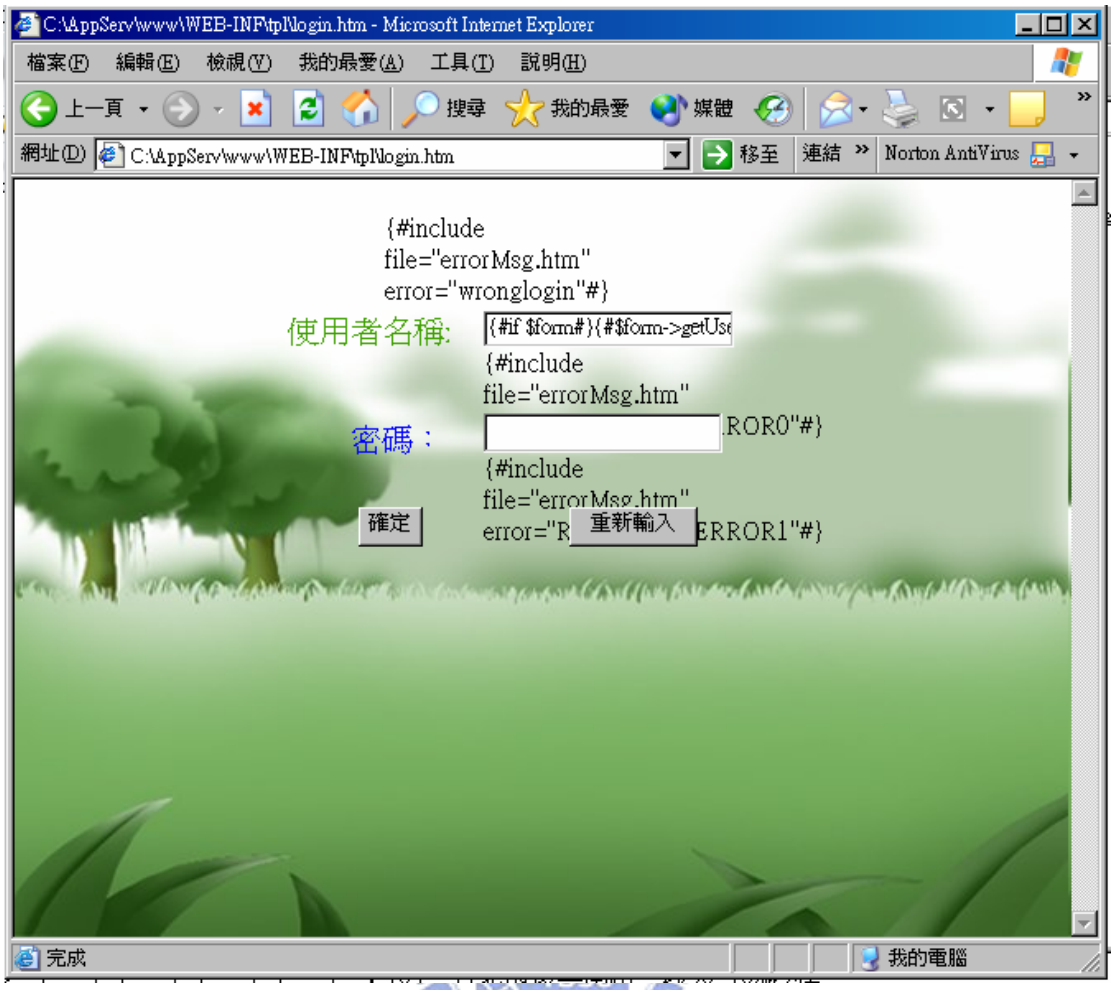

圖 37 產生之 HTML 登入畫面

在產生 HTML 畫面之前，系統即執行 EBS->XForms，再從 XForms->產生 HTML，此部份即爲 3.1.3 之第 2、第 3 步驟。

測試登入錯誤畫面如下：

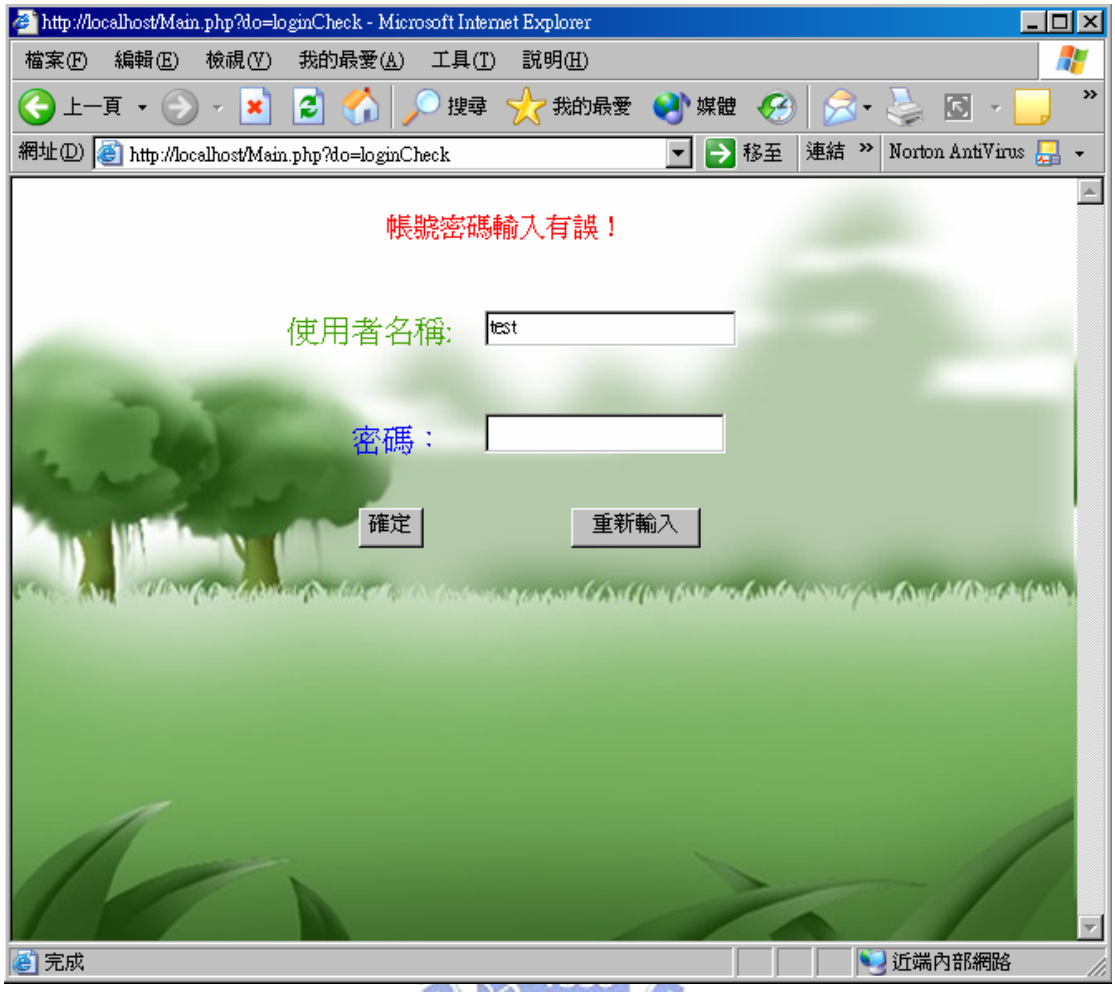

