國立交通大學

多媒體工程研究所

碩 士 論 文

豐富互動式網際網路應用之
多媒體編輯系統的設計及實作

Design and Implementation of an
RIA-Based Multimedia Authoring System

研 究 生：王乃宣

指導教授：陳登吉　教授

中 華 民 國　九 十 八　年 六 月

豐富互動式網際網路應用之

多媒體編輯系統的設計及實作

Design and Implementation of an
RIA-Based Multimedia Authoring System

研 究 生：王乃宣　　Student：Nai-Hsuan Wang

指導教授：陳登吉　　Advisor：Dr. Deng-Jyi Chen

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science
June 2009
Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

豐富互動式網際網路應用之
多媒體編輯系統的設計及實作

學生：王乃宣　　指導教授：陳登吉 博士

國立交通大學多媒體工程學系碩士班

摘要

全球上網人數持續攀高，散布於網路的資訊每年以倍數成長，瀏覽者的視聽覺感官越來越挑剔，所以能在短時間內編輯出吸引瀏覽者目光的互動多媒體是非常重要的。然而編輯者因工作或求學需要，常在不同時間或地點使用不同台電腦，易發生沒有慣用的多媒體編輯工具(multimedia authoring tools ;MAT) 或無法安裝慣用工具等等的不便，因此，編輯者需要一個具有高度操作性、可攜性及跨平台的工具。

本研究希望結合 Web 跨平台特性，開發一套可快速製作出豐富互動式作品的多媒體編輯工具。但傳統 Web 應用程式開發技術受限於 HTML 規範，不易提供如同單機應用程式一樣優異的操控性、傳統編輯器的設計常限制樣版多樣性、編輯器與樣版持續擴充，下載安裝時間越來越久等問題皆會大大降低使用意願。

本研究設計及實作一套具有高度操作性、可攜性及跨平台的多媒體編輯工具。探討 Rich Internet Application (RIA) 相關技術，協助改善傳統 Web 應用程式反應效率與網頁整頁刷新等人機互動問題，考量系統安全性、擴充性、維護性、可測試性與樣板的多樣性，採用服務導向系統架構與 Composite Application Library (CAL) 程式架構，將樣板與功能獨立模組化，分頁下載，並探討編輯者操作習慣與編輯模式，規劃人機介面與實做智慧型標籤。協助編輯者可隨時隨地使用最新的編輯工具，方便快速編輯出所需的多媒體檔案。

Design and Implementation of an RIA-Based Multimedia Authoring System

Student: *Nai-Hsuan Wang* Advisor: *Dr. Deng-Jyi Chen*

Department of Multimedia Engineering
National Chiao Tung University

Abstract

The internet users rise continuously for past ten years in the whole world, dissemination of information on the internet doubled every year and internet surfers are pickier on the visual sensory. Therefore helping those editors produce multimedia files that attract internet surfers in a short time without too much professional visual design training is very important. Due to the needs of the work or the school, editors often use different computers at different times and places, it lead editors have no familiar multimedia authoring tools (MATs) can be used or installed, etc. Therefore, the editor needs a high manipulated, portable and cross-platform tool.

Combing web cross-platform features to develop a MAT that can easy to create rich interactive creations is an urgent need to do. However, traditional web application development technology is bounded by the HTML specification; it's not easy to provide excellent human-computer interaction (HCI) like desktop applications. The design of traditional authoring tools is often reduce the template diversity. When authoring tools' functions and templates extended, users must spent more time for download and setup tools that will reduce user's desire for use.

Our goal is to design and implement a highly operational, portable and cross-platform multimedia authoring tool. Discuss Rich Internet Application (RIA) relative techniques, refining traditional web application HCI problems like efficiency of reaction and always renew whole page. In addition, in order to upgrade system's security, future extension, maintenance, unit testability and the diversity of templates, we use SOA and CAL architecture to modeling and independent the templates and functions, then user can dynamic download any pages when they want. Finally,

observe editors operation behavior, design user friendly interface and smart tags. Help editors use the latest editing tool to create rich interactive multimedia creations in anytime and everywhere.

誌謝

承蒙指導教授陳登吉老師兩年來耐心的指導與教誨，在此致上對老師無限的感謝。陳老師非常盡心負責，引導著我找到一個令我感興趣的研究方向，除了在研究領域上給予指引及點醒，讓我們學習到作研究的觀念及態度，也時時關心學生們生活上的問題。感謝曾建超、孔崇旭老師細心與不厭其煩的教導。老師辛苦了！敬祝老師：身體健康、平安喜樂。

感謝在交大和我同甘共苦、互相砥礪的同學、學長姐、學弟妹及朋友。可以認識許多來至不同實驗室的成員是我一生的榮幸。在課業上、生活上得以和逸珅、培舜、志軒互相指導、勉勵，也讓我學到很多書本上學不到的東西，充實了心靈。

最後我必須感謝我的家人，如果沒有他們的支柱與栽培，也就沒有今天的我。期望能在未來有好的成就、好的作為，不枉費他們的栽培。

目錄

表目錄

圖目錄

壹、 緒論

1.1 研究動機與目的

研發科技的進步促使桌上型電腦、筆記型電腦的價格下降與輕省筆電 (Netbook) 的興起，越來越多人同時擁有一台以上的電腦。也因生活上工作、求學等等的需求，經常會在不同時間地點接觸不同台的電腦。在不同作業系統的電腦環境裡使用傳統單機應用程式，經常會遇到一些不方便。例如：自己的電腦送修，到朋友家借用電腦。當須要使用到自己慣用的多媒體編輯工具時，發現朋友的電腦上並沒有安裝該套軟體，而慣用的多媒體編輯工具安裝軟體又因版權問題，在網路上並不容易可立即取得，馬上安裝在朋友的電腦上。當經過一番折騰，好不容易取得慣用的多媒體編輯工具安裝軟體時，卻發現自己與朋友的作業系統並不相同，無法順利將軟體安裝在朋友的作業系統上。或者是安裝成功後，卻發現不同作業系統版本的軟體，編輯介面不太相同等等的困擾。

反觀以開發者的角度來探討。現今業者會先針對 windows 作業系統開發完應用程式後，再繼續針對其他作業系統重新設計與撰寫同功能版本的應用程式。例如：由於 EeePC 蓬勃發展，為了搶得市場，於是將 Windows 上的程式移植至 Linux 平台上。以這種平台移植技術重新撰寫程式的開發方法，會出現許多困難，包含異平台程式碼同步更新、維護。再則，軟體上市後，隨著需求持續做更新，當有新的版本時，使用者勢必再次針對各台電腦分別下載安裝，由於太頻繁的更新會大大降低使用意願，因而也限制了開發者對於軟體更新的次數，導致使用者無法常常立即享有最新功能。為解決上述問題改採用以傳統網際網路應用程式技術開發系統，但因受 HTML 等網頁呈現規範限制住，系統並無法和單機程式一樣，提供流暢的人機互動與優越的感官視覺化效果，另外當檔案太大，下載時間太久，也會降低使用者持續使用意願。因此，如何開發出具安全性、擴充性、維護性、可測試性、可跨平台、可快速下載呈現、高流暢人機互動的系統來滿足這群使用者是一研究難題。

再加上由 Internet World Stats 取得全球上網的人數統計 [1]，年年持續攀高，如下圖所示，散布於網路上的資訊每年以倍數成長，瀏覽者的視聽覺感官也越來越挑剔。網站上必須放置豐富聲光視覺化效果的互動多媒體來抓住瀏覽者的目光。但做出吸引瀏覽者目光的互動多媒體檔案需要高專業視覺化效果、動畫設計知識與對製作工具的高熟悉度。未經過受訓的人很難在短時間內做出合適的作品。因此，提供一套協助使用者操作快速上手、方便快速編輯出富含高互動與豐富視覺化多媒體的編輯工具 (Multimedia Authoring Tools; MATs) 是非常重要的。

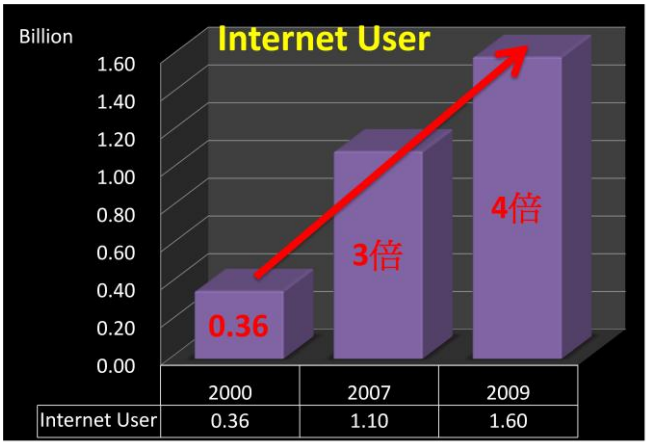


圖 1：全球上網人數統計

綜合上述問題，本研究擬提出『豐富互動式網際網路應用之多媒體編輯系統的設計及實作』，希望能設計及實作出一套可跨平台跨瀏覽器、操作互動流暢、易上手、編輯方便快速的多媒體編輯系統，以滿足廣大使用者與瀏覽者的需求。在設計當中並考量到系統安全性、擴充性、維護性、可測試性與樣板的多樣性，系統架構採用 Service-oriented architecture (SOA) 、程式架構採用 CAL 開發，將樣板與功能模組化分頁下載，提升樣版多樣性與使用者持續使用意願。

1.2 研究方法

本研究所設計與實做的 EzShow 系統必須考量到人機介面設計、系統架構的彈性與安全性，及編輯功能與樣板的擴充性。因此必須先了解現有網路應用程式開發技術、使用者編輯模式、使用者編輯習慣、系統架構設計與規劃、程式架構設計與規劃，進而規劃出整個 EzShow 系統架構及細部實作需求，最後進行作與展示。初步可以分為下列方法步驟：

- 相關背景研究及文獻探討:

收集關於程式開發技術的演進、探討能跨平台與跨瀏覽器的網際網路應用程式開發技術、使用者編輯模式的比較、使用者編輯流程的習慣、系統人機介面的設計、彈性且安全的系統架構、富含擴充性的程式架構等領域的相關研究與文獻探討，並分析比較不同技術或方法之間的優缺點。

- EzShow 系統分析與設計:

一個資訊系統要能真正成功，最重要的關鍵要素在於使用者需求的瞭解與需求項目的明確敘述與確認，於是將需求分成功能性需求(Functional Requirement)與非功能性需求(Non-Functional Requirement)個別做討論，前者包含 EzShow 系統必需提供的功能，包含操作介面、操作方法、功能限制、

技術性規格、技術性規範或限制等部分。後者包含：系統功能是否達到期望的要求、系統擴充性、維護性、使用的環境與工具等。

- EzShow 系統架構與實作:

在了解系統功能與非功能需求後，依需求選定開發技術與工具，並設計出具彈性與安全性的系統架構。在進行程式撰寫的同時，特別考慮到系統功能與新樣版的擴充性。此章節會詳細介紹整個系統架構包含元件與元件之間的互動，以及系統各模組詳細的實做方法，並配合範例說明程式架構與樣版的設計理念與系統易上手、方便快速編輯的特性。

- EzShow 應用範例展示:

在完成整套 EzShow 實做後，配合在 Windows XP 搭配 IE8 瀏覽器與 Mac OS 10.5.5 版本搭配 Safari 瀏覽器作為系統可在異平台與不同瀏覽器上呈現具有一致性效果，並展示一般 Web 應用程式較不易做出的互動式多媒體效果。

1.3 研究相關名詞

本研究相關名詞及參考範圍，敘述如下：

- 程式移植 (Program Porting) [2]

將來源平台的程式移轉到目標平台，使來源平台之程式，能夠在目標平台上正常使用。由於來源平台與目標平台上可使用的系統資源不同、程式語言的語法不同，故移轉的過程中需要一些改變或相關對應的技術。

- Service-Oriented Architecture; SOA [3] [4]

服務導向架構 (Service-oriented architecture) 是一種分散式系統架構。服務導向是指架構在應用程式上之服務，其作業系統、程式語言和其他技術等鬆散的耦合。採用開放的標準，如：XML 格式，將應用程序功能作為服務發送給最終用戶或者其他服務，提供網路服務單位建構一個具彈性、可重複使用的整合性介面，促進各相關用戶完美的溝通，盡速達到網路服務提升的目標。

- Model-View ; MV [5]

在 Model-View 設計中，將畫面呈現與資料邏輯分開實做。View 負責畫面的呈現，而 Model 則是與畫面無關的資料、邏輯控管。

1.4 研究範圍與限制

本論文討論豐富互動式網際網路應用之多媒體編輯系統的設計及實作。研究範圍大致可以分為三大方向加以描述。在選擇開發技術方面，涵蓋目前最流行的 RIA 開發技術，含 Microsoft Silverlight™ [6]、Abode Flex™ [7]、Sum JavaFx™ [8] 。在人機互動設計方面包含使用者編輯模式、使用者操作介面習慣。在開發實作方面使用 「WPF™ 的複合應用程式」 程式架構。

- 「WPF™ 的複合應用程式」程式架構 [9]

  隨著各功能組件的數量以及多樣性樣板的增加，需要掌控的專案因素會變得加倍複雜，仍無法完全解決建置可維護性應用程式時所遇到的老問題。因此本研究選擇採用複合應用程式的開發模式，讓各功能模組可在相同的應用程式中，彙總來自不同後端系統的資料，隨著系統需求的變更，新模組新增至系統時所面臨的阻力會比在無模組化的系統中來的更少。現有的模組可以更獨立地發展，因而提升可測試性。模組可由不同團隊來開發、測試及維護。

由於本研究所討論的應用程式為 Web-based 應用程式，研究限制為使用者的電腦必須具備上網功能，否則無法進行編輯器下載與資料溝通等動作。另外，使用者也必須先安裝基本執行環境，瀏覽器，如：IE、Firefox、Safari 等，瀏覽器需已完成安裝 Microsoft Silverlight™ player runtime2.0 版。

1.5 章節概要

本章介紹研究動機與目的，希望編輯系統能突破傳統網路應用開發技術的限制，採用視覺化樣版替換編輯機制與實做智慧型標籤，協助編輯者能快速上手立即編製出所需多媒體檔案，期許能在不同作業系統與瀏覽器上執行。

第二章，介紹相關研究議題，包括網路應用程式開發技術演進、多媒體編輯模式文獻探討、多媒體內容之物件屬性設定方法的介紹與比較。

第三章，進行系統功能性與非功能性功能需求的評估。功能性需求包含系統功能規劃與客戶端應用層介面功能需求，非功能需求包含系統架構與程式設計。

第四章，先介紹 EzShow 編輯系統的整體架構圖再分別介紹 Server 端與 Client 端編輯器的架構圖。

第五章，更詳細介紹整個 EzShow 系統架構與系統功能的實作。

第六章，EzShow 系統操作流程與實例展示。

第七章，結論與未來展望。

貳、 背景研究及相關文獻探討

網際網路上的資訊每年以倍數爆增的時代來臨，網路服務開始重視使用者的經驗與感受，豐富視覺化效果的介面設計，以及高互動性的網站設計已是一股新趨勢。本研究所設計與實做的 EzShow 系統必須包含四大特性：跨平台跨瀏覽器、操作互動流暢、易上手、編輯方便快速，著重於探討關於互動多媒體編輯系統相關開發技術與人機介面互動分析。

什麼是多媒體，多媒體以定義來說結合多種視聽覺感官和表現方式，以傳達訊息的一種技術，這些表現包含文字、圖形、圖像、聲音、動畫和視訊等，一種人機互動式訊息交流和傳播媒體。由於人類比較容易接收聲光影像的訊息，因此，利用多媒體來做為溝通工具，可提高目標對象的興趣，達到更有效率的溝通。多媒體本身的交談設計性和多類型傳播媒介特質，將可以提高溝通的效率，大幅降低所需的溝通時間。多媒體不同於傳統媒體的地方，最重要的一點在於它的交談性。資訊的流通，不再只是單方向的傳播或接收，而是一種雙方互動性的關係[10]。

目前的多媒體應用層面範疇很多，許多性質不同的業者，都可以找到他們市場的利基。如廣告、教學或遊戲用途。利用多媒體網頁將廣告變成有聲有色甚或是互動形式來吸引顧客，更能在第一時間提供顧客即時的消費資訊、產品資訊，但限制是如果多媒體內容檔案太大以致於下載時間較長，顧客容易失去耐心，針對此點，本研究在系統與分析將會提出改善協助。另外，利用多媒體作為教學用途，可以增加自學過程的互動性，吸引學生學習、提高學習興趣，及利用視覺、聽覺及觸覺三方面的回饋(Feedback)來增強學生對知識的吸收。

因此在開發本系統之前，我們會先去探討如何才能達到跨平台跨瀏覽器、操作互動流暢等特性，其相關研究包含單機應用程式與網路應用程式兩者之間的優缺點與近年來網際網路應用程式的發展演進相關文獻，其中會介紹到什麼是 Rich Internet Application (RIA) [3]，以及提到幾個當紅的 RIA 開發技術，例如：Microsoft Silverlight™、Abode Flex™、Sum JavaFx™，並嘗試分析他們之間的優缺點。在探討完關於跨平台跨瀏覽器、操作互動流暢特性的程式開發技術後，接著要探討系統的人機介面與操作流程要如何設計才能協助使用者達到易上手與編輯方便快速的特性，其相關研究包含使用者編輯模式的比較、使用者編輯流程的習慣、系統人機介面的設計，並分析比較不同方法之間的優缺點。

2.1 MAT 開發技術選擇

開發技術的選擇重點在於選擇一系列可以幫助完成程式開發工作的關鍵技

術。選擇對於專案開發團隊合適的開發工具可以有助於提升開發效率與軟體品質。

2.1.1 應用程式發展演進

首先針對傳統單機應用程式與網路應用程式開發及使用環境做深入的比較與探討。

1. 傳統單機應用程式：

通常傳統單機應用軟體在執行程式時可依賦予權限存取使用本機上所有允許的硬體資源。開發一般單機應用程式，通常會將所有的程式與資料一併安裝到同一台電腦之中，因此程式在運作時會使用同步的方式與底層的資料進行溝通。傳統單機應用程式比起傳統網路應用程式的優勢在於：

- 可提供豐富的使用者介面
- 可提供高流暢的人機互動
- 容易開發，提供開發者較高的生產力
- 程式反應時間較快

傳統單機應用程式比起傳統網路應用程式的限制在於：

- 軟體的功能需於客戶端執行，程式的執行體積大
- 使用者必須針對不同台電腦個別安裝與更新
- 無法跨平台，程式不容易開發與部署，程式移植工作繁複
- 程式更新困難，版本維護困難
- 客戶端必須要安裝特定的執行環境才能執行相關應用程式。Ex:加裝符合的 .NET 執行環境才可順利執行 Windows Application

由上述討論我們得知單機應用程式無法跨平台造成了使用者在使用上許多的不方便，接著開發者轉向開發網路應用程式。

2. 傳統網路應用程式：

不論在什麼平台上，只要有一個簡單的瀏覽器即可執行存放於伺服器上的各種應用程式功能，而其運作上幾乎完全都是透過網路來進行。客戶端在執行功能時是使用非同步的方式來呼叫伺服器上的功能。具無狀態 (Stateless) 特性，指的是瀏覽器是一個很單純用來呼叫遠端伺服器功能的程式，因此除了一些記錄帳號密碼的 cookie 之外，瀏覽器並不會存放任何資料在本機上。其優勢在於：

- 容易部署:軟體的功能完全存在於伺服器端，客戶端完全不需要進行任何安裝與部署的工作
- 容易管理與更新:伺服器的功能有所改變也不需要改變客戶端的程式。這種特色使撰寫需要大量部署的程式變得非常容易，客戶端隨

時都能夠使用置放於伺服器上的所有最新功能，網路管理員完全不需要煩惱在程式有所更新時，需要重複替換客戶端程式的作業，因此部署管理成本能夠幾近為零。

- 客戶端不必安裝特定的執行環境才能執行相關應用程式。Ex:客戶端無須加裝.NET 執行環境

2.1.2傳統網路應用程式限制

傳統網路應用程式呈現的幾個主要問題包含以下 [11] [12]：

- 必須要在有順暢網路的環境下才能夠執行軟體的功能
- 互動性較差的人機介面: 由於 HTML 規範等因素只提供一些簡單基本的控制元件導致在人機互動性與使用者介面上的開發限制眾多。瀏覽器除了能夠顯示基本的文件、圖片，一旦需要更多樣化、更複雜的操作介面與多媒體聲光效果時，瀏覽器本身無法提供足夠的支援。
  - Hyper Text Markup Language (HTML) ，由 Tim Berners-Lee 設計，為網頁創建和其它可在網頁瀏覽器中看到的信息設計的一種標示語言。IETF 用簡化的 SGML（標準通用標示語言）語法進行進一步發展出 HTML，由全球資訊網聯盟（W3C）維護。為結構性標記語言，使用標籤像是 <html></html>來結構化文章的段落、標頭、排列、超連結等等。詳細的 HTML 的文法定義於 W3C 組織文件下 [13]。
- 開發方式較為複雜: 在設計上需考量到使用者端環境的差異，包含不同平台、版本的瀏覽器，因此在設計網頁介面上會有許多限制。
- 處理過程的問題： 瀏覽器只負責網頁的顯示，而使用者在網上的操作皆需回傳至伺服器處理，當在伺服器端完成工作後再產生反應顯示在網頁上。在伺服器與使用者之間網路傳遞資料的一來一往之間會消耗大量的網路頻寬，加上若資料的傳輸量太大會導致程式的回應速度會極為緩慢。且複雜的網路應用程式通常完成單一個工作會需要使用者經過一連串網頁設定，使用者操作畫面設計受限。例如：當你在訂機票時，會經過一連串頁面的設定才有辦法訂到機票。
- 資料呈現方面的問題: 未支援更流暢的互動瀏覽機制。通常使用者必須輸入關鍵字去搜尋它們所需要的資料。如何可以提供一個高互動流暢性，而且更有效率的方式降低將資料呈現給使用者的方法。
- 等待結果回傳的問題: 傳統網路應用程式當使用者對網頁送出要求 (request) 後，就必須等待整面網頁重新整理 (refresh) ，

接收整頁新回傳的網頁，造成使用者和傳統網頁之間的互動相當有限。

雖然利用 AJAX 非同步訊息傳遞(Asynchronous Message Passing)的溝通方式，在送出訊息後 sender 仍可以繼續執行其他工作，若有回應依 sender 需求傳回，sender 及 receiver 可以並行執行它們的工作，讓我們在網路上瀏覽各種頁面時，不會因為傳統網頁同步訊息傳遞(Synchronous Message Passing)特性的問題，受限於與後端伺服器一再進行 Request - Response 溝通 [14]。但仍受限於 HTML 框架限制無法做到和單機應用程式一樣的豐富視覺化使用者介面與更流暢的人機互動。

因此，在 2002 年 3 月 Macromedia 公司在發表的一份白皮書，提到了關於 Rich Internet Application (RIA) 這詞彙，其特性同時具有傳統單機應用程式與網路應用程式的優點，並改善舊技術的限制。在更早期，與 RIA 有某些相同含義的網路應用技術概念就已經被一一提出。例如：微軟於 1998 年左右提出遠程腳本、Forrester Research 於 2000 年 10 月提出 X Internet、富客戶端(Rich client)等 [15]。隨著近年網路通訊技術的成熟與快速發展，網路服務的設計需重視使用者經驗與感受、豐富且視覺化的介面設計以及高互動性已是一股新趨勢。