圖 38 使用者登入錯誤之畫面

當使用者登入錯誤後即顯示”登入錯誤！”之訊息，並讓使用者可以再次輸入。

重新修改後之全新登入畫面：

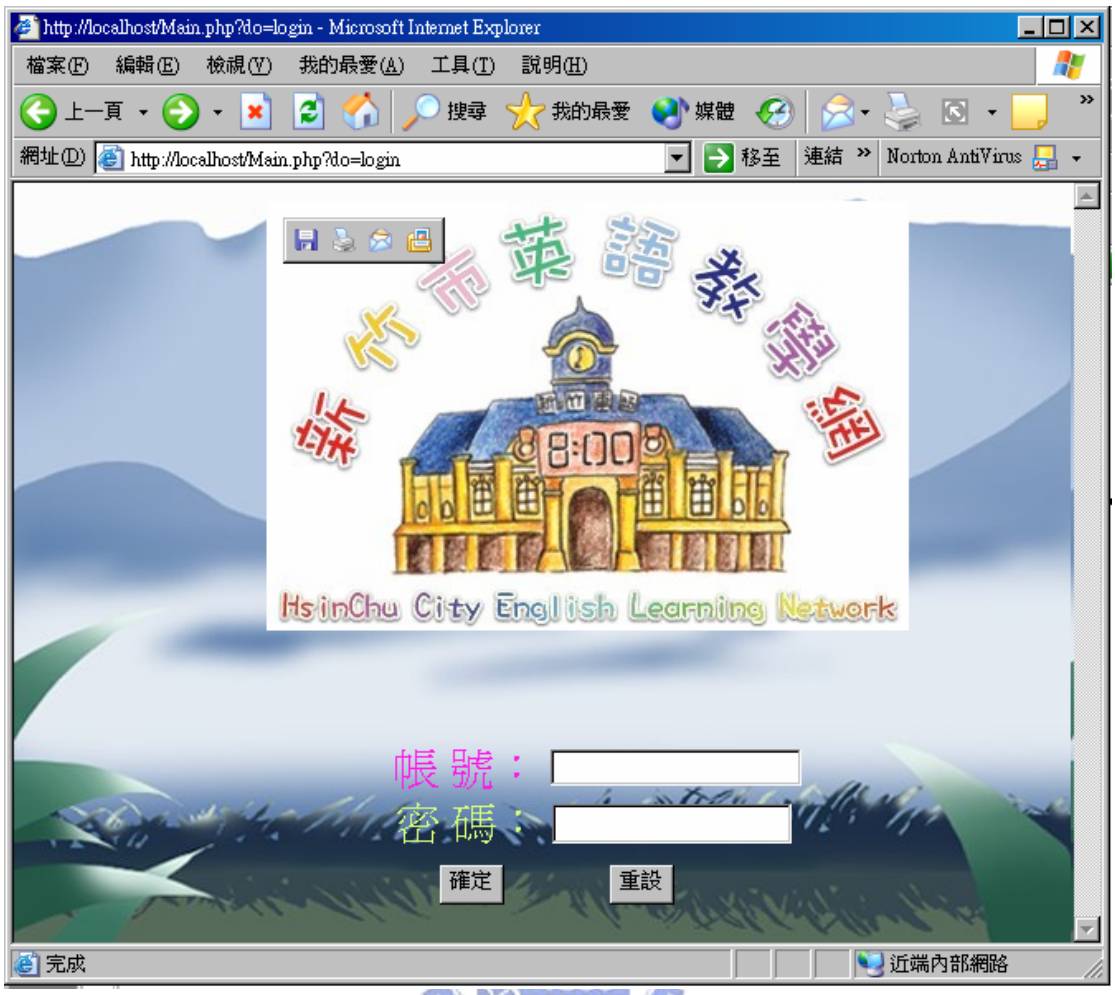

圖 39 修改樣版後之全新登入畫面

由上面 4 個步驟可以快速的完成一使用者登入之程式，UI 設計人員再加以修改美化網站樣版即可上線使用，維護檔案也不會跟程式設計人員有所衝突。

另外，由於爲英語教學網站，故需實作出”雙語”呈現之畫面，此部份我們採用樣版切換技術達成，程式僅需判斷使用者選擇中文或英文顯示，然後取出對應之 HTML 樣版顯示即可如圖 40 爲中文版之數位學習園地，圖 41 則爲英文版之數位學習園地。程式的網址皆相同，只有樣版之切換即可達到”雙語化”介面之目的。