2.1.3 RIA 介紹

RIA 的全名為 Rich Internet Application，是一種具有近似於傳統桌面應用程式使用者介面豐富、高互動性和網路跨平台特性，有別於傳統 HTML 框架式的設計，利用向量式動畫、多媒體以及資料庫的結合，來開發出新一代的網站體驗。

- 『RIAs introduce features and functionality of traditional desktop applications like animations and client-side computing to Web applications. The advantages of such Web applications include complex user interactions and the overcoming of page loading requirements of traditional Web 1.0 applications. Enterprises are rapidly adopting Web 2.0 features like RIAs as they see high business value in the innovation.』 [16] [17]

下圖為 RIA 的行為模式 [18]：

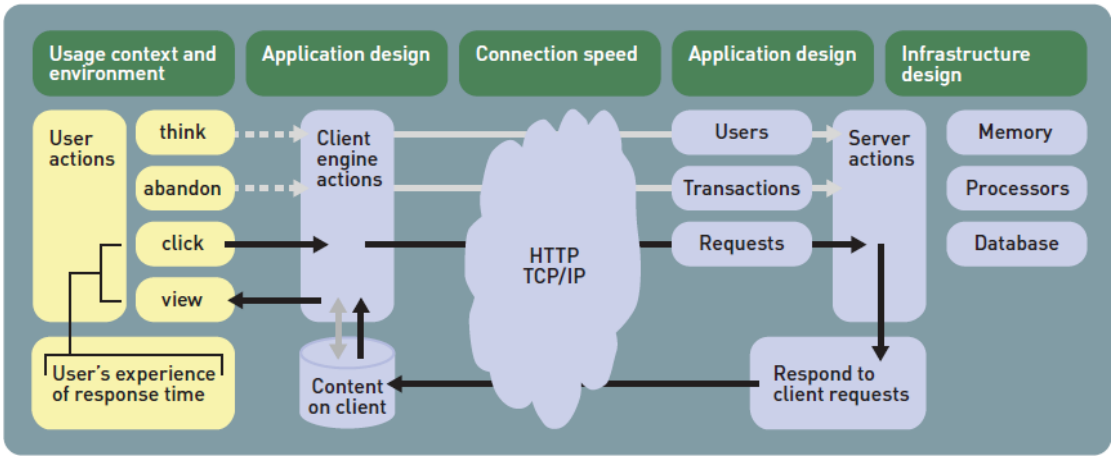


圖 2：RIA 行為模式

上圖左邊為使用者端，使用的內容和環境、使用者的動作等。最左邊為伺服器端，包含伺服器硬體規格。而應用程式在圖中間兩旁的部分，其特色為視需求將應用程式分成 Client 端與 Server 端，兩者之間靠網際網路傳遞訊息。這樣設計模式開發的 RIA 應用程式有以下特性[19] [20] [21]:

- **較多變化:**

相較於傳統網頁應用程式，RIA 的特色為採用非同步傳輸，使用 Client 端引擎與 Server 端互動，而不用等待使用者下達命令，可以提供更多像桌機應用程式一樣的高操作互動性與豐富使用者介面，可做較多複雜的行為。ex: 動畫、特效。

- 『These old-fashioned Web applications are being replaced by the so-called Rich Internet Applications (RIAs) which provide richer and more interactive user interfaces, similar to desktop applications.』 [19]
- 『RIAs typically feature asynchronous communication, in which the client engine can interact with the server without waiting for the user to perform an action such as clicking on a link. This increases responsiveness 』 [20]

- **易使用:**

擁有跨平台與瀏覽器的特性，使用者可做到免安裝、快速更新升級；開發者容易開發且可以降低開發時間與成本，不用再辛苦的針對各平台做程式移植動作。

- 『An RIA typically, though not always, runs within a browser and doesn’t require software installation on the client.』[20]

- **較安全:**

為了提升安全性，很多的 RIA Client 端應用程式跑在 Sandbox 環境上，這樣的執行方式比起直接在作業系統上直接執行來的更安全些。

- 『For security, most RIAs run their client elements within a sandbox.』 [20]

- **較有效率:**

RIA 提供一種新的 Client-Server 架構， Client 端處理使用者介面相關動作，只有當與 Server 處理有關的動作，如交給 Server 處理、儲存、要資料等 Client 才和 Server 互動，使用更聰明的非同步傳輸方式，將系統設計成 Client-Server 間只需傳遞少部分必要性資料，好處為節省網路傳輸頻寬、改善系統反映效率及讀取資料時網頁需重新整頁刷新等等傳統網路應用程式的限制。

- 『RIAs generally have clients handle user-interface-related activity, while the application server processes and stores data and streamlines data updates sent to the client. This frees server resources, allowing the same hardware to handle more client sessions concurrently. It also reduces client-server traffic, resulting in better performance.』 [20]

- 『RIAs provide a new client-server architecture that reduces significantly network traffic using more intelligent asynchronous requests that send only small blocks of data.』 [21]

近期微軟也積極投入 RIA 相關技術研發。將 RIA 縮寫重新定義為 Rich Interactive Application，與 Adobe 提出的 RIA 只差別在中間 I 字的縮寫，Internet 與 Interactive。

2.1.4 RIA 開發技術討論

目前 RIA 相關開發技術有許多套，我們將針對 Microsoft Silverlight™、Adobe Flex™、Sun JavaFx™深入了解，並嘗試分析他們之間的優缺點以及是否適合本研究開發使用。有關三者的技術比較如下表所示。

表 1：RIA 開發技術比較： Silverlight、Flex、JavaFx

項目 \ RIA技術	Silverlight (Microsoft)	AIR (Adobe)	JavaFx (Sun)
跨平台與瀏覽器 操作互動流暢 (UI呈現: 動畫、影音...)	RIA 特性 皆支援		
程式開發與既有技術的整合 (Javascript、DOM、AJAX、CSS...)	●	△	△
開發語言的選擇性與熟悉度	●	△	△
後端整合系統容易度與熟悉度	●	△	△
執行效能 InfoWorld評比分數排名	1	2	2
開發技術 InfoWorld評比分數排名	1	2	3
開發工具 InfoWorld評比分數排名	1	2	3

能力強度: ● > △　　※ InfoWorld 測試日期：依技術或工具發表而定 2008/4 ~ 2009/2

由上表所示，三者皆為 RIA 相關技術皆支援跨平台與跨瀏覽器與提供人機操作高互動流暢性。在程式開發與既有技術的整合這點是指 silverlight 本身以 DOM (Document Object Model) 型式公開其元件樹 (element tree) 。

- 文件物件模式 (Document Object Model ;DOM)

  『The DOM is a standard application programming interface (API) for HTML and XML documents that defines the logical structure of documents and the way a document can be accessed and manipulated. 』[22]

只要透過最原始單純的 javascript 就能夠讀取 silverlight DOM 內容，直接操控 Silverlight 現有元素的屬性或是動態新增元素，搜尋引擎可輕易爬文解析 (Crawl) ，而不需其他介接橋 (bridge) 或演算法。而 AIR 中 flash 技術所設計的內容是無法被搜尋引擎解析。actionscript 與 javascript 溝通需要外部溝通介面來協助。

另外 InfoWorld 組織 [23] 也針對上述技術與主要開發工具做出測試評比，如表 2、表 3 所示。

表 2：InfoWorld 對 RIA 技術比較： Silverlight、Flex、JavaFx

	開發技術 / 開發工具	Score	Capability(30%)	Ease of development(30%)	Documentation (15%)	Performance (15%)	Value (10%)
A	Silverlight 2	9.3	9	9	9	10	10
B	Adobe AIR 1.5	9.3	9	10	8	9	10
C	Sun JavaFX 1.0	8.7	9	8	9	9	9

由表 2 得知 silverlight2 的執行效率比 Adobe AIR 1.5 好，而 Adobe AIR 1.5 的發展容易度又比其他兩者高。由表 3 得知，VS2008 開發工具又比其他兩者評分來的高。

表 3：InfoWorld 對 RIA 工具比較： Silverlight、Flex、JavaFx

	開發技術 / 開發工具	Score	Capability(30%)	Ease of development(30%)	Documentation (15%)	Performance (15%)	Value (10%)
1	VS 2008 Professional and 2008 Team Suite	9.3	10	9	9	9	9
2	Adobe Flex Builder 3.0 Professional	8.9	9	9	8	9	9
			Features40%	Ease-of-use20%	Integration20%	Performance 10%	Value 10%
3	NetBeans 5.0	8.1	8	9	7	8	9

silverlight 可和.NET 平台技術整合，可和程式語言：ASP.NET AJAX、C#、VB、標記語言：Extensible Application Markup Language (XAML) 、資料庫語言：LINQ、建立和執行分散式系統間連線的技術：Windows Communication Foundation (WCF) 、使用者介面技術：Windows Presentation Foundation (WPF) 等等技術做整合，而且有微軟雲端平台、Digital Rights Management (DRM) 等機制做配合，故 silverlight 的開發、維護與擴展方面在未來具有很大的優勢。

三者各有其優勢，在本研究先期背景研究階段，JavaFx 仍尚未正式釋出穩定版本，所以無法將其納入開發技術選擇的考慮。本研究採用 Silverlight 搭配 ASP.NET、AJAX、LINQ to SQL、WCF 等相關技術進行開發。

什麼是 XAML?

XAML 是微軟開發的一種 XML 的使用者介面描述語言，與 HTML 類似，用來描述使用者介面。可以定義文件、圖像、控制元件的佈局、2D 和 3D 物件、旋轉（rotations）、動畫（animations），以及各式各樣的效果。WinFX XAML Browser Application（XBAP）是用 XAML 作介面描述，在瀏覽器中執行的程式，可取代過去的 ActiveX、Java Applet、Flash。

本質上屬於一種.NET Programming Language，於共同語言執行時期（Common Language Runtime）與 C#、VB.NET 等同。

2.2 多媒體編輯模式

關於 MAT 在操作上的編輯模式考量包含兩大重點

1. 編輯製作過程：如何能協助使用者快速製作出具故事情境的多媒體
   - 元件式排版
   - 樣板式替換
2. 操作介面設計：如何協助使用者易找尋在那裡可設定元件的屬性
   - 工具列設定
   - 步驟式設定
   - 智慧型標籤

本節將先針對編輯製作過程做探討，2.3 節再介紹操作介面設計。

2.2.1 元件式排版

想做出吸引瀏覽者目光的多媒體檔案需要高度專業視覺化效果設計知識與對製作工具的高熟悉度，未經過受訓的人很難在短時間內做出合適的作品。多媒體製作者在編輯多媒體檔案時，須先根據檔案內容收集相關的多媒體素材，針對各個素材元件設定相關的屬性和動作，接著編輯各素材的呈現順序，當編輯完畢後，製作者預覽編輯後的結果，若對其中某個環節不滿意，則需重新修改直到滿意為止 [24] 如下圖所示。在整個製作過程當中，需花很多時間在重複性動作。

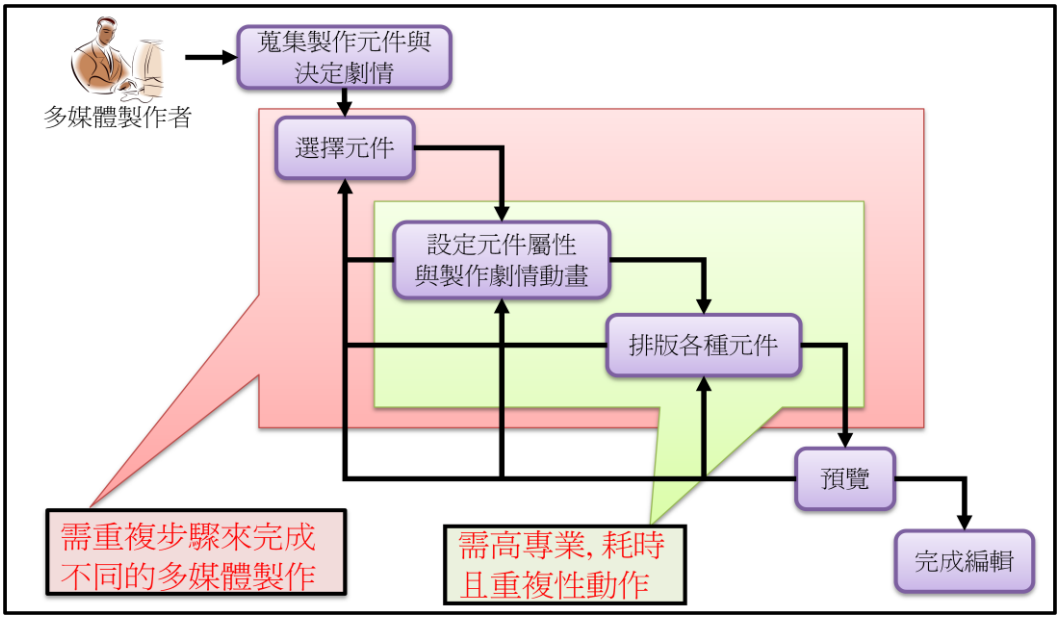


圖 3：元件式排版操作流程

2.2.2 樣板式替換

樣板式多媒體內容 [25] [26] 的定義是具有劇情的多媒體內容，其內容組成可由多種多媒體物件組成。而基於他們的設計用途以及有些同類型的多媒體內容編排方式的固定性，在設計內容時可以用樣板形式來設計。可以節省需要進行重複性動作所耗費時間，且不需要相關專業能力也可以做出來視覺效果酷、炫且含劇情動畫、互動式的豐富多媒體。樣板式替換編輯模式如下圖所示：

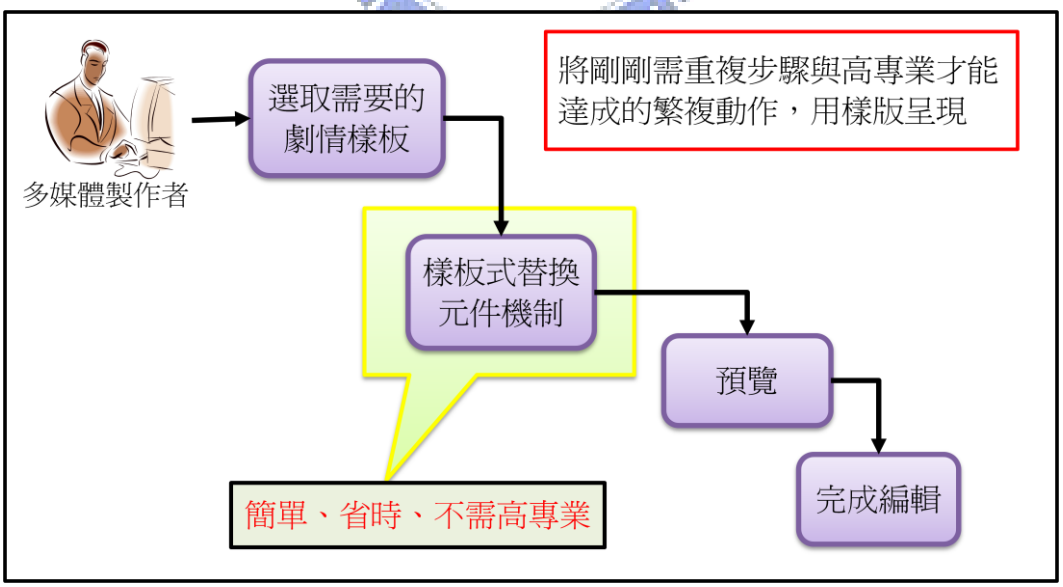


圖 4：樣板式替換操作流程

首先經由於專業人士，將常用的編輯習慣作成可重複使用的樣板，製作者在製作新多媒體時，只要將既有的樣板拿來重用，針對內容作出修改，而不需要從

頭開始擺放圖片、設定動畫特效等等。省去這些重複的編輯動作，也能達到同樣的呈現效果。

2.2.3 多媒體編輯模式討論

經過了解元件式排版與樣板套用替換編輯模式後，接下來比較兩者的特色，如下表所示：

表 4：元件式排版與樣板套用替換編輯模式討論

特性比較	樣板式替換	元件式排版
整合性	優(多功能)	缺(單功能)
操作複雜度	簡易	複雜
多媒體製作速度	快	慢
快速劇情編輯	可	無
操作角度 (View)	角色操作(Top View)	元素操作
操作彈性	普通	優
製作成本	低	高

由上表中的比較可以得知，樣板式替換比起元件式排版，使用樣板套用機制將能夠簡化多媒體教材的開發流程、使用視覺化的編輯方式將可降低操作的困難度、可快速編製多媒體檔案、提供劇情套用、操作成本較低但操作上較無彈性。本研究為了協助新手也能快速製作出所需多媒體檔案，所以在本研究中引入樣板套用的機制，使用樣板套用的編輯模式來取代傳統元件式排版編輯模式。

2.3 功能介面展現方式

在討論完關於 MAT 系統在操作上的考量第一點關於編輯製作過程後，接下來探討第二點，關於功能介面設計，如何協助使用者可以容易找尋到其元件屬性設定介面位置。

人機介面一般指介於使用者與硬體之間，設計使用者與硬體互動溝通相關的軟體，目的在使得使用者能夠方便有效率地去操作電腦以達成雙向之互動，完成所希望藉助電腦完成之工作。用戶介面定義廣泛，包含了人機互動與圖形使用者介面。其涵蓋之範圍包括：必須有人類感官與作用體所產生輸出、輸入與運作內容三方互動設計。需考慮到人類有感官知覺、心理情緒、認知、學習、記憶、反

應、以及處理資訊的模式、個別背景之差異等等每一項都和用戶介面有密切的關係，直接或間接地影響用戶介面的效能。適當的說明文件，讓使用者明瞭系統究竟可以做與如何來達成工作。並讓使用者能夠花最少的時間在介面的熟悉上，而將時間投注於完成其預定之工作項目。最簡單的輔助說明就是圖形或符號[1]。以下針對三種屬性設定方式進行討論。

1.工具列設定

2.步驟式設定

3.智慧標籤。

2.3.1 工具列設定

最常見到的設定方式為使用工具列按鍵方式來執行以及編輯檔案。下圖為工具列的設定介面：

圖 5：工具列設定介面範例

其優勢在於工具列有現成元件控制項可以使用，開發困難度較低。其限制在於功能介面位置固定太佔版面、當要設定特定物件屬性時，能與不能設定的功能皆同時顯示，容易造成使用者混淆，易發生像是不知道哪裡可以設定元件的屬性、不小心關掉工具列時，可能發生不知道去哪裡重新開啟設定介面、在學習上是需要經過較長時間的熟悉與學習各元件屬性設定位置。

2.3.2 步驟式設定（精靈模式）

傳統網頁礙於開發技術仍不成熟，要開發出像單機應用程式一樣使用工具列的設定方式較為困難，於是大多採用步驟式方式 (Page by Page edit sequence) 來執行以及編輯檔案，又稱為精靈模式。

此方式為執行預先定義好的步驟程序，每步都會顯示設定的操作說明，引導使用者做設定動作。下圖為步驟式設定的界面：

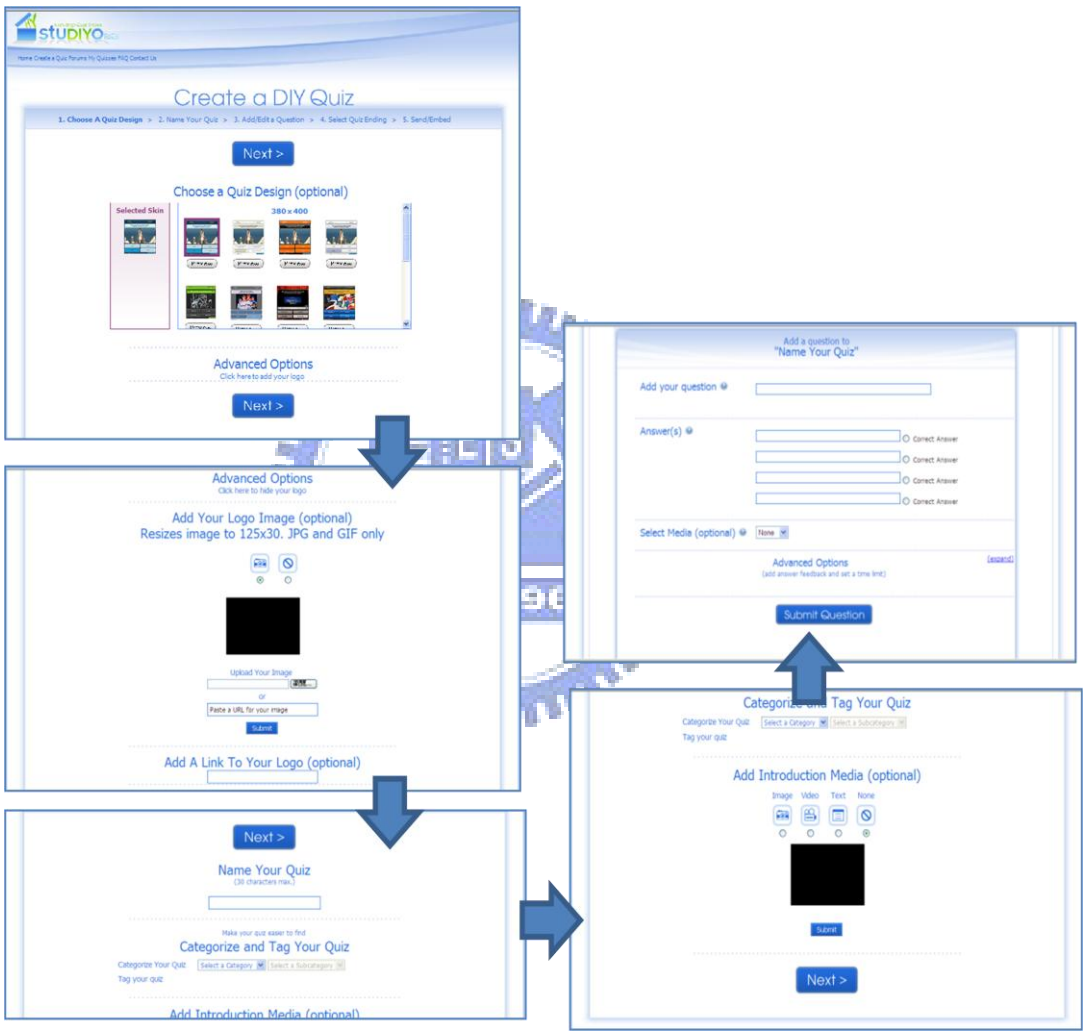


圖 6：步驟式設定介面範例

其優勢在於操作流程固定，每頁有輔助說明可以很清楚知道該設定什麼，操作簡單適合新手使用。其限制在於編輯缺乏彈性，無法隨使用者心境決定設定步驟，當有修改的需求時必須以回到上一頁或下一頁的方式進行修改，且不同頁間有用到共通控制項，其屬性設定需個別做開發，不管是對於使用者或開發者來說，都是非常沒有彈性的方式。

2.3.3 智慧型標籤 (Smart Tag)

智慧型標籤是一種在 Microsoft Word 97 開始出現的一種協助工具，其想法為有需要時才顯示相關資訊，Information On Demand (IOD) 的概念。當使用者將滑鼠指標移上去時，可以看到一個快顯功能表 (Context Menu)，如下圖所示，方便使用者可以執行特定工作。

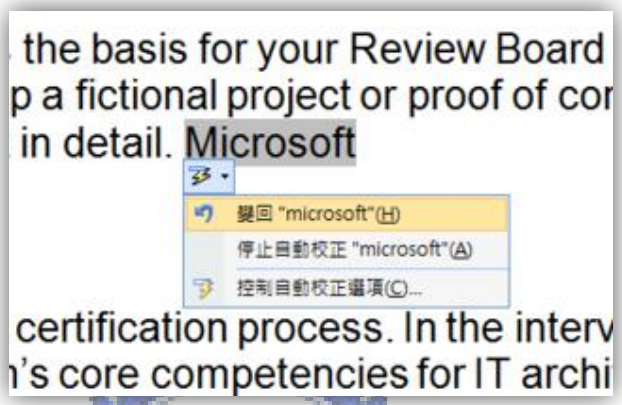


圖 7：智慧標籤

其優點在於使用者容易找尋元件屬性設定位置，新手也可以快速上手，本論文將其觀念套用至編輯器編輯過程當中，以下舉例做說明：點選圖片，出現圖片的智慧型標籤，圖片的智慧型標籤可選擇替換圖片的來源，不用特別去找尋哪裡可以選擇圖片來源，使用者可輕易的將圖片替換成自己想要的，這方面的功能尚無相關套件可直接使用，需自行設計開發所需功能。

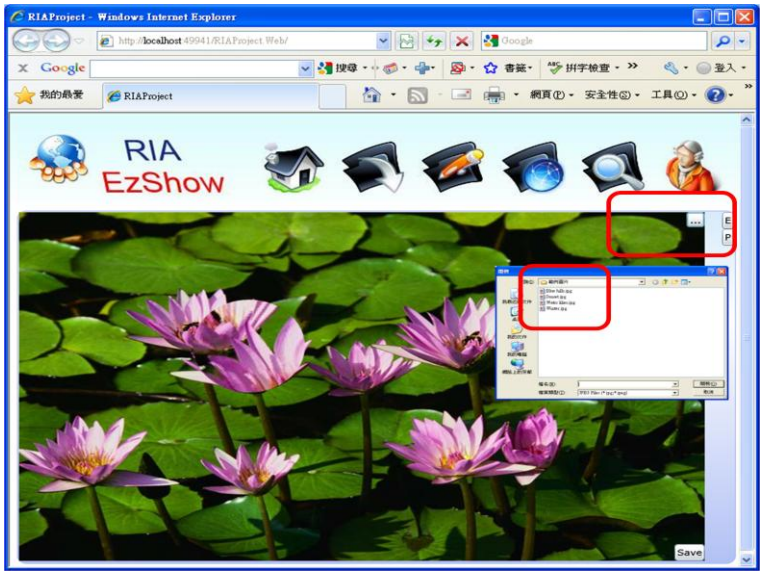


圖 8：Smart Tag 設定範例

2.3.4 功能介面展現方式討論

經過了解包含：1.工具列設定、2.步驟式設定、3.智慧標籤功能介面展現方式後，接下來比較三者較適合使用的時機。如下表所示。

表 5：功能介面展現方式討論

展現方式	在系統中適合使用的時機
1.工具列	軟體的全域功能 (ex: 開檔、存檔、說明、發佈...)
2.步驟式	固定、簡單的流程 (ex: 引導操作、編輯檔案的步驟...)
3.Smart Tag	較細節的設定 (ex: 元件屬性的設定...)