圖 40 中文版之數位學習園地

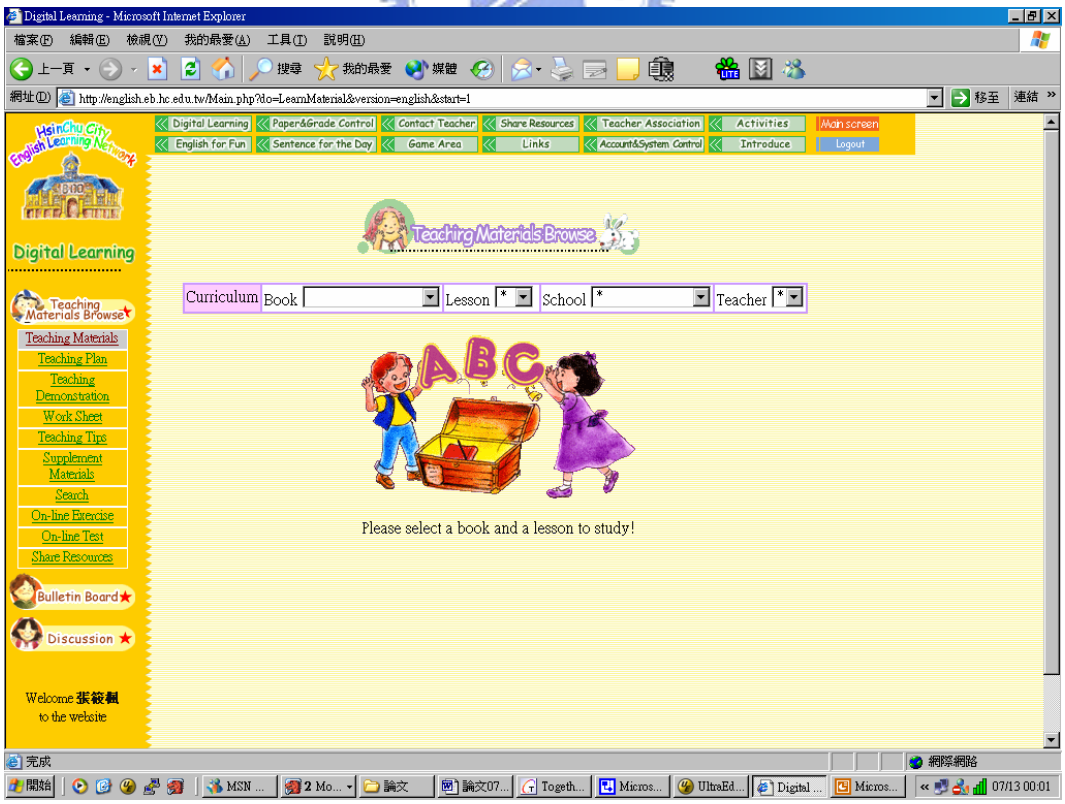

圖 41 英文版之數位學習園地

本網站系統並實作出 44 個模組，以下列舉 6 項較重要之模組：

1. 網站使用者管理模組
2. 活動公告模組
3. 教材上傳管理模組
4. 測驗上傳管理模組
5. 數位學習模組
6. 線上測驗模組

此網站之開發由於採用視覺化方式建構，加上伺服端採用 MVC 架構，開發時間大幅縮短，且後續系統維護，UI 設計人員修改版面，皆較傳統方式改善許多。

5.2 系統評估與比較

首先比較各種類之網站應用程式開發工具，如下表：

表 5 網站應用程式開發工具比較

	開發速度	後續程式及 版面維護難度
一般編輯器	慢	高
無樣板系統之視覺化編輯器	慢	高
有樣板系統之視覺化編輯器	中	低

由上表可以得知有樣板系統之視覺化開發編輯器，較能符合網站開發程式之需求，因其開發速度較傳統方式快，而後續程式及版面維護難度更遠較傳統方式低。

接著我們再比較第二章所分析過之網站應用程式開發系統與本研究設計實作之系統，如下表：

表 6 本研究與現有網站應用程式開發系統之比較

	Dreamweaver	Code Charge Studio	TurboPHP	PHP Editor	本研究
MVC	NO	NO	NO	NO	**YES**
Template	NO	YES	YES	NO	**YES**
資料庫抽象層	NO	NO	NO	NO	**YES**
視覺化 網頁元件	YES	YES	YES	NO	**YES**
視覺化 網站系統元件	NO	NO	NO	NO	**YES**

由表 6 可以清楚看出，除了本研究應用 MVC 設計樣式、資料庫抽象層及實作視覺化網站系統元件外，其餘系統皆未提供相類似之功能，而這些功能正是能讓開發人員以更短的時間建構，以更安全的方式維護網站應用程式之關鍵。

六、 結論

6.1 總結

網站應用程式爲新一代應用程式平台，網站應用程式成長量也以驚人的速度在增加，使用程式語言的選擇也非常多，但多數的整合開發環境(IDE)都不是針對網站應用程式設計的，大多是適用一般應用程式設計的平台。