為了協助使用者能夠花最少的時間在介面的熟悉，將時間投注於完成其預定之工作項目。我們將採用工具列設定方式於主選單的設計，在樣板編輯過程步驟中採用步驟式設定引導使用者編輯流程，在樣版內容替換編輯時採用智慧標籤的設定方式。

參、 系統分析與設計

首先瞭解什麼是系統再針對系統組成要素做分析與規劃。Bertalanffy Von 於 1968 年提出：系統即是組織的元素交互作用所形成的複合體 [27]。Flagle 於 1960 年提出：系統是一群交互作用的元素所組合成的整體，用以完成預定的功能 [28]。簡單來說，系統即為一群要素的組合，經由要素彼此交互作用，可達成共同目標。系統要素包含硬體、軟體、資料、人員、人工作業程序。一個系統要能成功，最重要的關鍵要素就是需求的瞭解與需求項目的明確敘述。本研究依使用者需求將系統分成四大項說明。

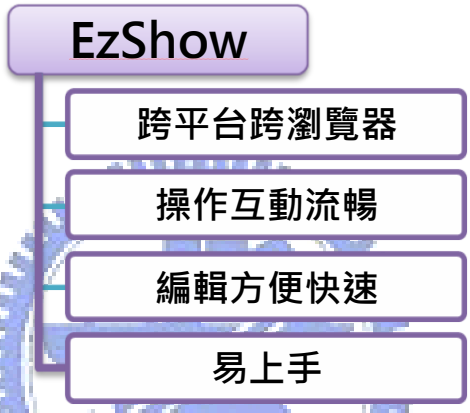


圖 9：使用者需求 - EzShow 特色

- **可跨平台跨瀏覽器與操作互動流暢**

需考慮系統的運作不能限制於特定平台，並且須具備高互動性操作性，讓每位網路使用者不需要安裝，都能隨時隨地直接在網路上做編輯。由上章討論將採用 Silverlight 技術來開發。

- **編輯方便快速**

需考慮到大部份的一般使用者並沒有美工排版或程式設計等專業背景，將採用模組化樣板的編輯方式，以提升編輯效益，讓使用者不需要使用者自己撰寫動畫或特效程式。

- **易上手**

需考慮到操作介面容易上手，適合每位使用者操作。只需短時間接觸，即可使用此系統製作出豐富互動的多媒體。在操作過程中，將融入 Smart Tag 觀念，使元件屬性設定更為直覺。另外在人機介面設計部分，也考慮到 Levie & Dickie 於 1973 年提出的使用者瀏覽習慣是由左上角至右下角 [29]，所以本系統各頁

面將會依使用者操作與瀏覽習慣做操作動線規劃。下面將會針對以上的需求做更深入探討，將需求分為功能性需求與非功能性，並將系統要素硬體、軟體、資料、人員與人工作業程序納入考量。

3.1 功能性需求

本編輯系統以使用者的需求來看，大致可以劃分成三部分去做深入探討，包含前置作業、伺服器端與使用者端編輯器各需要提供的功能，如下圖所示：

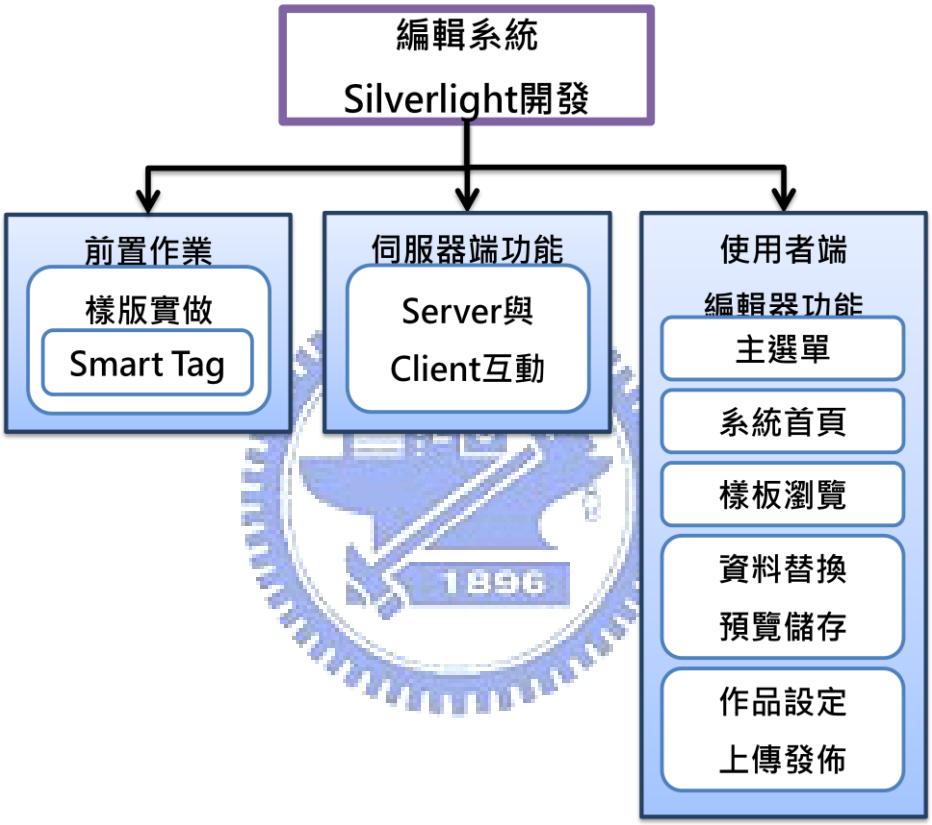


圖 10：EzShow 系統功能性需求

- **前置作業**

  必須先請專家做出各種豐富高互動樣版

- **伺服器端功能**

  接收使用者端傳來的需求，經過一連串伺服器端處理後，再回覆使用者端結果

- **使用者端編輯器功能**

  - **首頁設計**

在第二章節提到：『為了協助使用者能夠花最少的時間在介面的熟悉，我們將採用工具列設定方式於主選單的設計，在樣板編輯過程步驟中採用步驟式設定引導使用者編輯流程，在樣版內容替換編輯時採用智慧標籤的設定方式』。我們將系統的首頁的規劃分成上下兩部分：

1. 主選單：放至在系統頁面上方，方便使用者切換頁面使用，並會在選單按鈕加上提示，協助使用者更明白各按鈕功能
2. 首頁內容：會放至系統介紹以及各功能介紹，協助使用者快速熟悉 Ezshow 所有系統功能

■ **編輯替換**

整個編輯替換的動作有三個使用者頁面：

1. 選擇樣版：需提供樣版快照縮圖與樣版介紹以方便使用者選擇需要的樣版
2. 資料替換與預覽存檔：此畫面需包含：視覺化屬性設定介面、屬性設定與樣板互動、編輯中可以預覽效果、編輯完成將替換資料傳至伺服器端儲存
3. 作品設定與上傳發佈：在完成存檔後，使用者可針對作品相關資訊做設定或將作品發佈。使用者未對作品資訊做設定將會直接給系統預設值。

下面我們將接著介紹上述三項：前置作業、伺服器端功能、使用者端編輯器功能。

3.1.1 前置作業

本研究採用樣版式編輯方法，所以前置作業必須先產生好樣版，如下圖所示。

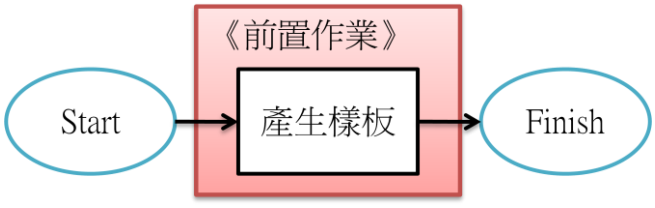


圖 11：系統功能性需求 1：前置作業產生樣板

樣板的製作與產生需考慮樣版的多樣化效果與互動能力，在網頁上呈現不能受 HTML 語法的限制。

- **硬體需求：**

  需要有檔案伺服器來儲存樣版

- **軟體需求**

  - 樣板 - 服務功能:可整合各種多媒體形態圖片,例如:圖片(PNG、JPG)、影片(VIDEO)、音樂(WMV、WMA、MP3)，並且可以依樣版規劃做出特殊劇情動畫

  - 樣板 - 操作介面、操作方式：依照專家自由做設計

- **資料需求**

  - 樣板 - 輸入項目：樣版上各式元件素材 (文字、圖片、影片、音樂、其他)

  - 樣板 - 輸出項目：樣版的模型、樣板展示用的替換資料，未來會被使用者替換掉的資料

- **人員與人工作業程序**

  樣板開發團隊，擁有製作豐富互動式多媒體樣板專長的專家，依需求設計與做出樣版

3.1.2 伺服器端功能

伺服器端的主要功能為接收使用者端編輯器與樣板傳來的需求,經過一連串伺服器端處理後，再回覆給使用者端結果，如：樣版下載、替換資料上傳、替換資料下載、使用者認證等等。

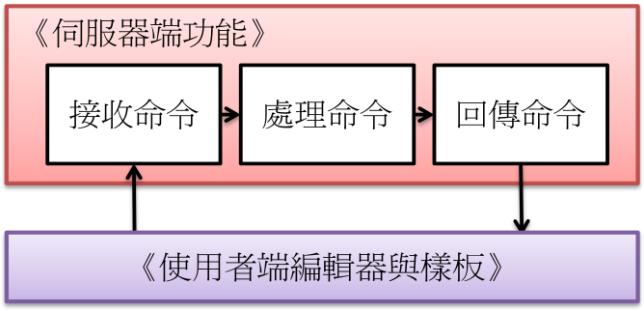


圖 12：系統功能性需求 2：伺服器端所需功能

3.1.3 使用者端編輯器功能

使用者端編輯器硬體、軟體、資料、人員、人工作業程序需求：

- **硬體需求**

  使用者操作編輯器時所需的硬體（電腦）、存放網頁的網站伺服器、存放編輯器的檔案伺服器、儲存樣版的檔案伺服器、儲存替換資料與資料描述檔的資料庫

- **軟體需求**

  完成編輯器的功能，下面小節將依序更詳細介紹。

- **資料需求**

  - 編輯替換 - 輸入項目：編輯器、樣版、替換資料、參數設定
  - 編輯替換 - 輸出項目：替換資料、作品描述檔

- **人員、人工作業程序**

  編輯器功能開發團隊，系統服務功能開發團隊。個別模組進行開發。

接下來各小節將持續探討編輯器所需要提供的功能，依序為首頁設計、編輯功能：選擇樣版、資料替換、預覽存檔、作品設定、上傳發佈。

3.1.3.1. 編輯器首頁設計

編輯器首頁設計步驟如下圖：

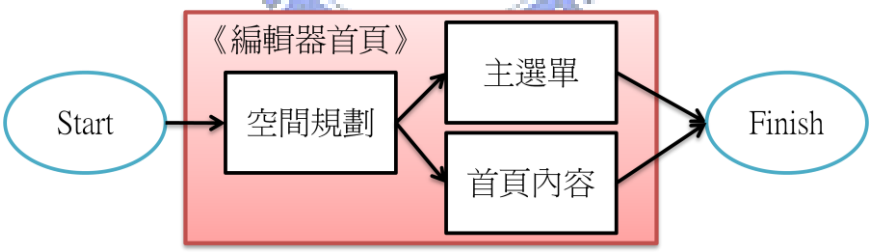


圖 13：系統功能性需求 3：編輯器首頁設計步驟

編輯器首頁設計在軟體需求方面分成三大部分個別討論：

- **服務功能：**

  主選單：協助使用者方便隨時切換到編輯器中任一個頁面

  首頁內容：協助使用者明白系統供用與操作說明

- **操作介面：**

  - 編輯器首頁設計介面規劃如下圖所示：

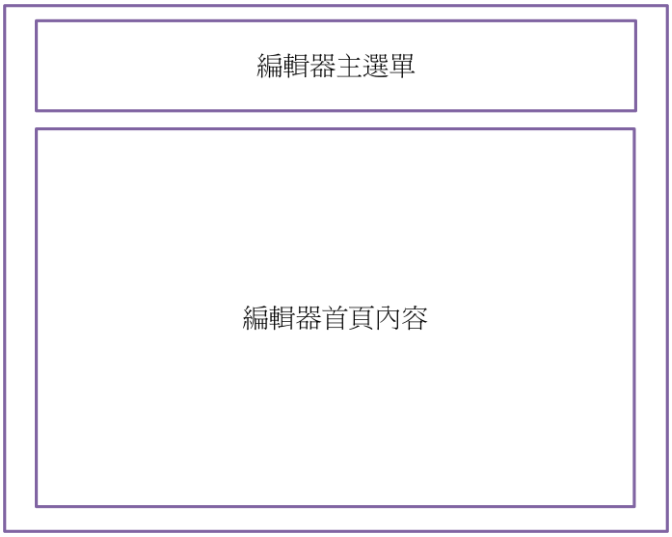


圖 14：編輯器首頁介面規劃

- 上面為編輯器主選單，下面放置編輯器首頁內容

■ 編輯器首頁之主選單設計介面規劃如下圖所示：

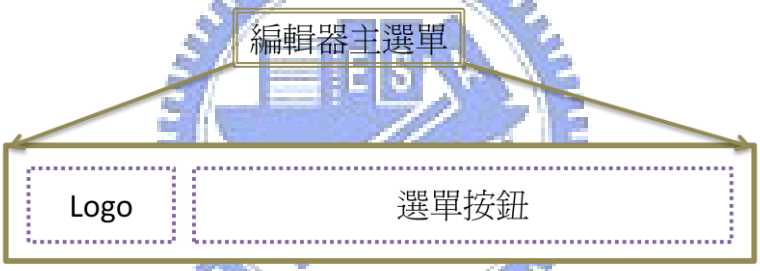


圖 15：編輯器首頁之主選單介面規劃

- 左邊為編輯器標誌，右邊為選單按鈕，將以圖示顯示按鈕
- 當使用者將滑鼠移至選單按鈕上面，選單按鈕需放大，協助使用者知道目前已經將滑鼠移植該按鈕，並出現提示，告知使用者按鈕為什麼功能
- 當使用者以滑鼠點選按鈕，按鈕需有動畫特效，協助使用者知道已經點選按鈕
- 按鈕的功能由左到右依序安排為：
  - ◆ 1.首頁
  - ◆ 2.選擇樣版、3.編輯替換&預覽存檔、4.設定&發佈作品
  - ◆ 5.瀏覽作品

◆ 6.帳號資訊

■ 編輯器首頁之首頁內容設計介面規劃如下圖所示：

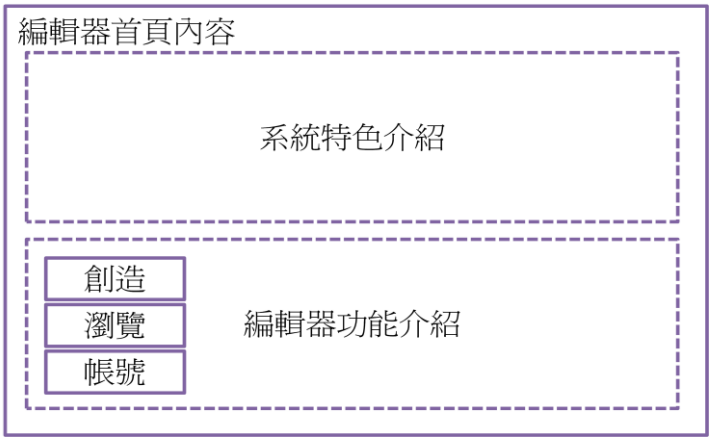


圖 16：編輯器首頁之首頁內容介面規劃

– 上面為介紹系統特色，下面為介紹系統功能

– 採用編輯器製作出一個系統特色說明的互動式多媒體

– 編輯器功能介紹：包含如何做出作品、或當沒有點子時可以先觀看其他人的作品、帳號資訊設定，並提供快速按鈕，協助使用者可以依需求直接切換至目標頁面。

● **操作方式：**

關於編輯器首頁操作方式的部分，使用者在畫面上的操作動線以實心箭頭呈現，滑鼠點選左鍵動作以空心箭頭呈現，如下圖所示：

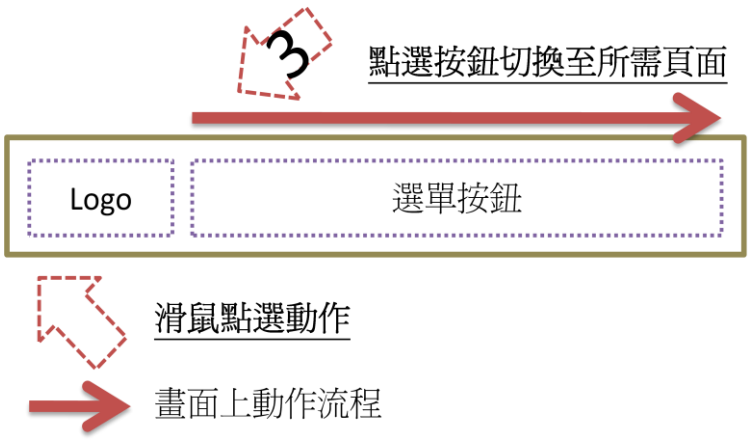


圖 17：主選單頁面操作流程動線與滑鼠動作動線

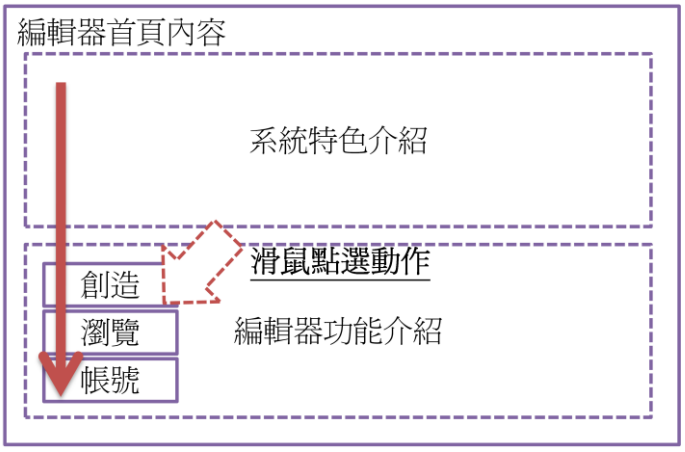


圖 18：首頁內容頁面觀看與操作流程動線與滑鼠動作動線

3.1.3.2. 選擇樣版

在編輯替換的步驟大至為：Step1：選擇樣版 → Step2：資料替換 → Step3：預覽存檔，如下圖所示。而作品設定與上傳發佈使用者可以選擇暫不做設定。

圖 19：系統功能性需求 3：使用者編輯替換的步驟

選擇樣版在軟體需求方面分成三大部分個別討論：

- **服務功能：**

  編輯第一步為選擇樣版。首先瀏覽各式樣版，然後點選一個自己喜歡的樣版，將樣板下載至使用者端後再切換至進行編輯替換的動作。由於樣版是儲存在伺服器上，所以需提供編輯器遠端下載樣版功能。

- **操作介面：**

  樣板選擇操作介面規劃如下圖所示：

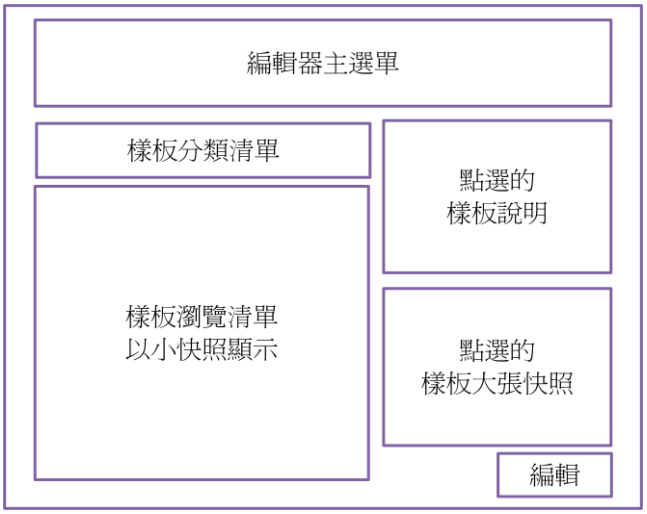


圖 20：樣板介面規劃

- 在編輯器上需規劃出一塊空間放置樣版的瀏覽清單與規劃出一塊空間放置樣板詳細說明的區域。
- 需將樣板分門別類，讓使用者可選擇僅瀏覽特定類別的樣版，加速樣版選擇速度。預設是顯示所有的樣版。
- 當使用者在樣板瀏覽清單上點選某個特定樣板時，在樣板詳細說明區域當中需展示樣版的名稱、敘述及快照，讓使用者當作選擇樣版的參考。
- 在瀏覽清單中，項目以小快照顯示。這樣瀏覽清單才可以在一個頁面中放置較多的選項。而在樣板說明區，因為顯示的資訊為使用者特地點選的項目，所以此區的樣版以大快照顯示。
- 當使用者點選完成後，需切換至編輯替換資料的頁面，為了避免使用者不小心點選到，於是將編輯按鈕規劃在整個畫面右下角。

- **操作方式：**

關於樣板選擇操作方式的部分，使用者在畫面上的操作動線如下圖實心箭頭所示。而滑鼠點選左鍵動作如下圖空心箭頭所示：

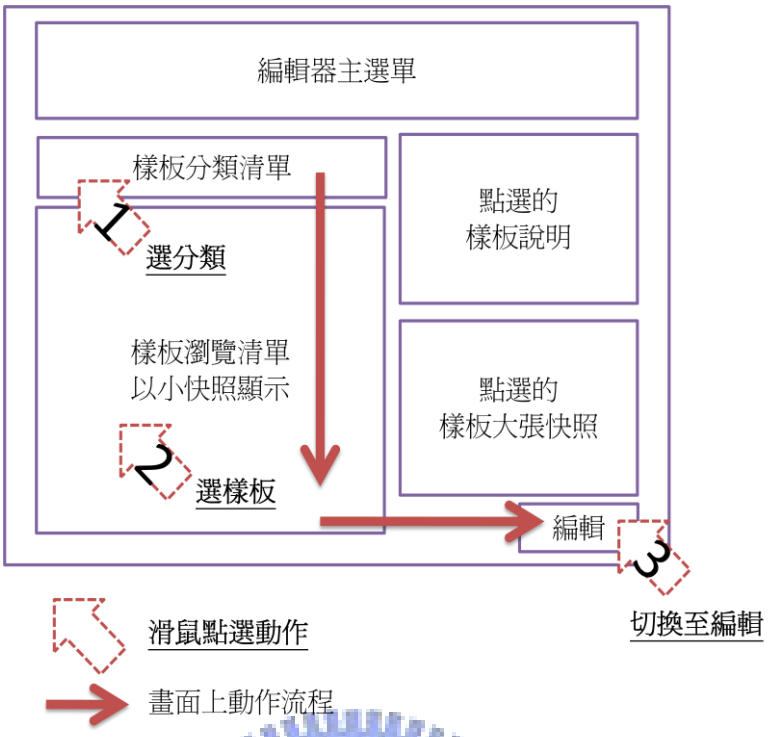


圖 21：樣板選擇頁面操作流程動線與滑鼠動作動線

3.1.3.3. 資料替換

完成編輯替換的第一步驟：選擇樣版後，接下來就是進行編輯替換中的第二步驟：資料替換。在步驟二資料替換軟體需求方面分成三大部分個別討論：

- **服務功能：**

  進行資料替換的動作。由於屬性設定採用智慧標籤方式，所以必須實做視覺化屬性設定介面並且處理屬性設定介面與樣板之間的溝通互動。

  - 視覺化屬性設定介面：提供使用者進行屬性設定的界面。採用智慧型標籤方式做設定。
  - 屬性設定與樣板互動：在使用者設定完屬性後，需立即將參數資料傳遞至樣版整合呈現。

- **操作介面：**

  資料替換操作介面規劃如下圖所示：

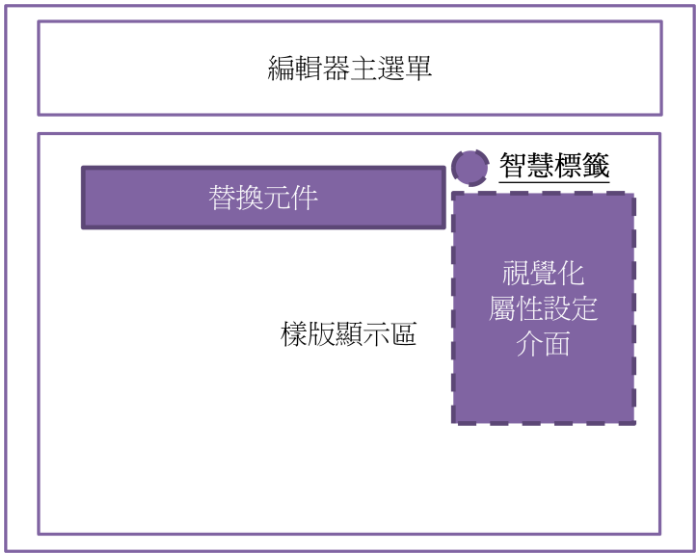


圖 22：資料替換介面規劃

- 需規劃一個樣板顯示區放置樣版，使用者可利用智慧標籤在樣板上面做資料替換。
- 各替換元件都有自己專屬的智慧標籤。
- 智慧標籤包含告知圓點與視覺化屬性介面。
- 告知圓點會顯示在替換元件右上角，告知使用者此元件可以設定屬性，以及協助使用者可以立即找到此元件設定屬性界面的地方。
- 視覺化屬性介面一開始並不會顯示，當使用者點選告知圓點後才會顯示，視覺化屬性介面裡面含有許多參數設定的控制項。使用者利用這些控制項進行元件屬性設定。

- **操作方式：**

關於資料替換頁面操作流程動線與滑鼠動作動線的部分如下圖所示：

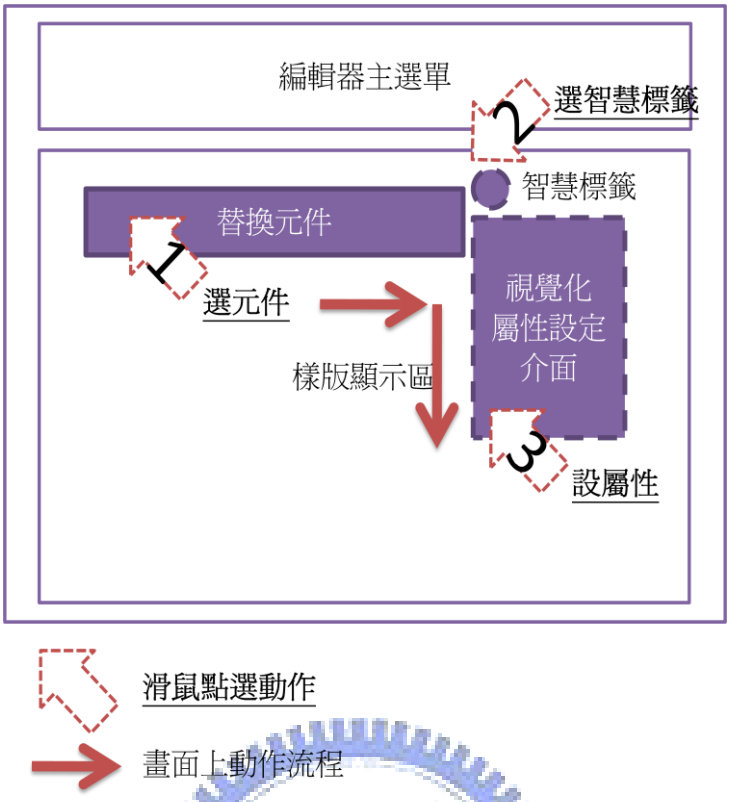


圖 23：資料替換頁面操作流程動線與滑鼠動作動線

3.1.3.4. 預覽存檔

完成第二步驟資料替換接下來就是進行第三步驟預覽存檔。在預覽存檔軟體需求方面分成三大部分個別討論：

- **服務功能：**

  進行預覽存檔的動作，在編輯的過程中，可以切換編輯模式與檢視模式來達成編輯中預覽。當確認完成編輯後，按下存檔按鈕，會將替換資料存至伺服器，接著畫面會由編輯頁面切換至作品資訊設定與發佈頁面。

- **操作介面：**

  預覽存檔的操作介面規劃如下圖所示：

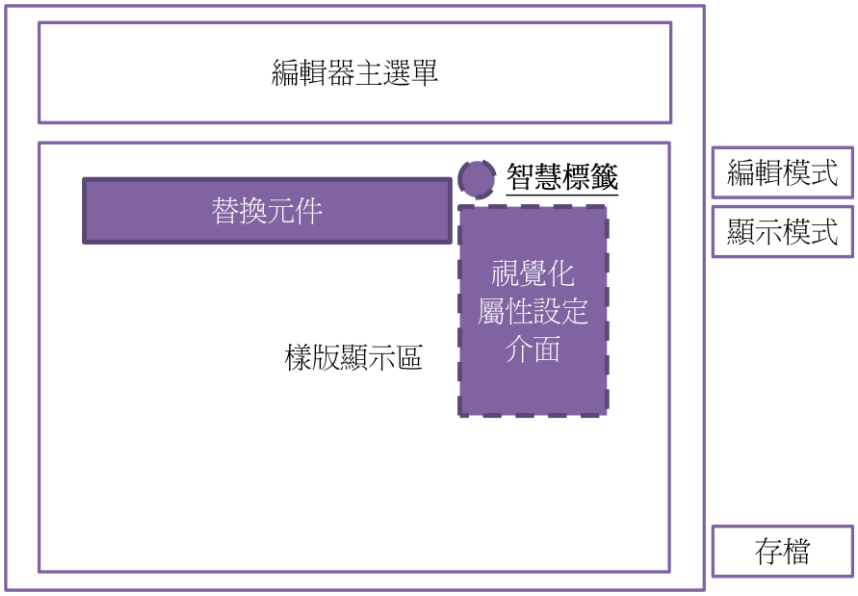


圖 24：預覽存檔介面規劃

- 有些樣板替換後可立即呈現效果，有些樣板含有與使用者互動的動作，需接受使用者命令後才會顯示其效果，所以需設置兩個按鈕，為編輯模式與顯示模式。編輯模式會顯示智慧標籤，而顯示模式呈現出來的就是未來編輯完成後的樣貌與效果，此模式不會顯示智慧標籤。
- 當是編輯模式的時候，編輯模式的按鈕是不可以按的。
- 當是檢視模式的時候，檢視模式的按鈕不可以按的。這樣的設計可以告知使用者目前是哪種模式，以及可以切換的模式
- 當資料替換完成後，接著需進行存檔動作。規劃一個存檔按鈕於編輯器頁面右下角，方便使用者進行下一個動作。

- **操作方式：**

關於預覽存檔操作方式的部分，使用者在畫面上的操作動線如下圖實心箭頭所示。而滑鼠點選左鍵動作如下圖空心箭頭所示：

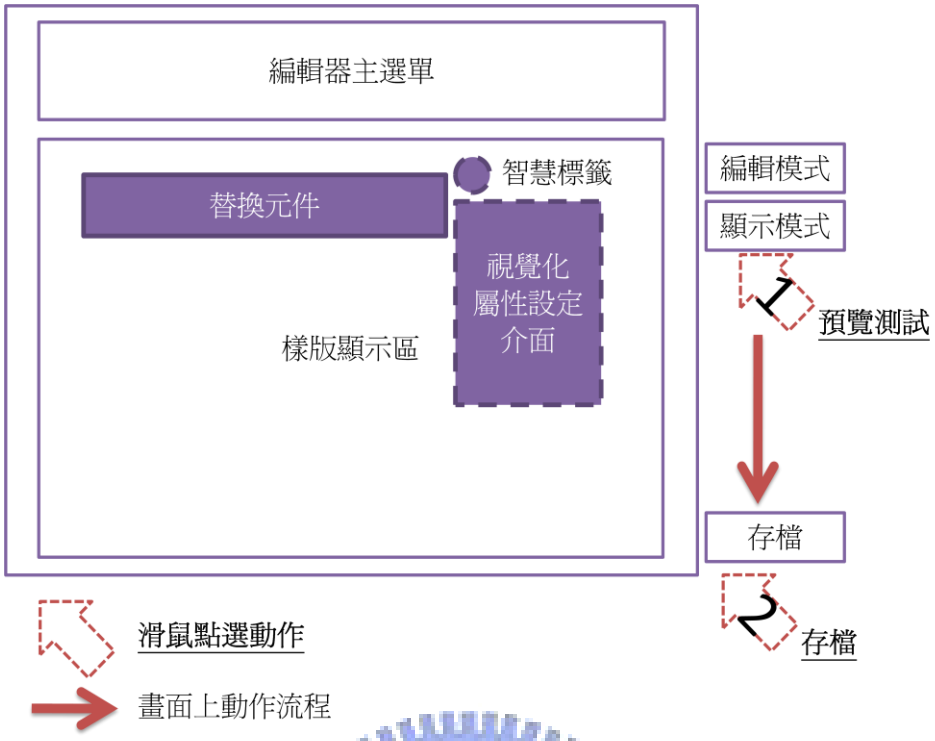


圖 25：預覽存檔頁面操作流程動線與滑鼠動作動線

3.1.3.5. 作品設定

在完成作品編輯並將替換資料、作品描述檔儲存至伺服器的資料庫後，接下來可選擇是否執行的功能包含設定作品與發佈作品。如下圖所示。

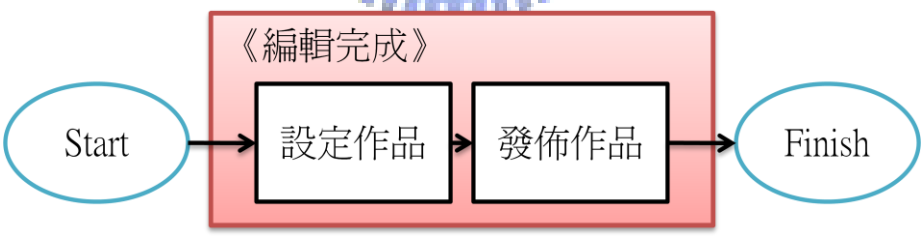


圖 26：系統功能性需求 3：編輯器編輯多媒體檔案的步驟

設定作品細節資訊功能的軟體需求方面分成三大部分個別討論：

- **服務功能：**

  在編輯畫面按下存檔按鈕後，接著畫面會由編輯頁面切換至作品資訊設定與發佈的頁面。在此頁面裡，需提供使用者設定自己編好的作品相關資訊，例如：作品描述、作品是否分享給別人觀看、是否允許別人也可以發佈自己的作品。

- **操作介面：**

作品設定的操作介面規劃如下圖所示：

圖 27：作品設定介面規劃

- 規劃出一塊區域放置輸入作品資訊的選項與說明。
- 當輸入完成後，接著需進行第二次存檔動作。規劃一個設定按鈕於編輯器頁面右下角，方便使用者進行下一個動作。
- 如果使用者未設定即離開，則預設資訊為作品描述：未說明、是否分享：不分享、是否發佈：不發佈。

- **操作方式：**

關於作品設定發佈頁面操作流程動線與滑鼠動作動線的部分，如下圖所示：

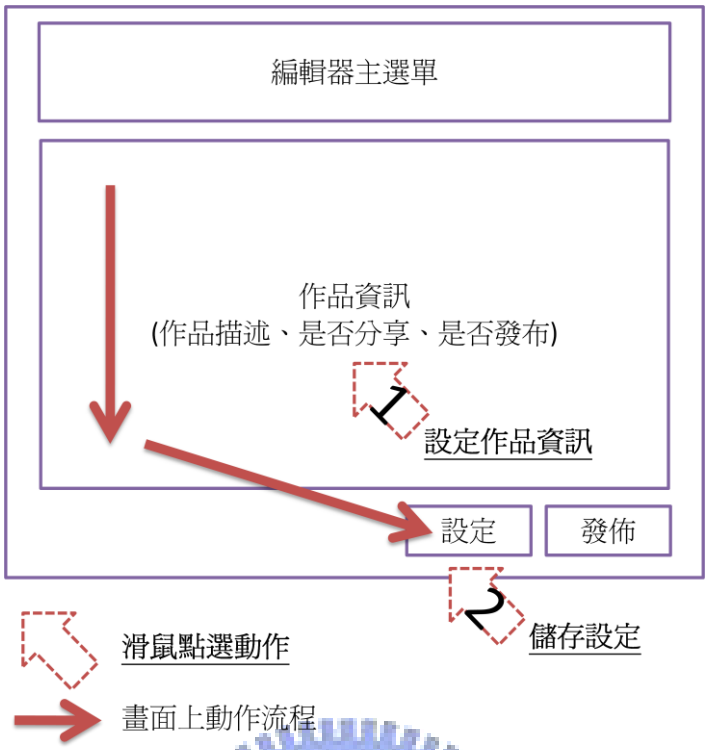


圖 28：作品設定發佈頁面操作流程動線與滑鼠動作動線

3.1.3.6. 發佈作品

設定作品細節資訊後接下來使用者可以考慮是否將作品發佈至所需網頁上。此功能軟體需求方面分成三大部分個別討論：

- **服務功能：**

  在設定完作品資訊，按下設定按鈕後，接著使用者可以將自己的作品發佈至所需網頁上。使用者按下發佈網址的按鈕後，網頁會跳出一個顯示框，裡面會有作品發佈網址，使用者可將網址貼至欲顯示的網頁裡。

- **操作介面：**

  發佈作品網址的操作介面規劃如下圖所示：

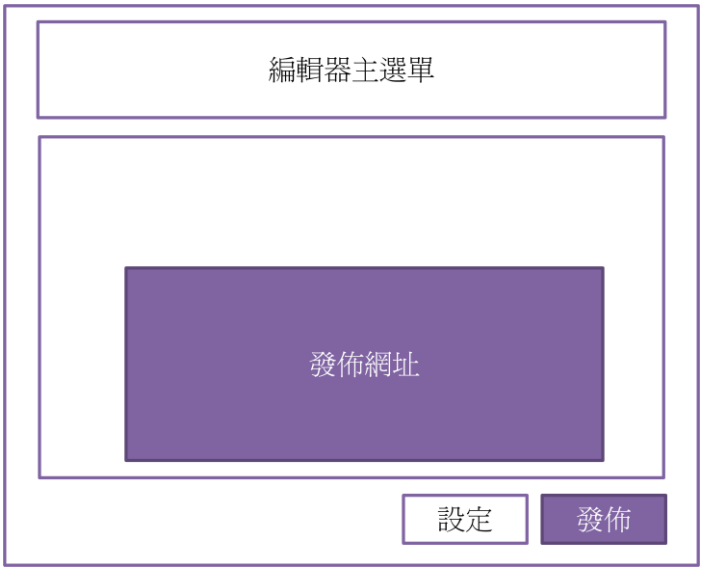


圖 29：發佈作品介面規劃

- 規劃一個設定按鈕於編輯器頁面右下角，方便使用者進行發佈網址的動作。
- 規劃一個跳出框放置發佈作品網址。

- **操作方式：**

關於作品設定發佈頁面操作流程動線與滑鼠動作動線的部分，如下圖所示：

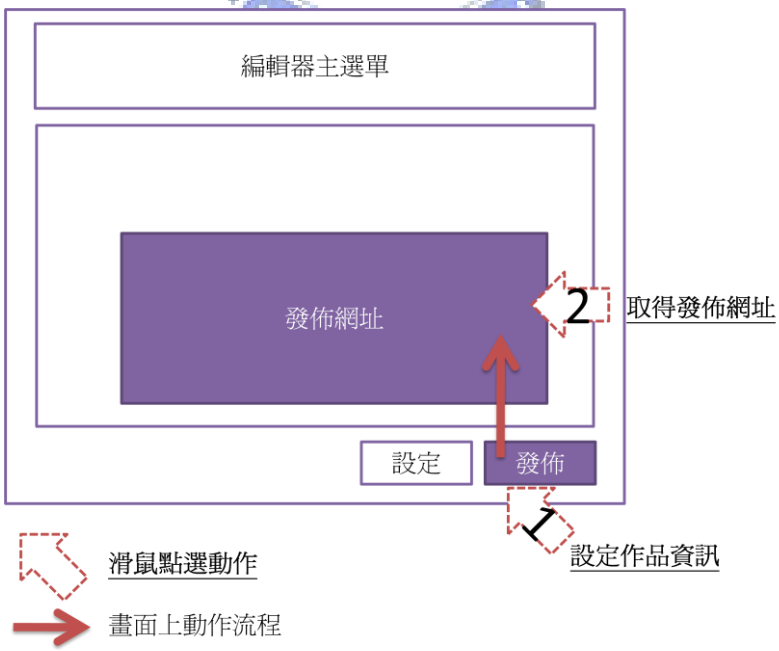


圖 30：作品設定發佈頁面操作流程動線與滑鼠動作動線

3.2 非功能性需求

建構應用系統的關鍵在於系統架構基礎是否良好，其次為建構一個有延展性的系統。對於開發人員來說，最常面臨需不斷更改已經設計規劃好的程式。而系統又可以分割成若干個子系統，子系統間透過介面進行訊息的交換與交互作用，必須考慮到子系統之間的耦合力要越低越好，程式的架構也是一樣的。如何能提供彈性且安全的系統架構與富含擴充性的程式架構，本章將針對於此三點進行深入探討：系統架構分層（展現層、邏輯層、資料層）、編輯器分頁呈現、樣版設計。

3.2.1 系統架構分層

將系統架構分層的設計原則來自於希望提升系統的彈性與安全性。將系統功能分成伺服器端與使用者端。編輯器執行於使用者端的瀏覽器上面，伺服器端提供使用者端編輯器存取資料庫的服務，取代編輯器擁有太高權限直接存取遠端資料的風險。採用 Service-oriented architecture (SOA) 通訊服務的架構，它是一種設計模式與設計方法，將系統劃分為三層：展現層、邏輯層、資料層，如下圖所示。

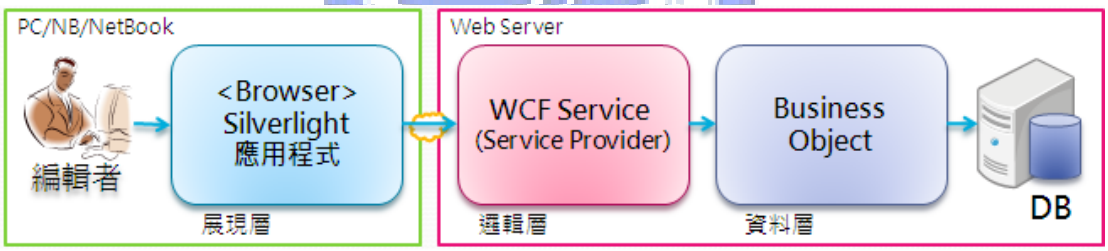


圖 31：EzShow 系統採用 SOA 架構圖

- 展現層：圖中最左邊為展現層，將使用者介面交由使用者端編輯器處理。
- 資料層：圖中最右邊為資料層，資料庫存取交由後端伺服器處理。
- 邏輯層：圖中間為邏輯層，提供服務介面，負責界接前端編輯器需求與後端資料庫存取回應 。