本研究實作視覺化網站元件，不僅像一般軟體，僅提供 HTML 表單元件，只著重在畫面的設計上，也提供網站模組之元件，事實上程式設計師應該僅就使用者介面的需求作出可用的使用者介面，其他細部調整應該給 UI 設計人員處理。

本研究具體規劃並實作出視覺化網站應用程式的開發環境，網站架構採用 MVC 設計樣式，產生的應用程式有控制元(Controller)及表現層(View)、商業邏輯(Model)的整合框架，能減少程式設計者花費在畫面設計所花的時間，而專注在 Model 的設計。

此外本研究多數的核心語法處理，都採用 XML 爲基礎之技術，如 XForms、XSL/XSLT 等，由於 XML 檔案非常容易使用程式處理、轉換，或是資料交換之用，XML 相關的技術、規格、標準也逐漸成爲重要的實作基礎，所以系統核心的處理採用 XML 的方式，會讓語法未來的修改、轉換等更有彈性。

6.2 未來發展方向

由於尚未有完整之視覺化網站元件的輸入程式，所以網站視覺化元件需由程式碼轉入視覺化編輯系統中，目前僅提供使用者管理之元件，其他元件，如討論區元件、留言板元件等，已有實作之程式碼，但尚未轉入視覺化開發系統中，所以充實擴充系統可重用之網站視覺化元件及建立自動化轉換視覺化元件是未來系統改進重點。

另外，MVC 中之 Controller 所需之 XML 設定檔，應可再改進爲視覺化編輯，如此可以非常容易了解整個網站運作之流程，此部份若完成則整個開發系統真正爲全視覺化開發之環境。

由於系統核心多爲 XML 架構，目的程式碼除了 PHP 外，應該也可以實作出其他平台之語言，如 Java、ASP 等，這也是系統完備後可以再努力的目標。

參考文獻

1. Weiquan Zhao , Jian Chen,“CoOWA: A Component Oriented Web Application Model”, 31st International Conference on Technology of Object-Oriented Language and Systems,p1~p9,Nanjing、China,September,1999.
2. Burnett, Margaret, Goldberg, Adele, and Lewis Ted, “Visual object-oriented programming Concepts and environments”, Manning, Greenwich, 1995
3. 賀元， 賴明宗， 劉燈著，世紀末軟體革命C++,GUI與物件導向理論，傳徵，台北市，民 83
4. Shih-Fang Chuang,”The Design and Implementation of a Visual Language For Scenario Based Electronic Book”, NCTU Taiwan, Master Thesis,2000
5. “Macromedia Dreamweaver MX 2004”, http://www.macromedia.com/software/dreamweaver
6. “Code Charge Studio”, Yes Software, http://www.codecharge.com
7. “TurboPHP Rapid web development”,http://www.turbophp.com/
8. “XForms 1.0”, http://www.w3.org/TR/2003/REC-xforms-20031014/
9. “XForms”, http://www.ws.org.tw/xml_family_show.jsp?TOP_NUM=3&num=8
10. “JavaServer Pages Fundamentals”,http://java.sun.com/developer/onlineTraining/JSPIntro/contents.html
11. Ted Husted,Cedric Dumoulin,George Franciscus,David Winterfeldt,”Struts in Action - Building web applications with the leading Java framework”,Manning,2003
12. John Wildenauer ,”The Model View Controller Framework for PHP Web Applications,” http://www.phpmvc.net
13. Jakarta Struts Web Application Framework http://struts.apache.org/
14. Chorng-Shiuh Koong, “A Component-based Visual Scenario Construction Environment for Non-Programming users to Create Interactive Electronic Books,” NCTU Taiwan,Ph.D. dissertation,2000
15. “SableCC Java parser generator”, http://sablecc.org/
16. “The Extensible Stylesheet Language Family (XSL)”, http://www.w3.org/Style/XSL
17. Chung-Hua Hu , Feng-Jian Wang,”Towards a Practical Visual Object-Oriented Programming Environment: Desirable Functionalities and Their Implementation”,National Science Council NSC 87-2213-E009-002,1998
18. Takao Shimomura,Muneo Takahashi,Kenji Ikeda,Yoshio Mogami,” Web application generator by image-oriented design”, ACM SIGSOFT Software Engineering Notes,Volume 28 Issue 2,p1~p6,March 2003.