SOA 架構的特性如下：

- 鬆散耦合 (Loose Coupling)，降低系統相依性 (Dependency)

- 提高系統安全性（Security），分層把關，限制使用者端存取系統資料的動作
- 服務介面設計與管理
- 標準化的服務介面易維護
- 可重用降低開發/異動成本

3.2.2 編輯器分頁呈現

將編輯器分頁呈現的設計原則來自於希望提升使用者使用 EzShow 的意願。在平常瀏覽網頁的過程當中，常遇到當網頁開很久仍未完全開啟，我們就會放棄去瀏覽那個頁面，轉向去看其他的頁面。反觀網路應用程式也是一樣的，當程式或檔案很大需很長的下載時間，就會降低使用者使用的意願。且在開發當中遇到隨著 EzShow 持續擴充功能，整個編輯器的檔案也越來越龐大，持續增加下載編輯器需等待的時間，造成使用者越來越不想使用，由其當使用者身處在頻寬不是很足夠的環境下。

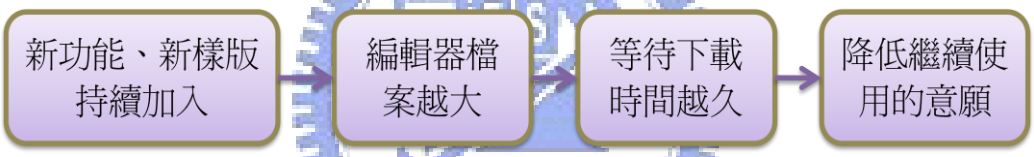


圖 32：編輯器分頁呈現設計考量

另外，考量到使用者並非每次皆會使用到編輯器中所有的功能，因此，並不需要讓使用者一次就將所有編輯器相關的資訊全部下載。有鑑於此，我們必須針對這點想辦法改進編輯器與相關檔案的下載方式，提升使用者使用 EzShow 的意願。 為了解決此問題，我們將 EzShow 編輯器的部分頁面切割成獨立模組，需用到才下載，如下圖所示。

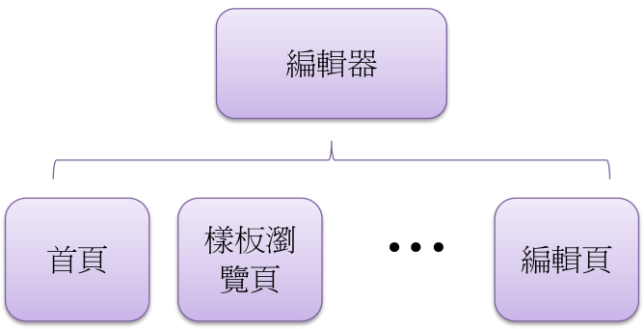


圖 33：EzShow 編輯器分頁呈現

範例：首頁，需要設計成檔案小，可立即下載呈現。而在樣板清單瀏覽頁，提供樣版的縮圖，讓使用者預覽所需樣板，確定要使用在將真正的樣版檔案下載至使用者端。

3.2.3 模組化樣版設計

樣版採用模組化設計的原因來自於傳統編輯器的做法常會限制住樣版的多樣化。傳統編輯器會將樣板可以執行的動作或設定等功能，全部開發在編輯器裡面，這樣的實作方式會導致樣版的多樣性受到限制。例如：編輯器僅提供單純的文字、圖片、影片、聲音做替換及基本屬性設定。這時後想要融入一個多樣化的投影片切換動畫會受到許多限制，因為編輯器並不支援此功能。例如無法設定投影片切換的速度。如果想納入此類型樣版，勢必花費許多時間與成本重新對編輯器重新設計與開發。但這的解決方式又造成另一個問題，當編輯器納入的功能越來越多後，勢必造成整個編輯器檔案越來越大，許多使用者不會用到的功能也必須一併下載至使用者電腦裡，是非常不經濟的做法。

有鑑於上面的問題與提升樣版的擴充性及多樣性，樣板的設計需考慮到將設定元件屬性的編輯與編輯器獨立分開，讓編輯器無法限制住樣版可替換的元件類別或屬性，讓樣版製作專家可以依造設計隨心所欲製作出樣版和樣板上允許改變的元件。

將樣版獨立於編輯器，各樣版採獨立化模組化的設計方式。接著來探討樣板需提供的架構與行為模式。每個樣板包含畫面呈現、替換資料、播放邏輯與編輯邏輯，如下圖所示。在程式撰寫時需考慮到將樣版的畫面呈現與撥放、編輯邏輯分開，降低彼此之間的相依度，以便提升我們未來在做樣版修改或操作資料替換時的容易度。例如：未來再做樣版修改時，只更動前端畫面呈現的部分，重新排成自己要的畫面，但不去更動到動畫撥放的邏輯。

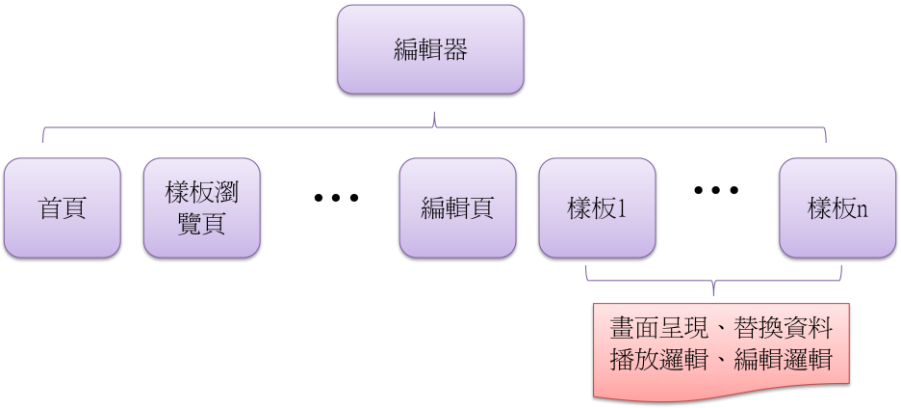


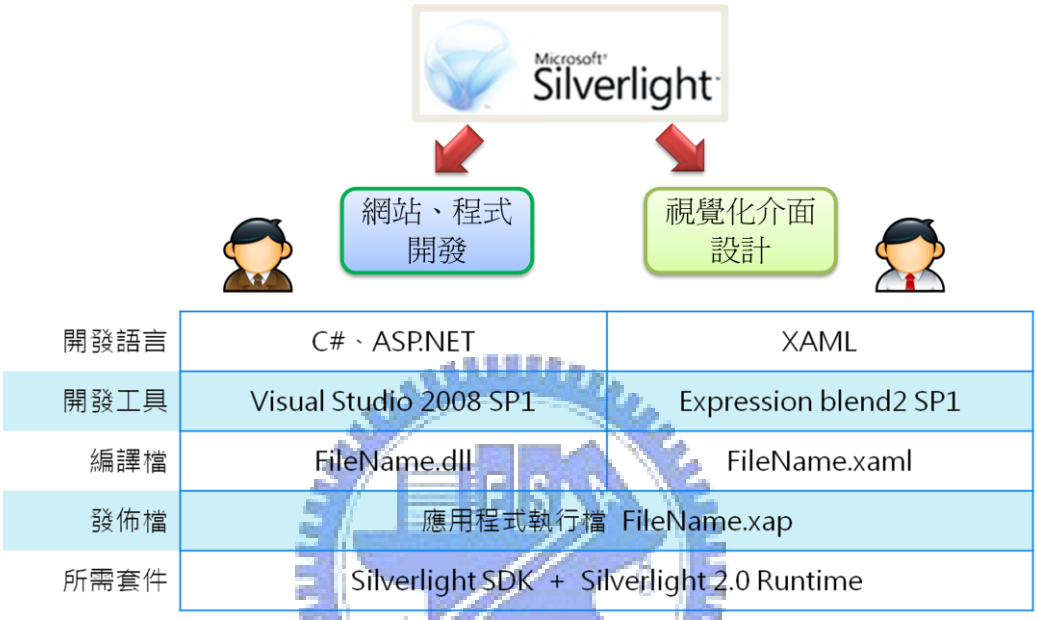
圖 34：模組化樣版設計與樣版的組成

另外，樣版採獨立化模組化的設計方式的另一個好處是，樣板與多媒體完成檔可單獨播放觀看。因為採用 RIA 技術開發加上樣板與編輯器獨立，所以不需編輯器或撥放器仍可單獨執行觀看。使用者將完成的作品發佈至特定網頁後，觀看網頁的人不用在另外安裝撥放器即可觀看，其細節之後實做會再詳細介紹。

肆、 系統架構與實作

4.1 開發工具與環境

本系統開發工具與環境如下圖所示：



開發語言	C#、ASP.NET	XAML
開發工具	Visual Studio 2008 SP1	Expression blend2 SP1
編譯檔	FileName.dll	FileName.xaml
發佈檔	應用程式執行檔 FileName.xap	
所需套件	Silverlight SDK + Silverlight 2.0 Runtime	

圖 35：EzShow 開發工具與環境

在網站與功能程式方面的開發採用 ASP.NET 中 C#程式語言，開發工具為 Visual Studio2008 SP1，檔案附檔名為.cs，編譯後的檔案格式為.dll。在視覺化介面設計與部分動畫效果設計方面採用 Extensible Application Markup Language (XAML) 標記語言，開發工具為 Expression blend2 SP1，檔案附檔名為.xaml。開發環境需安裝 Silverlight SDK 與 Silverlight 2.0 Runtime，Visual Studio2008 SP1 包含.Net framework 3.0 以上版本。

4.2 EzShow 系統架構圖

本系統架構採用 SOA 架構設計模式規劃，如下圖所示：

- 展現層：瀏覽器上的編輯器使用者介面為展現層
- 資料層：網路伺服器區右上的資料庫存取 LINQ to SQL 為資料層
- 邏輯層：網路伺服器區中提供服務介面的 Web Service 為邏輯層

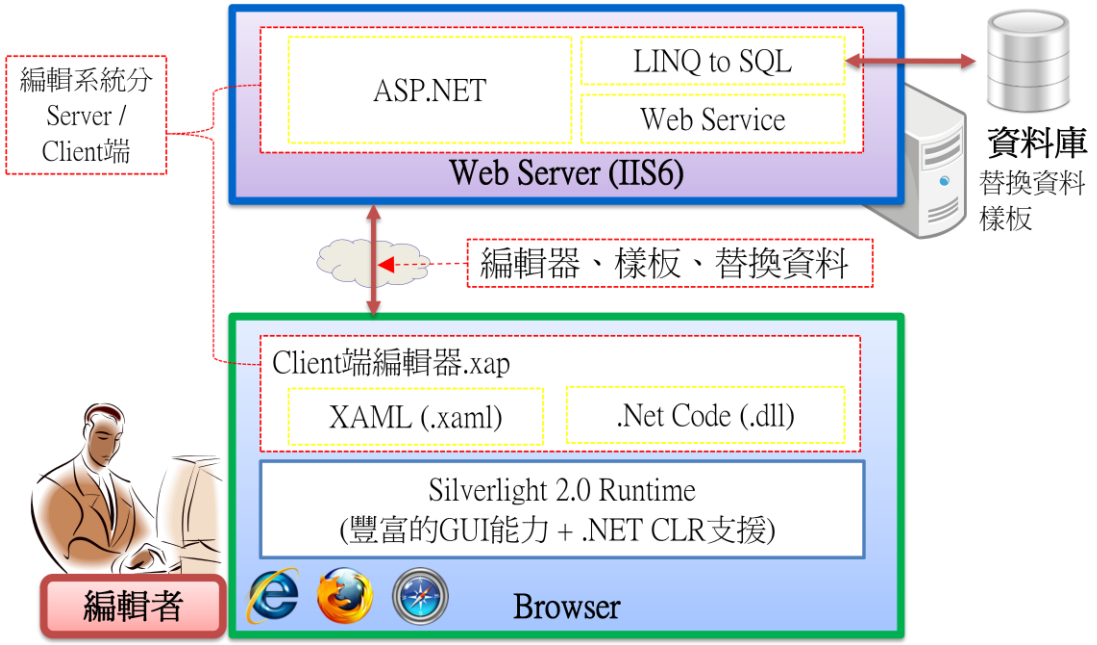


圖 36：EzShow 系統架構圖

使用者透過瀏覽器上網下載編輯器、下載樣版，上傳替換資料、下載樣版、下載替換資料、下載資料描述檔。以下將架構組成分成三個部分討論。

1. 編輯者端瀏覽器

下表為 Silverlight runtime 支援的瀏覽器類型與各瀏覽器支援的 Silverlight runtime 版本 (2009 年 6 月正式版有 1.0 與 2.0)。

表 6：可安裝 Silverlight 的瀏覽器與其支援的版本

OS / Browser	IE7	IE8	Firefox	Safari
Vista / windows7	1.0 2.0	2.0	1.0 2.0	1.0 2.0
XP /2003 /Home Server	1.0 2.0	2.0	1.0 2.0	1.0 2.0
MAC OS 10.4/10.5 Power PC			1.0	1.0
MAC OS 10.4/10.5 Intel			1.0 2.0	1.0 2.0
Linux X86 (Moonlight)			1.0 2.0	

在使用 EzShow 前需預先安裝好 Silverlight runtime 2.0 版本，為安裝則會自動引導使用者安裝。開啟已經安裝 Silverlight runtime 的瀏覽器，連線到系統，瀏覽器會自動將 EzShow 編輯器下載回使用者電腦瀏覽器裡的暫存區，下載回的編輯器副檔名為.xap，其壓縮檔內主要會包含畫面呈現的.xaml 檔案

與.dll 程式邏輯檔案。

2. 網站伺服器

網站伺服器負責存放 EzShow 系統所有的東西，包含編輯器、樣版與網路服務 (Web Service) 提供，等待使用者透過網際網路提出需求。使用者連線至網站上將編輯器取回使用，當選擇好想要編輯的樣版再將樣板下載回使用者端。使用者編輯完成想要執行儲存資料或瀏覽相關資料再透過網站提供的 Web Service 與後端資料庫做溝通。網站以 ASP.NET 實做，將 EzShow 系統放置在網站裡。Web Service 中關於資料庫存取部分以 LINQ to SQL 實做，藉由使用 LINQ to SQL 元件，可以用 Language Integrated Query (LINQ) 功能來存取 SQL 資料庫。

3. 儲存替換資料的資料庫

用來存放使用者替換完成的替換資料。資料表包含各樣版的替換資料表、各作品描述檔的資料表、使用者基本資訊資料表。使用者端想要存取這些資料需藉由編輯器透過伺服器上提供的 Web Service 存取。

4.3 EzShow 功能模組與實做

本節將依上章節系統分析與設計提出的需求進一步討論 EzShow 編輯系統的功能模組實做，共劃分成三部份：

- 前置作業：樣版產生
- 伺服器端系統功能
- 使用者端編輯器功能：使用者端編輯功能與編輯器畫面製作

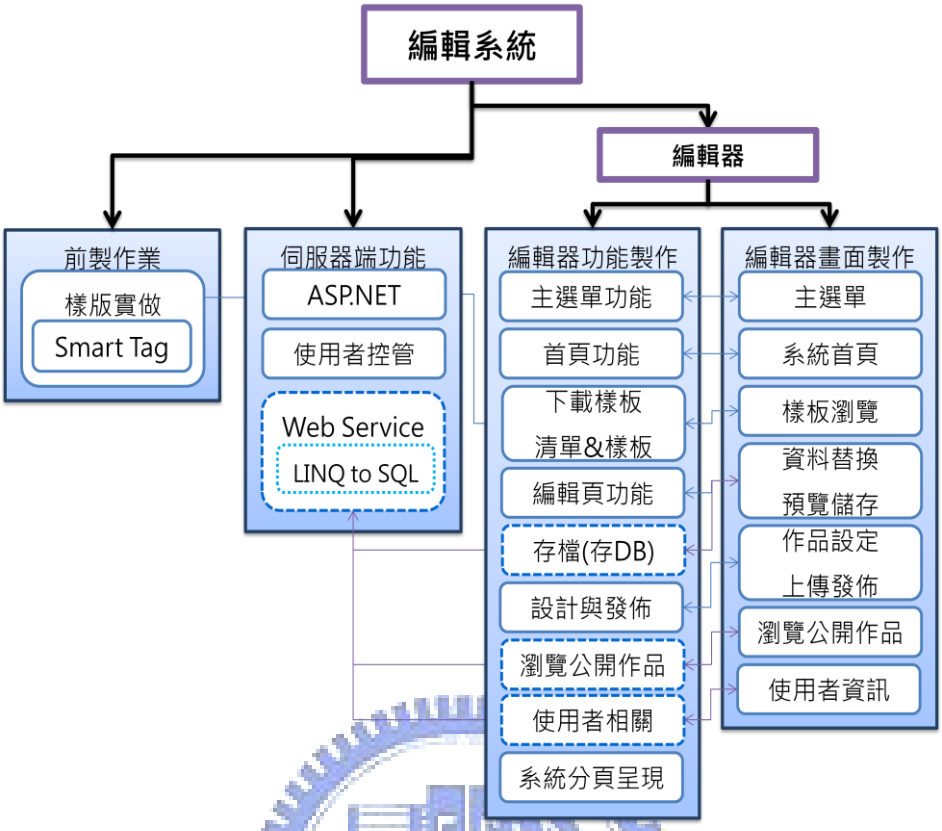


圖 37：EzShow 系統功能開發模組圖

使用者在整個編輯流程當中，首先會接觸到編輯系統伺服器端，接著從伺服器上下載編輯器，最後下載樣版做編輯，所以將以下順序依序介紹系統實作細節，特別會強調系統需求中提出的部分予以說明：伺服器端系統功能、使用者端編輯實做、樣版製作。樣板模組化與編輯器功能實做採用同樣程式撰寫設計模式的架構將一併說明。

4.3.1 伺服器端系統功能實做

伺服器端的主要系統功能如下圖：

圖 38：EzShow 系統功能開發模組圖

需先建置一個 ASP.NET 網站，將我們開發的應用程式放在上面跑，並對網站伺服器做一些特別的設定，接著設定與實作使用者控管的部分，最後提供網路服務給編輯器使用。提供使用者端存取伺服器端資源有兩個地方：

1. 網站檔案系統：樣版
2. 資料庫：樣版的資訊、作品描述檔、使用者的替換資料

架構圖如下：

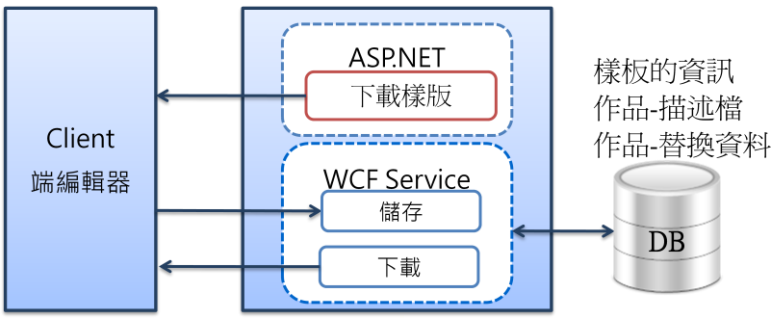


圖 39：提供使用者遠端存取伺服器資源架構圖

與樣版相關的功能會發生在選擇樣版與瀏覽公開作品的網頁上，編輯器與伺服器的溝通動作細節如下：

- 當使用者切換至選擇樣版的頁面，編輯器會自動送出取得瀏覽所有樣板清單的命令至伺服器端。待伺服器回傳清單資料至編輯器，再由編輯器將資料呈現在瀏覽樣版清單區。
- 當使用者自行點選瀏覽特定類別樣版，編輯器接收使用者命令後，會將資料類別過濾後呈現在瀏覽樣版清單區。
- 當使用點選特定樣版，編輯器會將特定樣板的資料顯示在樣板詳細說明區。這時候呈現的資料都是在頁面剛載入時已經下載好的。
- 當使用者確定選擇的樣版後，將會主動按『編輯』按鈕，編輯器會被動送出取得特定樣版詳細資料的命令至伺服器端，這時候才是發出將真正需編輯的樣版下載至使用者端命令。待伺服器以檔案下載的方式回傳樣版至編輯器，再由編輯器將樣板呈現在資料替換頁。而按鈕功能也會將頁面由選擇樣版頁切換至編輯替換頁。
- 在使用者瀏覽公開作品時，會先至資料庫裡下載作品描述檔，接著依造描述檔的內容下載樣版，再將描述檔給樣版，由樣板去下載替換資料，整合呈現出作品的效果。

實做 Web Service 提供使用者存取後端資料庫中的樣版資訊、作品替換資料與作品描述檔。將所有程式邏輯及服務內容全部放在服務裡面，實作一個標準

的介面與編輯器做溝通。介面定義的方式、資料格式、與溝通管道必須是產業標準 (http、XML、SOAP 等)。本系統 Web Service 開發技術採用 Windows Communication Foundation (WCF) 。WCF 是一個元件，它是微軟平台上 SOA 架構，用來建置分散式與可互動式的應用程式。使用者端的編輯器呼叫 WCF Service 時，只會管服務介面的輸入與輸出，不會去管服務實作的細節。

WCF 主要分為 Client 端與 Server 端，兩端之間進行通訊的流程如圖所示。

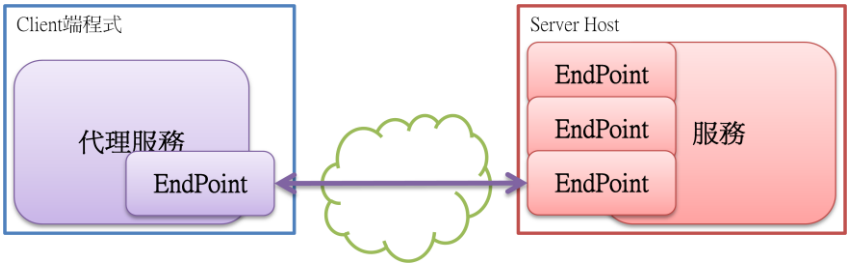


圖 40：WCF 工作原理

Server 端首先要建立服務，然後用 EndPoint 與 Client 端通訊。Client 端會在用應時產生一個代理服務，接著才使用者這個代理服務。EndPoint 包含契約內容、繫結格式和服務位址等，讓 Client 端知道 Server 端提供何種服務、如何叫用服務、可從何處取得服務。以下為 Server 端存檔時 WCF Service 的實做步驟：

1. 建立一個 WCF 介面，命名為 SaveService.svc 副檔名為.svc
2. 製作一個 LINQ 物件，命名為 TemplateData.dbml，副檔名為.dbml
3. 開啟 TemplateData.dbml 開啟伺服器總管，將需要的資料庫拖拉到 TemplateData.dbml 檔案裡
4. 撰寫 WCF 介面，也就是 Endpoint 內容，定義提供服務的輸入與輸出，介面需有 [OperationContract] 屬性，表示此為一個服務操作合約介面。

4.3.2使用者端編輯器實做

本章將重點放置於如何時做出一個有彈性、容易擴充新功能、容易維護的程式架構的設計，以及編輯器的非功能需求。在編輯器非功能性設計需求方面特別提到以下幾點：

- 改善下載方式增加使用意願
  - 採用編輯器分頁呈現 - 易將程式分割成多個片段
- 提升樣板擴充性與多樣化

- 採用樣版模組化設計 - 易擴充新的多樣化樣板

此章節將依序先介紹整個程式的設計架構之後再針對各需求進行開發。

4.3.2.1.程式架構設計

考慮到編輯器各功能由鬆散低相依性的模組組成，並樣板視為是編輯器功能的一部分，在編輯器執行時動態載入所需功能或樣板模組，也就是說將一個應用程式封裝檔載入到另外一個應用程式封裝檔中執行。程式撰寫採用複合應用程式設計架構與其程式庫協助開發 (Composite Application Library; CAL) 。其好處為系統可以更容易演進，隨著系統需求的變更，新模組新增至系統時所面臨的阻力，會比在非模組系統中更少，現有的模組可以更獨立地發展，因而提升可測試性。模組可由不同團隊來開發、測試及維護 [9]。主要程式各主要模組互通抽象概念如下圖：

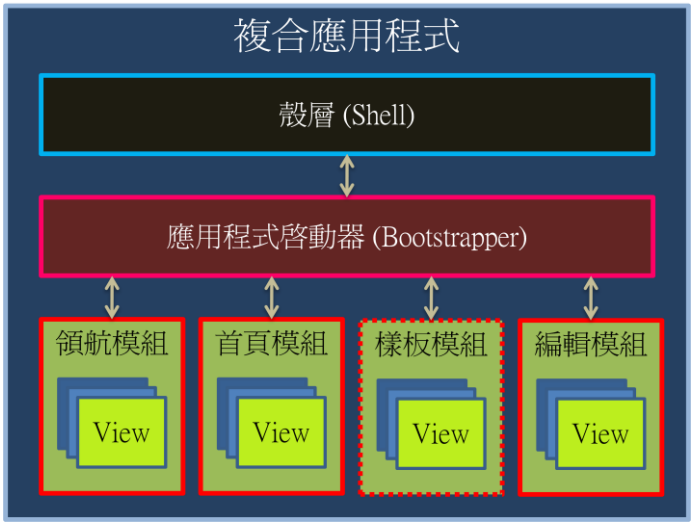


圖 41：編輯器與樣板程式架構抽象概念圖

Shell 如同作業系統裡的觀念一樣，為編輯者與各編輯器功能及模組的界接。應用程式啟動器為將預定使用的模組先註冊至編輯器當中。各樣版模組在編輯器一開啟動時並不載入，等待使用者下命令後才將所需樣板載入。Client 端編輯器與樣板複合應用程式更細節的互通模式如下圖：

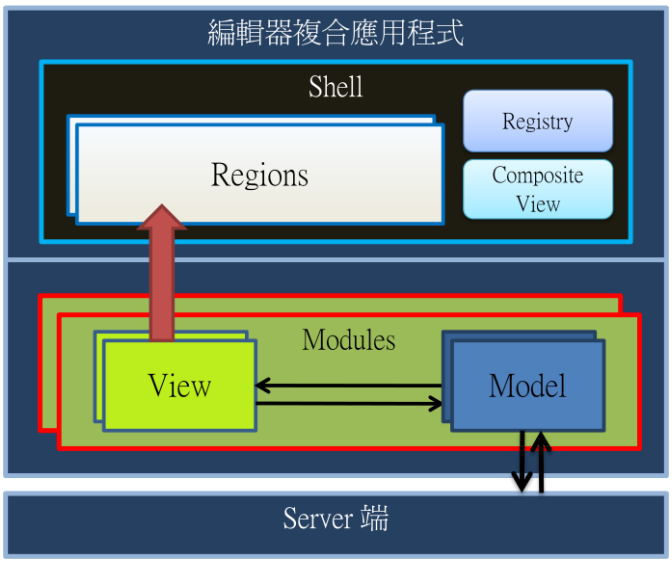


圖 42：Client 端編輯器與樣板元件間溝通圖

EzShow 在 Shell 檔案裡面規劃出兩個 Region，一個為 MenuRegiony 做為存放主選單。另一個為 WorkRegion，將要執行的頁面或樣版載入於此，如下圖所示。

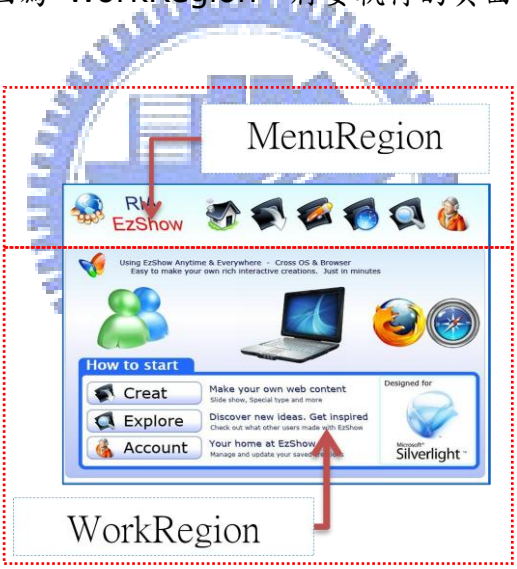


圖 43：Client 端編輯器之 MenuRegion 與 WorkRegion

Registry 為上張圖應用程式啟動器的動作。這個檔案裡只描述未來有甚麼模組可以被載入，並不是實行載入檢視頁動作的地方。複合檢視 (Composite View) 是在說明 shell 與每個模組的檢視頁都可以由多個 Region 組成。他們的 Region 中可以載入其他模組的檢視頁，所以稱複合檢視。我們將每個模組 (module) 的內容又分為檢視 (View) 與模式 (Model)，如下圖所示：

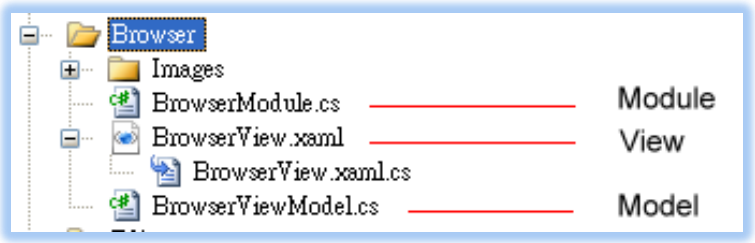


圖 44：Client 端編輯器樣版瀏覽頁程式架構圖

將畫面呈現與實作邏輯分離。模組會構成檢視，而 shell（又稱為主要複合檢視） 上面的區域會放這些檢視。模組之間絕不會直接參考彼此，也不會直接參考 shell，它們會利用服務與彼此及 shell 進行通訊，以回應使用者動作。

介紹完程式主架構後，接下來介紹在使用 CAL 建置我們的複合應用程式時，必須先將幾個核心複合服務初始化。首先要啟動載入器，執行產生複合應用程式時的所有必要功能，啟動載入器 (Bootstrapper) 如同是 CAL 應用程式的 Main 方法。

1. 設定容器

   容器是複合應用程式中所有會用到的應用程式服務之存放區。它負責在需要時插入這些服務。

2. 建立 Shell

   在 Shell 定義出想要讓模組載入的區域。考慮到確保啟動載入器的初始化能夠在顯示 Shell 之前完成，所以 Shell 由應用程式專屬啟動載入器內的 CreateShell 方法所建立，而不直接宣告在 App.Xaml 中。

3. 載入模組

   CAL 包含 ModuleLoader，它會載入每一個模組的組件，然後加以初始化。模組可對其他模組指定相依性。ModuleLoader 將建置相依性樹狀結構，並根據這些規格，依正確順序將模組初始化。

後續持續擴充系統功能的方法：

1. 製作出所需功能的模組
2. 將模組載入至系統
3. 啟動模組，呈現使用

圖 45：將新功能模組載入至編輯器中實做順序說明

4.3.2.2.編輯器分頁呈現

為了降低使用者下載模組與啟動編輯的時間，於是我們將程式模組個別封裝成應用程式封裝檔 (.xap)，並將檔案較大的圖片獨立於應用程式封裝檔之外，讓網頁上的文字相關資料可以快速立即呈現，而圖片在接著動態載入。實做編輯器以下載資料流的方式動態載入所需模組達到分頁呈現。

編輯操作過程中，當使用者切換至編輯頁時，編輯器會動態將編輯頁的檢視頁面載入至欲顯示的 WorkRegion 區域。在擇定樣版後，將樣板模組動態載入至編輯器中欲顯示的區域，架構如下圖所示：

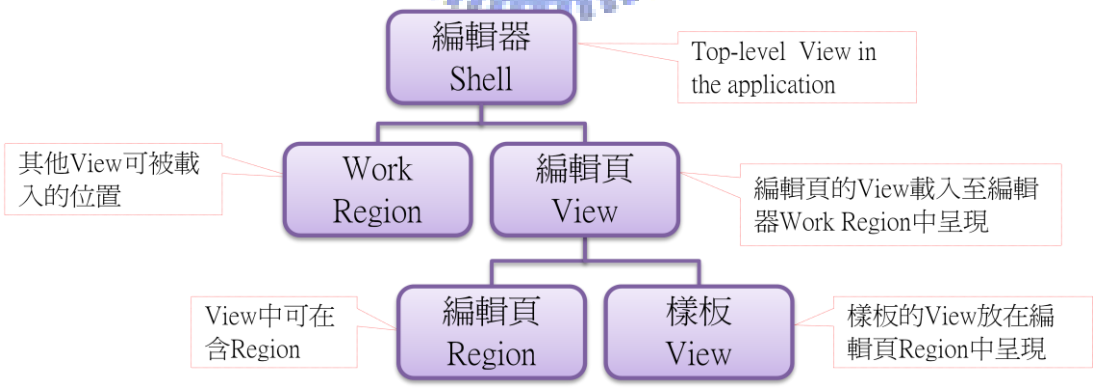


圖 46：複合檢視實做例子架構圖

4.3.2.3.編輯流程功能模組

編輯器包含了五大模組，樣板讀取模組，樣板載入模組，樣板編輯模組，樣板儲存模組，瀏覽模組等五大模組，其架構圖如下：

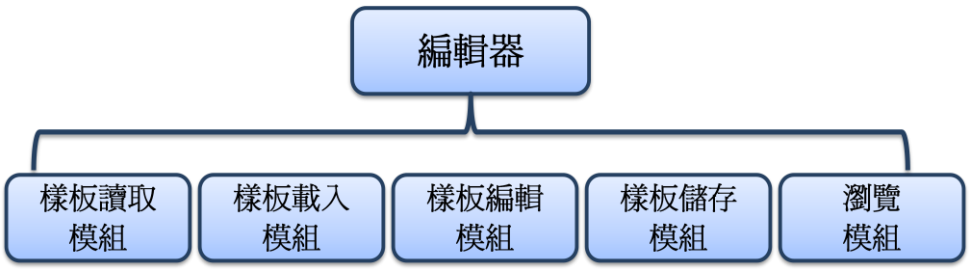


圖 47：編輯器之模組架構圖

各模組環環相扣，在樣板讀取模組由編輯器向伺服器要求下載樣版清單的資料，然後列出給使用者觀看，以便讓使用者選擇所須樣版。當使用者點選所須樣板後，接著執行樣板載入模組，此模組可讓使用者任意載入所須樣板。當完成下載樣版至編輯器後，樣板編輯模組才會正式啟動。樣板編輯模組會展示所載入的樣板，並且列出可編輯替換的元件與其屬性，以便提供使用者做編輯替換動做。使用者在完成編輯替換後，點選儲存，樣板儲存模組即可以動作。樣板儲存模組，是儲存使用者編輯後的所有屬性值，如此一來本次所編輯的樣板，即可以在瀏覽模組中呈現，以讓後續的瀏覽模組繼續運作，讓使用者可讀取完成的作品。

編輯器內部各模組間互動之流程圖如下圖所示：

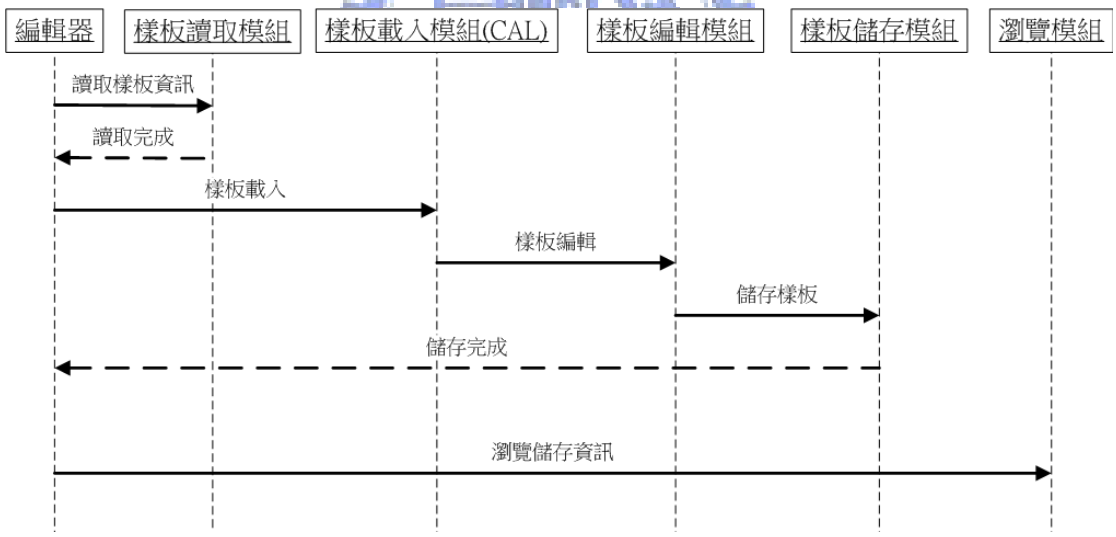


圖 48：編輯器流程圖

在最後儲存資料時，必須要呼叫到資料庫模組。樣板資料庫模組是一個標準的 SOA 程序，採用 WCF 技術實做，藉由呼叫所提供的介面來完成資料儲存或是讀取的功能。本模組分有四大塊功能以及一個實體資料庫，四大功能分別為：樣板列表讀取模組、樣板列表儲存模組，樣板細部資料讀取模組、樣板細部資料儲存模組。各模組皆會對應到實體資料庫，用以讓資料可以永續儲存，各模組間並沒有相依性，可獨立運作。

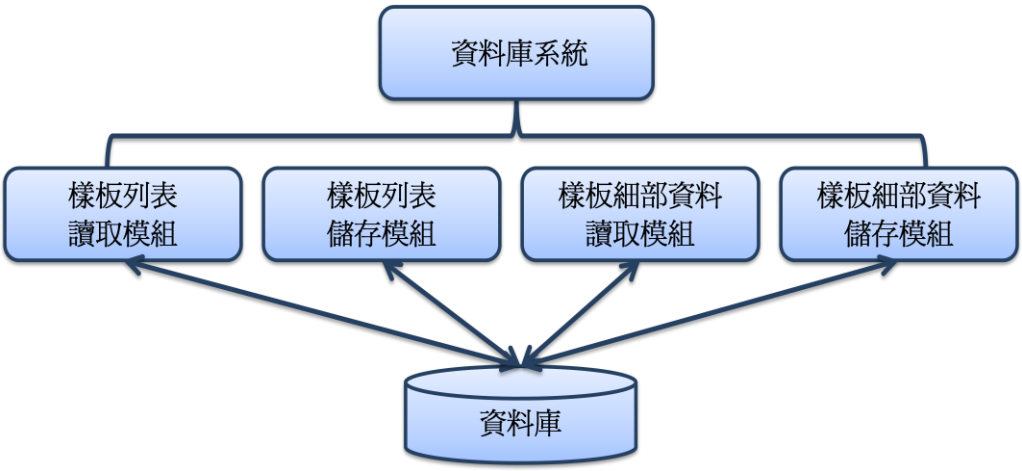


圖 49：資料庫系統之模組架構圖

資料庫系統內部各模組間互動之流程圖如下圖所示：

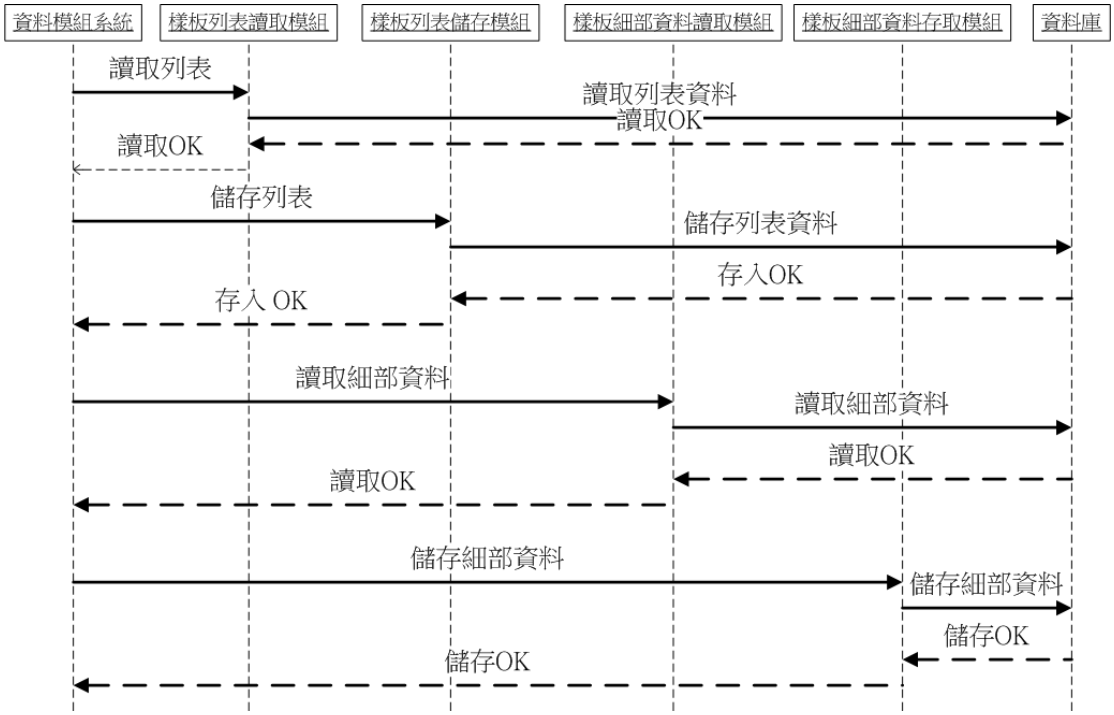


圖 50：資料庫系統流程圖

4.3.2.4.檢視流程功能模組

使用者在編輯完成後，會產生相當多的作品檔案，這些作品檔案皆存在資料庫內，透過檢視模組從資料庫讀取出樣板，然後列出給使用者選擇所要展示或觀賞的樣板。本模組分成五大模組，瀏覽模組、樣板載入模組、樣板細部資料讀取模組、樣板執行模組、樣板展示。使用者在選定所要觀看的作品後，作品會由樣板載入模組呼叫。樣板會呼叫自己本身，在樣板執行模組會去讀取使用者所選擇的樣板編輯的細部資料。在取得細部資料後，樣板以及這些資料就會被樣板執行

模組整合執行，即是完整的作品呈現。也就是透過樣板執行模組及樣板展示模組將細部資料呈現在使用者眼前。

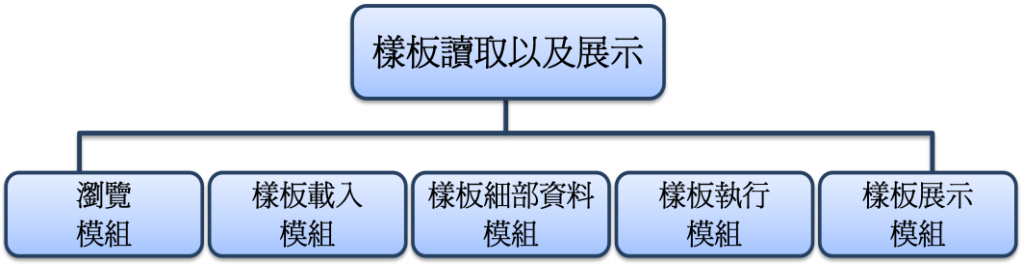


圖 51：檢視模組-樣板讀取及展示架構圖

在檢視作品時，各功能模組之間的溝通如下圖所示：

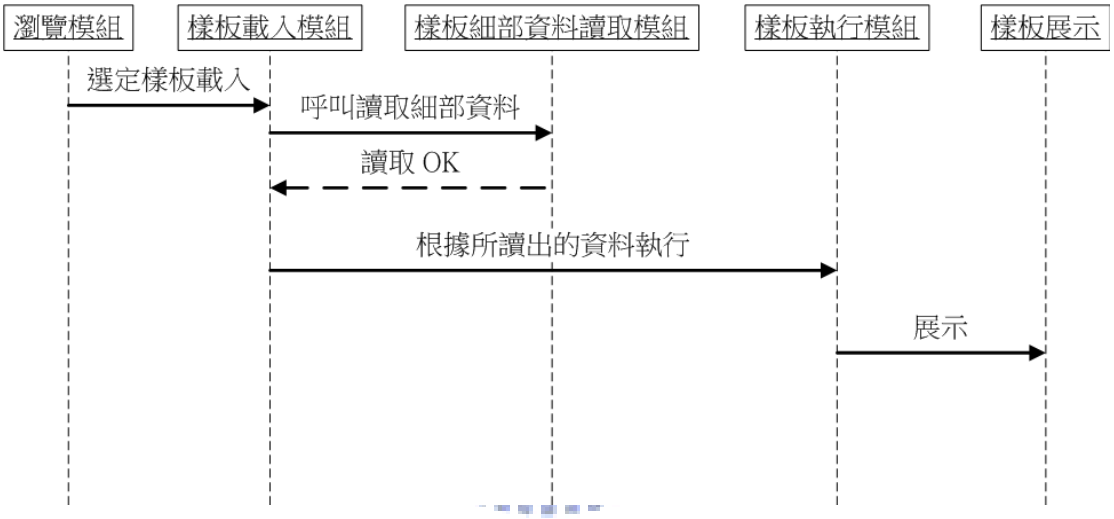


圖 52：檢視模式-樣板讀取及展示流程圖

4.3.3樣板與智慧型標籤 (Smart Tag)

在樣板的製作過程當中包含製作資料替換編輯邏輯 - 智慧型標籤。

圖 53：系統架構之樣板作業部分

在程式撰寫階段，遇到如何才能支援樣板的多樣性不受編輯器限制，將配合下圖來說明。

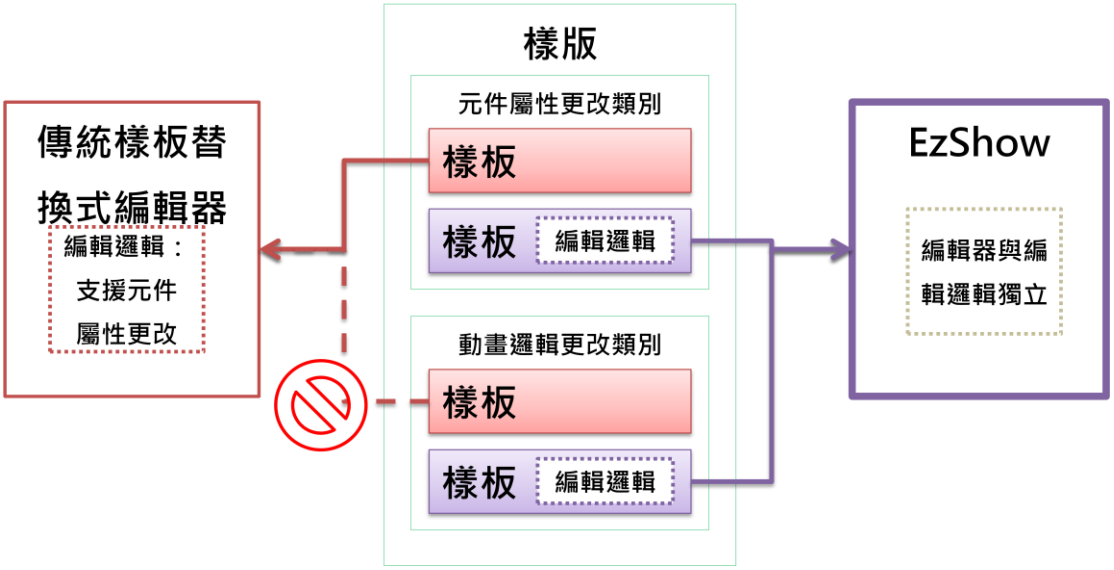


圖 54：EzShow 與傳統編輯器對樣板的通用性

編輯器與編輯邏輯的實做分兩種討論：

- 上圖左側 - 傳統編輯器：將替換邏輯實做在編輯器之內
- 上圖右側 - EzShow 編輯器：將替換邏輯與編輯器獨立

上圖中間 - 樣板分為兩大類，包含單純的元件屬性更改類別與動畫邏輯更改類別。EzShow 樣板包含資料替換邏輯。

常見元件不外有文字、圖片、影片、聲音等，非常容易實做出元件屬性的替換編輯邏輯。但動畫邏輯，也就是劇情邏輯會隨著樣板的類型與提供的故事劇情有所不同，可以設定或替換的內容是無法評估的。採用傳統編輯器將資料替換邏輯做在編輯器上的做法，在想增加多樣性樣版就會受到限制。如：無法設定特殊的動畫劇情。如果想新增一個較特殊的類別就必須將編輯器的編輯邏輯整個重新規劃與實做。使用者端也必須重新安裝或更新，操作介面必然也會更動。當增加的功能越來越多，編輯器的大小也會隨之增加，造成使用者負擔第二層的加重。

但採用編輯邏輯與編輯器獨立的做法，並不會造成使用者的負擔，相反的，不需去更動到編輯器，可非常容易提供使用者更多樣版的選擇。而且，考量到使用者並不會在每次編輯當中都用到所有的資料替換編輯邏輯，只會用到樣板上可設定的部分，所以本研究將樣板的替換編輯邏輯放置在樣板模組內一併製作。

接下來介紹智慧型標籤，在樣板替換編輯過程中採用智慧型標籤機制是為了增加編輯軟體的可操作性，也就是操作的容易度。協助使用者在編輯過程當中，最常遇到找不到元件屬性設定位置等問題。元件指的是樣板裡各種被編排的東西，

如：文字、圖片、影片、聲音等。

智慧型標籤是一種協助工具，我們將資料替換邏輯寫智慧型標籤裡，在我們的實做中，呈現包含兩個部份：

- 提示圓點：告知使用者這個元件是可以做替換或相關屬性設定
- 屬性設定介面：元件可設定的屬性選項

當使用者將滑鼠移至提示圓點，點選要設定元件的智慧型標籤時，會顯示一個快顯功能表，也就是屬性設定介面，放著許多該元件的屬性設定選項。編輯器、樣版與智慧型標籤的關係以圖表示如下：

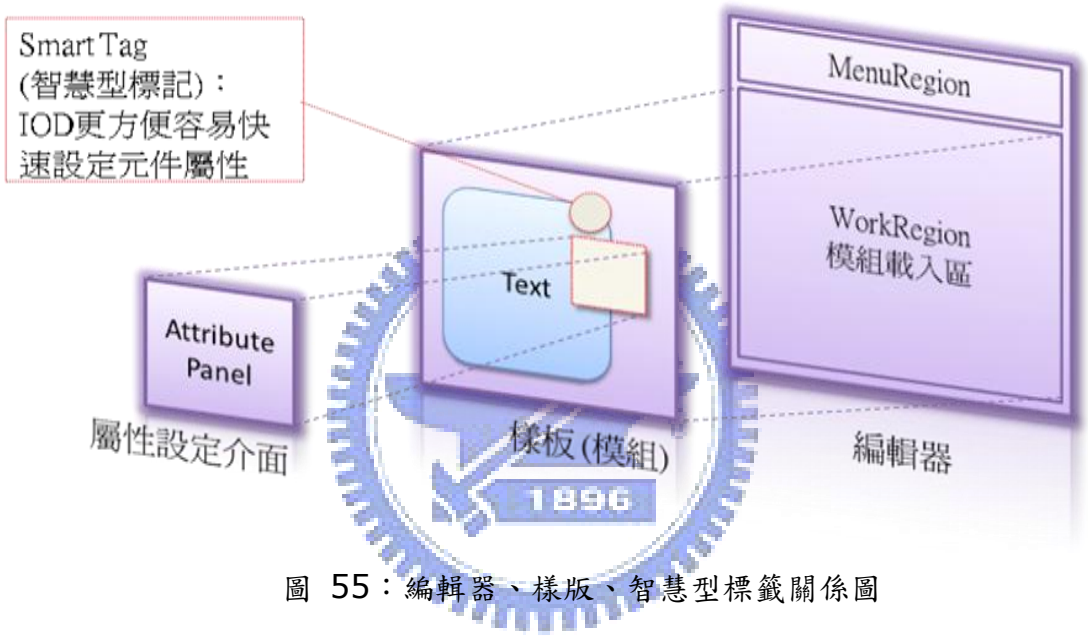


圖 55：編輯器、樣版、智慧型標籤關係圖

每個元件的智慧型標籤分為三部分實做，其呈現圖如下圖所示：

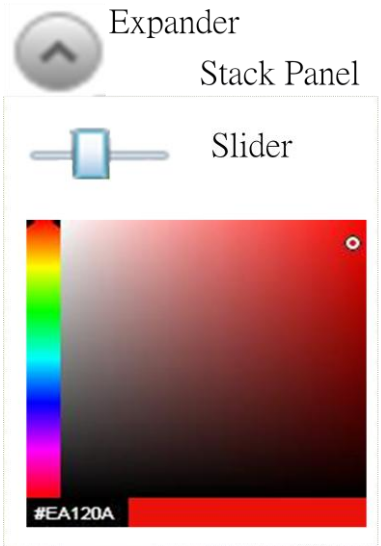


圖 56：文字元件的屬性設定介面

- Expander
  是用來指示此元件是可設定元件屬性，使用者可以從這裡開始對樣板的內容做替換與相關設定
  - Expander 的特性為初次點選會顯示某種物件，在這裡物件指的是設定屬性的控制項，再點選會隱藏該物件
  - ➢ 所以當可設定元件屬性不止一種時，本論文選擇使用這種控制項來實做智慧型標籤，將 Expander 視為提示圓點。
- Panel
  是用來放置各種屬性設定的控制項，也就是 Controller
  - Panel 類別的物件特性在於可以一次收納很多個不同種類的物件在一起，以方便管理
  - ➢ 由於上述的 Expander 一次只能針對一個物件做處理，但是每個元件的屬性設定控制項並不會只有一種，所以我們必須選擇有 Panel 特性的控制項來幫助我們實做智慧型標籤，在本研究中選用 Stack Panel 控制項
    - Stack 的特性在於會將物件一個一個依序由上往下排好
- Controller
  各控制項是用來設定元件屬性值與回覆更動的屬性值給樣版
  - 各種不統的控制項有不同的特性
  - ➢ 本論文各種屬性設定控制項，將依使用者操作電腦的習慣與方便性做為選擇

以上面兩張圖為例子來看，是針對 Text 類型的元件做屬性設定，智慧型標籤提供了兩種屬性設定，包含文字大小與文字顏色。

在文字大小的設定方面，有三種方式可以實做：

- TextBox：讓使用者輸入文字大小的尺寸
- ListBox：讓使用者點選下拉選單提供的尺寸選項
- Slider：讓使用者將指標移動到所需尺寸

以下為各控制項的操作方法：

- TextBox：滑鼠點選控制項輸入欄位 ＋ 鍵盤輸入動作
- ListBox：滑鼠點選控制項 ＋ 滑鼠點選下拉選單選項
- Slider：滑鼠點選 Slider 的指標

在選擇實作控制項考慮到使用者的方便性：

- 採用 TextBox 無法預期使用者透過鍵盤會輸入什麼資料，可能會增加無法評估的錯誤發生機率。
- 由使用者平常操作電腦的習慣來看，連續使用滑鼠比起使用滑鼠與鍵盤的操作模式來的更為順暢。
- 而完成設定所需點選滑鼠左鍵的次數越少越方便

所以在設定文字大小方面採用 Slider 控制項做為實做。使用者在編輯文字大小屬性時，移動 Slider 指標，而樣版上的文字也會立即顯示出效果。

在文字顏色設定部分因為沒有現成的控制項可以使用，所以我們必須自己時做一個選色器，本研究採用複合應用程式的複合顯示特色來實做選色器，將選色器的屬性設定元件採獨立模組化設計，增加其可重複使用性。實作方法為：

1. 實做一個選色器模組
2. 在會用到選色器功能之元件的屬性設定 Stack Panel 上面規劃一個 WorkRegion，供選色器模組中的檢視畫面放置
3. 將選色器模組中的檢視頁面指定至 WorkRegion 中顯示

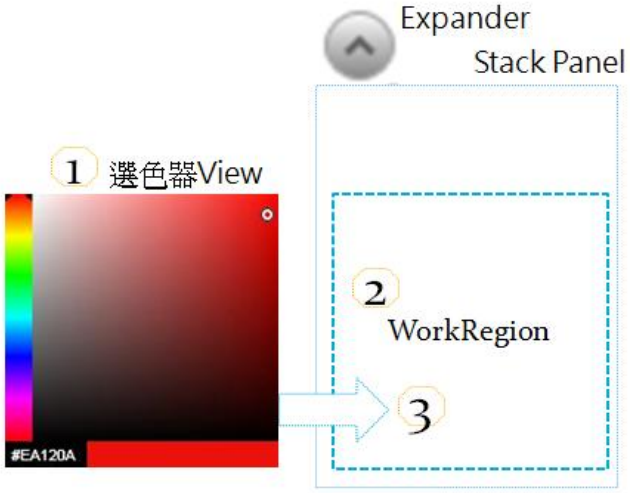


圖 57：選色器控制項實做概念圖

接著，當設定文字大小的 Slider 控制項或文字顏色的選色器接收到使用者更動屬性事件，會將更動值傳遞至樣版上，樣版接收後依值改變樣板上元件的屬性，然後立即呈現給使用者觀看。

下列採用圖文聲-多媒體廣告樣板說明樣板與智慧型標籤整合後的程式架構，在此樣板的智慧型標籤當中將採用到剛剛說明的選色器模組。

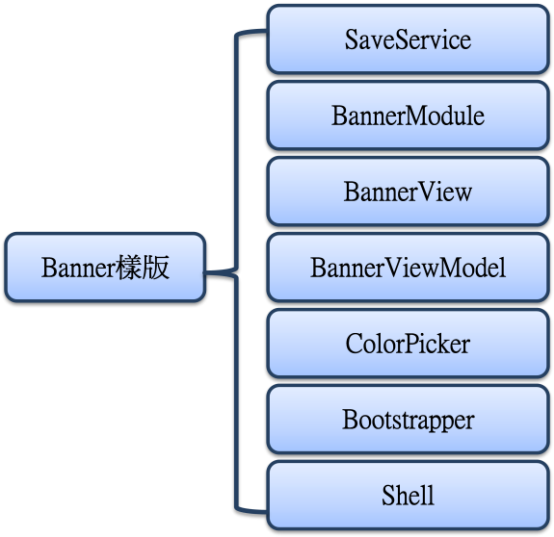


圖 58：圖文聲-多媒體廣告樣板與智慧型標籤架構圖

SaveService 模組負責的工作為當使用者編輯完成後，將編輯者替換資料上傳至伺服器。BannerModule 為整合 BannerView 與 BannerViewModel 供程式架構上更上層呼叫此樣板時所使用。BannerView 負責樣板檢視呈現的部分，接受編輯者設定值立即呈現效果於畫面上、BannerViewModel 可存放此樣板中操作控制邏輯、ColorPicker 則是選色器，當編輯者需要使用到選色器實，其選色器會由 BannerView 呼叫使用。

4.4 編輯器下載與操作流程

完成整個系統的實做後，接下來看使用案例圖。

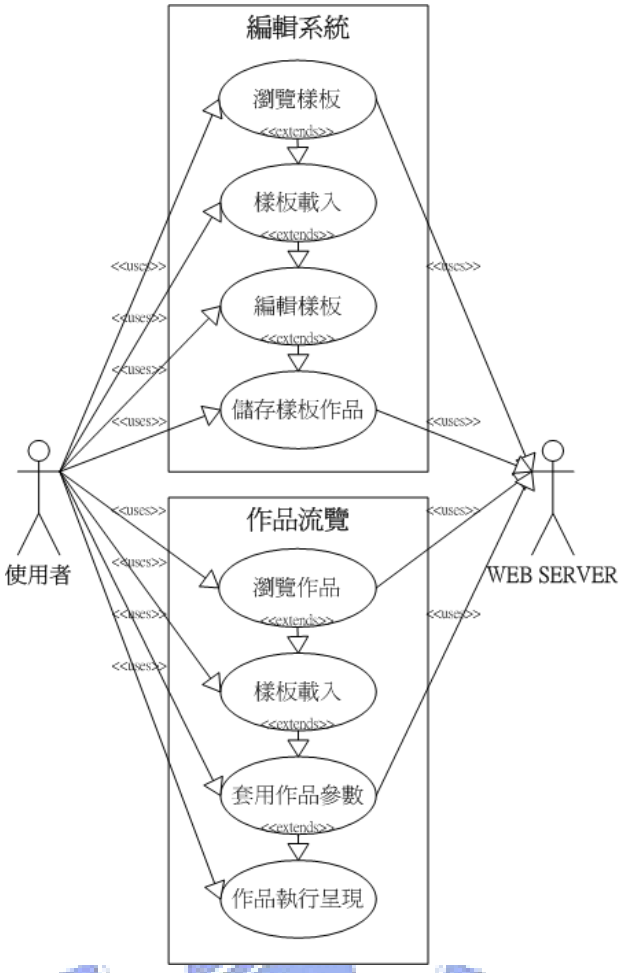


圖 59：編輯系統與作品瀏覽的使用案例圖

使用者如何使用編輯器的操作流程如下圖所示：

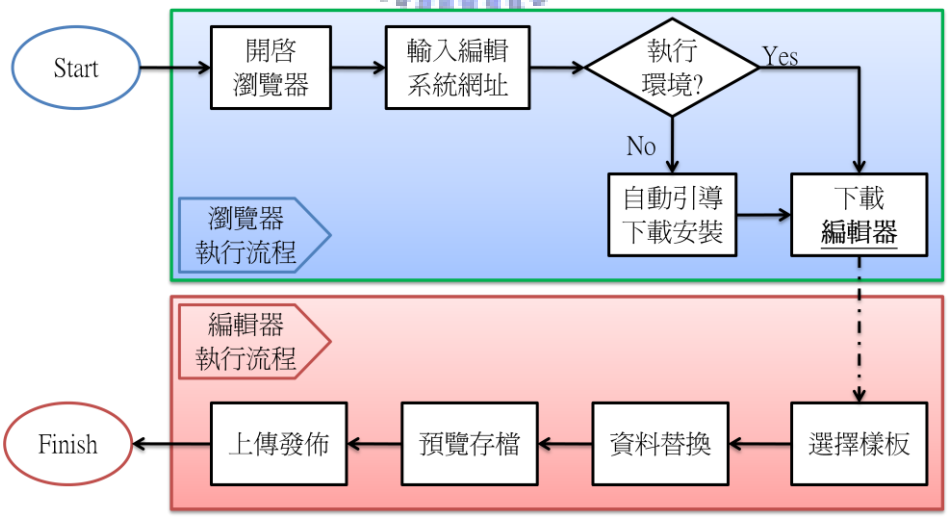


圖 60：編輯器下載與操作流程圖

分兩個階段來看，瀏覽器執行流程與編輯器執行流程。

- **瀏覽器執行流程**

使用者首先開起電腦裡的瀏覽器，輸入編輯系統的網址，透過網際網路連線到編輯系統伺服器端。因為我們的檔案為 silverlight 格式，這時候伺服器會檢查使用者是否有足夠的環境可以執行此類型檔案。當使用者未安裝 silverlight 2.0 runtime 時，會給予使用者下載安裝的頁面，詢問使用者是否自動引導安裝。安裝好執行環境後，使用者將可順利將編輯器下載至使用者電腦裡。

- **編輯器使用者執行流程**

使用者透過網頁執行編輯器，第一步驟為選擇所需樣版，接著進行資料替換的編輯，完成編輯後進行預覽與存檔動作，最後使用者可以選擇是否將檔案在其他網頁中顯示，如需要就可以選擇發佈，取得檔案的網址，完成整個編輯流程。編輯操做循序圖如下所示：

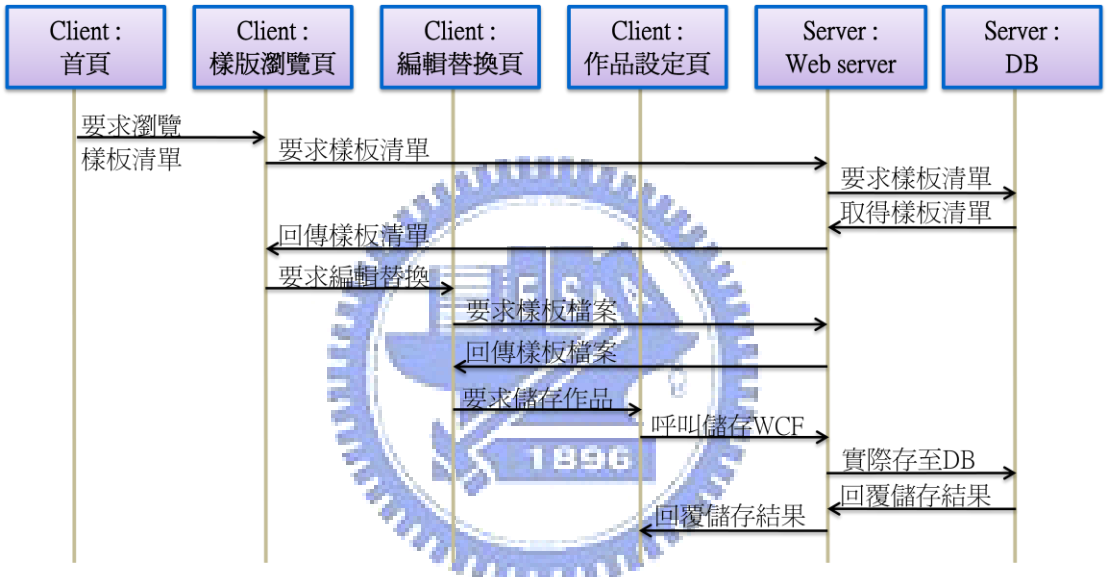


圖 61：編輯器編輯操作流程圖

- **編輯器瀏覽者執行流程**

在編輯器中作品瀏覽頁瀏覽作品的流程圖如下所示：

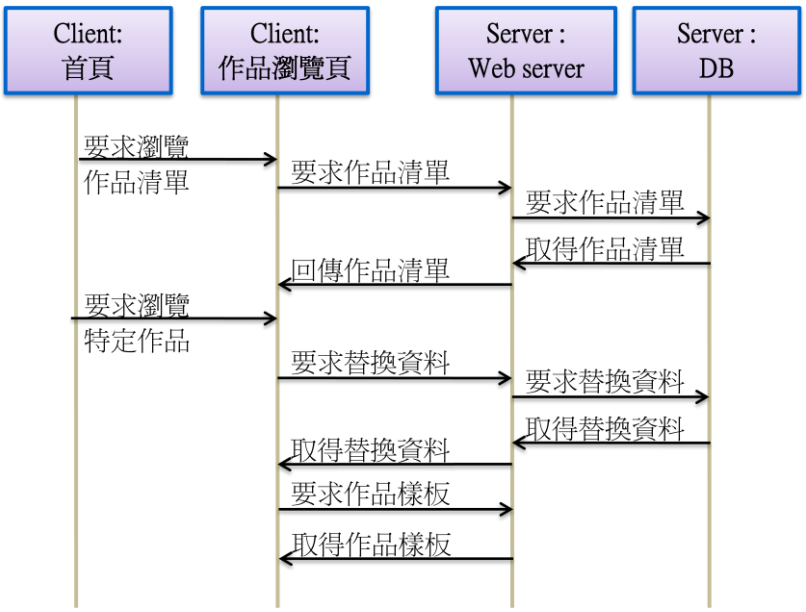


圖 62：編輯器瀏覽作品流程圖

使用發佈網址瀏覽作品的流程圖如下所示：

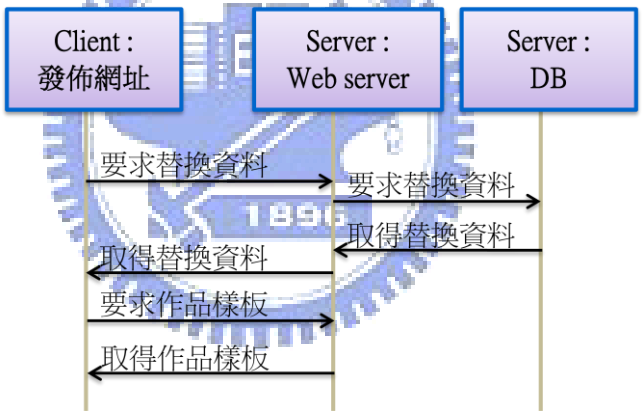


圖 63：使用發佈網址瀏覽作品流程圖

伍、 應用範例

本系統採用 RIA Silverlight 技術開發，有別於傳統平台依系統特性做移植，並結合樣板與 Smart Tag 編輯模式開發 EzShow。展示情境圖如下圖，分兩個環境各別進行樣板的展示與功能測試，證明同個系統在兩個異質平台上呈現與功能執行皆順利。

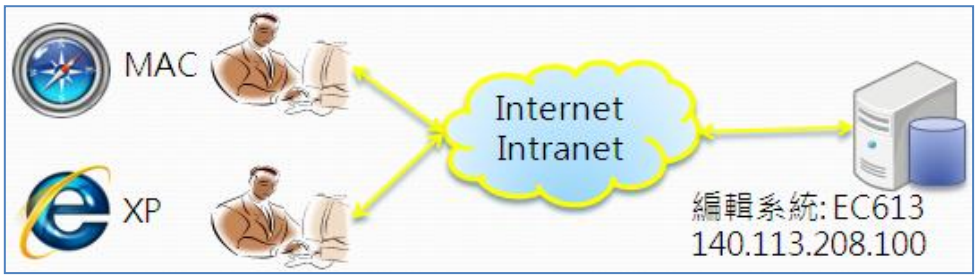


圖 64：展示範例架構圖

- 環境一：XP+IE

圖 65：XP+IE：測試機器的系統環境

- 環境二：MAC + Safari

圖 66：MAC + Safari：測試機器的作業系統環境

5.1 在 XP 使用 IE 測試

因傳統網路應用程式不易做到多變化外型的播放器，在本章節將採用骨頭造型播放器做為展示。首先，輸入系統網址開啟首頁，其 MenuRegion 與 WorkRegion 呈現效果如下圖所示：

圖 67：XP+IE：測試首頁呈現

測試在 MenuRegion 的主選單是否可正常執行，包含測試選項的滑鼠偵測功能、提示功能、滑鼠事件功能，如下面兩張圖所示：

- 當滑鼠移過 (Mouseover) 選項，選項的圖片會放大
- 當滑鼠停留在選項上面，會出現選項功能提示
- 當滑鼠點選了選項之後，選項的圖片會翻轉並縮小呈現

圖 68：XP+IE：測試按鈕的滑鼠偵測功能與提示功能

圖 69：XP+IE：測試按鈕的滑鼠點選事件偵測功能

測試首頁動畫是否可正常執行。

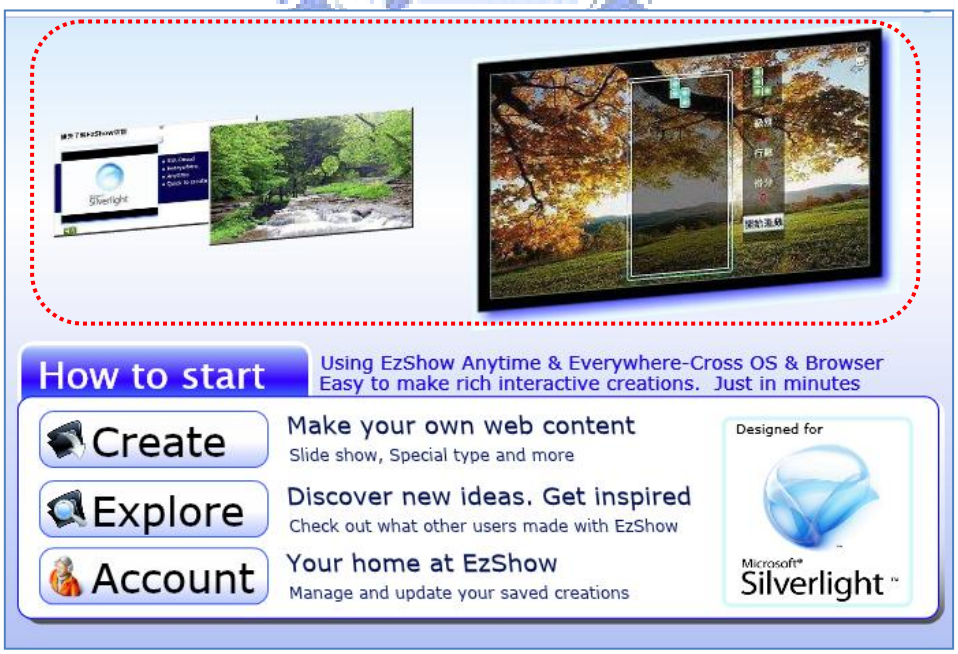


圖 70：XP+IE：測試首頁內容之動畫效果

測試首頁輔助說明 (How to Start) 選項與編輯說明是否可正常執行。

圖 71：XP+IE：測試首頁編輯器功能介紹的快捷按鈕

當首頁功能測試完成後，接著開始編輯功能，Step1：選擇樣版，目前系統提供三種類別的樣板供選擇。在樣板列表可以看到樣板的縮圖，點選所須樣版會顯示出樣板的介紹，當擇定後點選開始編輯按鈕，即可以進入到編輯畫面。

圖 72：XP+IE：測試樣版選擇頁四大功能

進入到編輯畫面後，首先使用者端的編輯器會至伺服器端下載擇定的樣板，之後將樣板的顯示頁面呈現在 WorkRegion 中，由下圖智慧型標籤可得知可替換的元件。

圖 73：XP+IE：測試遠端下載樣版與智慧型標籤顯示呈現效果

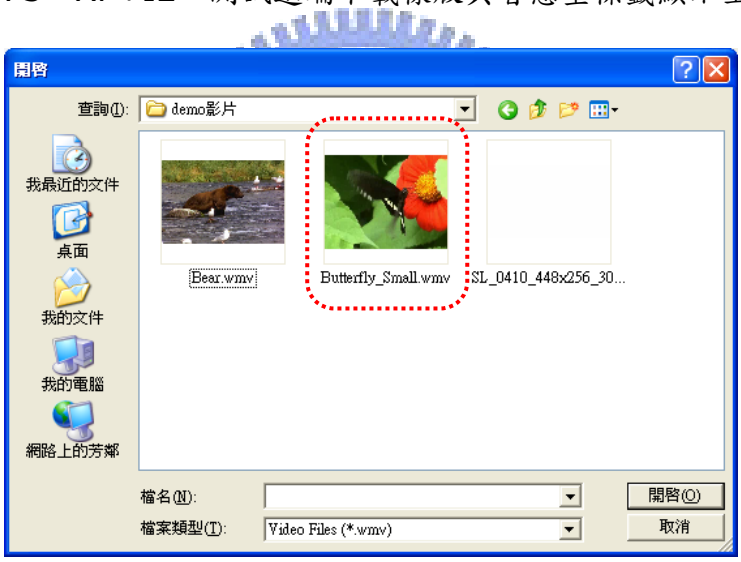


圖 74：XP+IE：測試替換圖片功能

擇定替換資源後，將其替換，其替換完後的效果如下圖所示，下圖為在編輯模式當中。

圖 75：XP+IE：測試替換資源後效果呈現

可任意切換編輯模式或替換模式，下圖為預覽模式，其預覽按鈕顯示為無法使用。當預覽完成編輯後，按下存檔按鈕將替換資料存至遠端伺服器上。

圖 76：XP+IE：測試換圖後編輯檢視顯示與存檔效果

當按下存檔按鈕後，在上傳資料的同時，會將頁面切換至作品設定與發佈畫面。還在上傳中會顯示『作品資料儲存中…!!請稍候』等字樣，且儲存作品介紹

的按鈕也是無法執行的，直到儲存完畢後使用者才可以繼續進行下一步驟。

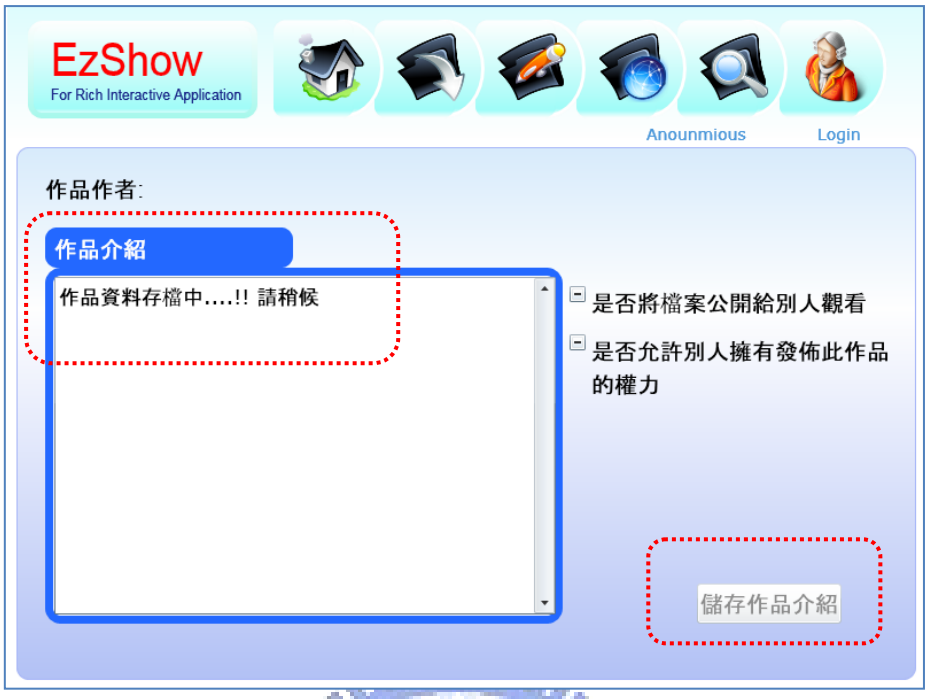


圖 77：XP+IE：作品設定與發佈頁面載入

頁面切換至作品設定與發佈畫面，使用者為設定即離開，相關資訊將為預設值。可輸入作品介紹、是否公開檔案、是否允許別人發佈等設定。

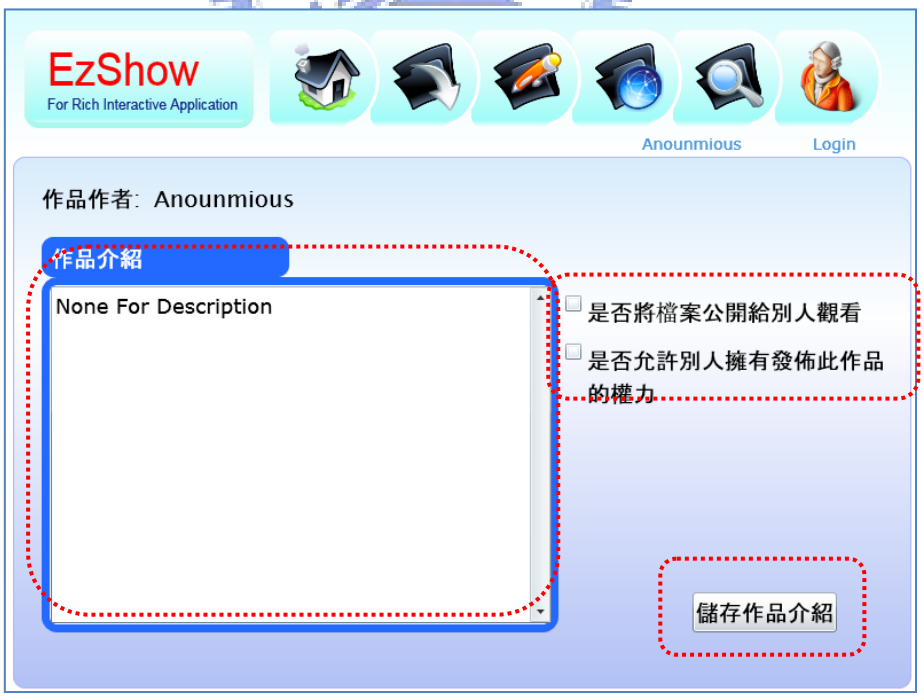


圖 78：XP+IE：作品設定與作品發佈頁面呈現

完成輸入與權限設定，按下儲存作品介紹按鈕後，按鈕文字會切換顯示成儲

存中請稍候，待儲存完成後會自動切換至作品瀏覽頁面。

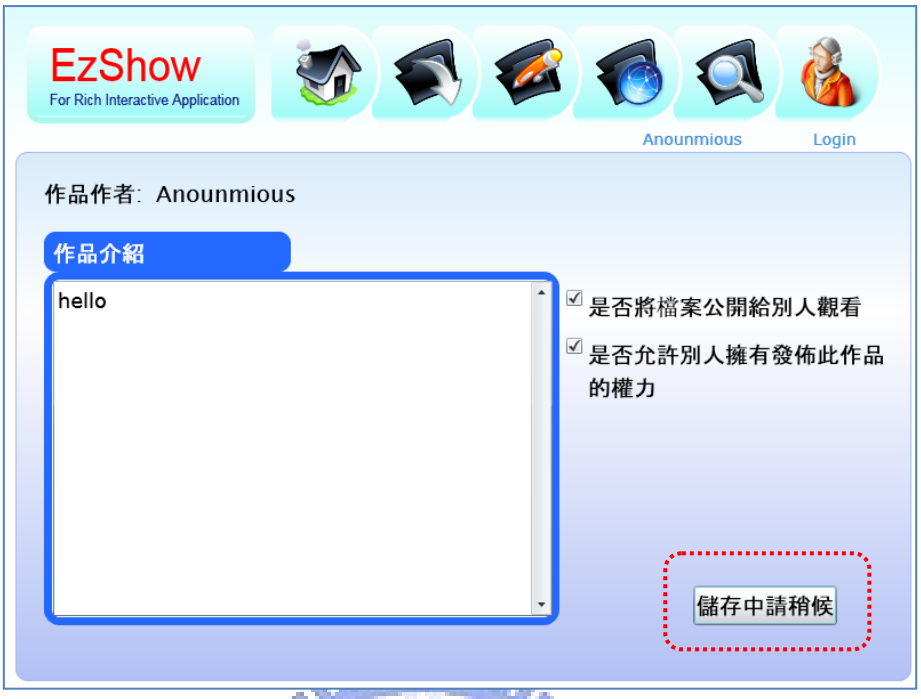


圖 79：XP+IE：儲存作品設定與發佈權限頁面

在作品瀏覽頁面第一筆呈現出剛剛完成的作品。

【作品的作者】	【上傳時間】	【修改時間】	【作品介紹】	【瀏覽】	【發佈】
Anounmious	7/15/2009 1:31:1	7/15/2009 1:38:2	hello	瀏覽	取得發佈網址
Anounmious	6/19/2009 11:56:	6/19/2009 11:56:	None For Description	瀏覽	取得發佈網址
Anounmious	6/19/2009 11:49:	6/19/2009 11:49:	XP + IE	瀏覽	取得發佈網址
NH	6/18/2009 9:04:1	6/18/2009 9:04:3	Water effect	瀏覽	取得發佈網址
NH	6/18/2009 9:00:2	6/18/2009 9:00:3	image rotator	瀏覽	取得發佈網址
NH	6/18/2009 8:58:3	6/18/2009 8:58:5	Silverlight - bone	瀏覽	取得發佈網址
Anounmious	6/18/2009 8:47:4	6/18/2009 8:48:0	http://www.microsoft.com/	瀏覽	取得發佈網址
Anounmious	6/18/2009 5:27:1	6/18/2009 8:19:1	XP+IE+AD2	瀏覽	取得發佈網址
Anounmious	6/18/2009 4:04:3	6/18/2009 4:04:5	XP+IE+廣告	瀏覽	取得發佈網址
Anounmious	6/18/2009 3:32:3	6/18/2009 3:32:5	XP + IE + Bone	瀏覽	取得發佈網址
NH	6/17/2009 9:12:2	6/17/2009 9:12:4	XP + IE + AD1	瀏覽	取得發佈網址
Anounmious	6/17/2009 5:20:5	6/17/2009 5:21:3	XP + IE + Flickr 39351584@N03 9f0ff3fd560c8c23a7a285ab9df234b7 認證模式： 網站 (http://www.flickr.com/photos/hero7478/sets/72157619651876319/)	瀏覽	取得發佈網址
Anounmious	6/17/2009 5:18:4	6/17/2009 5:19:0	XP + IE + flickr	瀏覽	取得發佈網址
nh	6/17/2009 1:23:0	6/17/2009 1:23:1	MAC + Safari + AD1	瀏覽	取得發佈網址

圖 80：XP+IE：測試作品瀏覽頁面

在作品瀏覽頁面點選剛剛完成的作品之取得發部網址按鈕，即可以立即取得

發佈網址。

圖 81：XP+IE：測試發佈網址呈現的效果

當在新網頁上輸入發佈網址即可以看到作品呈現效果。

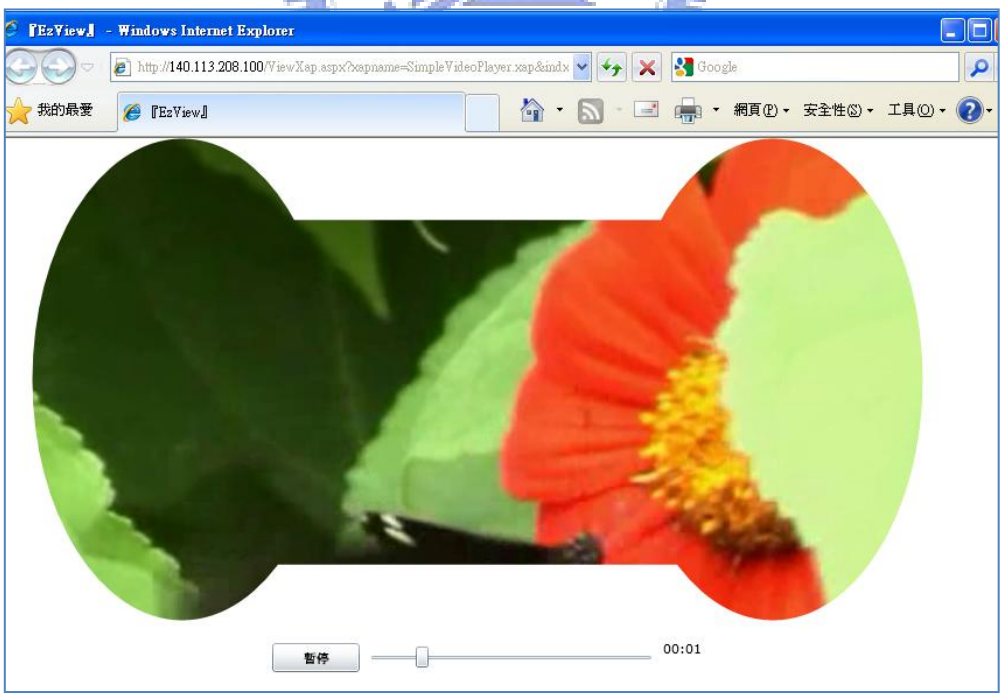


圖 82：XP+IE：測試發佈後檔案呈現效果

最後我們將畫面切換至使用者帳號頁面，測試使用者帳號頁面呈現效果。

圖 83：XP+IE：測試使用者帳號頁面顯示

5.2 在 MAC OS 使用 Safari 測試

本小節測試將按照上小結的步驟依序進行編輯器功能測試，確認相同編輯系統在 XP 與 MAC OS 上皆能完整使用與呈現，在此採用圖文聲-多媒體廣告樣板做為測試，可使用的情境包含廣告、新聞、公告等等領域，下圖為系統首頁。

圖 84：MAC + Safari：測試首頁呈現

和上小節相同，我們先測試系統一些基本功能，下圖為測試按鈕的滑鼠偵測功能與提示功能與下下張圖為測試按鈕的滑鼠點選事件偵測功能。

圖 85：MAC + Safari：測試按鈕的滑鼠偵測功能與提示功能

圖 86：MAC + Safari：測試按鈕的滑鼠點選事件偵測功能

在完成後上步驟後，繼續測試首頁內容之動畫效果以及測試首頁編輯器功能介紹的快捷按鈕。

圖 87：MAC + Safari：測試首頁內容之動畫效果

圖 88：MAC + Safari：測試首頁編輯器功能介紹的快捷按鈕

當首頁功能測試完成後，接著開始編輯功能，Step1：選擇樣版，目前系統提供三種類別的樣板供選擇，包含特殊效果、圖片切換、趣味遊戲等。在樣板列表可以看到樣板的縮圖，點選所須樣版會顯示出樣板的介紹，當擇定後點選開始編輯按鈕，即可以進入到編輯畫面。

圖 89：MAC + Safari：測試樣版選擇頁四大功能

圖 90：MAC + Safari：特殊效果類樣版-1

圖 91：MAC + Safari：特殊效果類樣版-2

圖 92：MAC + Safari：圖片切換類樣板

圖 93：MAC + Safari：趣味遊戲類樣板

本測試擇定採用圖文聲-多媒體廣告樣板做為測試，可使用的情境包含廣告、新聞、公告等等領域，可替換的功能包含圖片、標題文字、內容文字以及聲音。

圖 94：MAC + Safari：測試樣版遠端下載與編輯頁面顯示呈現

下面範例為示範內容文字的智慧型標籤，使用者輸入所須內容、文字粗體、文字斜體、文字大小以及文字顏色等功能。

圖 95：MAC + Safari：文字屬性之智慧型標籤

下圖為完成測試替換圖片與文字功能後的效果。

圖 96：MAC + Safari：測試替換圖片與文字功能

可切換至預覽模式觀看整體效果，並可以測試音效是否撥放正常。

圖 97：MAC + Safari：測試換圖後編輯模式顯示

圖 98：MAC + Safari：在樣板中開啟或是關閉背景音樂的測試

整個作品資料設定頁面的功能操作與呈現和在 XP ＋ IE 上測試效果一致。當按下存檔按鈕後，在上傳資料的同時，會將頁面切換至作品設定與發佈畫面。還在上傳中會顯示『作品資料儲存中…!!請稍候』等字樣，且儲存作品介紹的按鈕也是無法執行的，直到儲存完畢後使用者才可以繼續進行下一步驟。

圖 99：MAC + Safari：測試換圖後編輯檢視顯示與存檔效果

圖 100：MAC + Safari：測試作品設定與作品發佈頁面

下圖為測試瀏覽別人作品頁面，會列出開放瀏覽權限的作品。

圖 101：MAC + Safari：測試瀏覽別人作品頁面

下圖為測試發佈網址效果，直接將網址以物件型式鑲在所須網頁中。

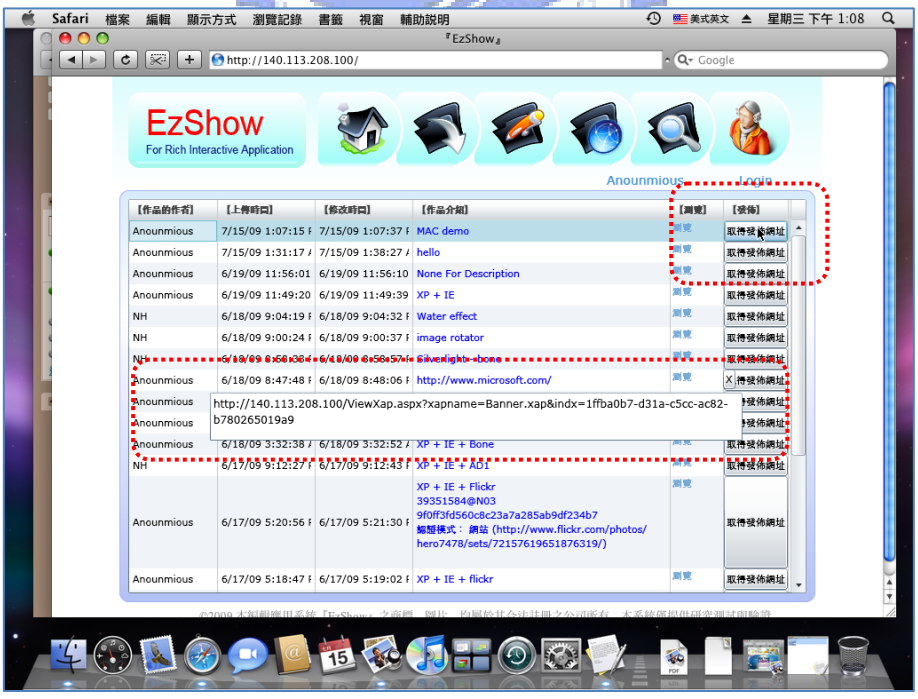


圖 102：MAC + Safari：測試發佈網址效果

下圖為測試發佈後檔案呈現效果，未來作品將以物件型式鑲在所須網頁中，其將呈現的效果如下圖所示。

圖 103：MAC + Safari：測試發佈後檔案呈現效果

最後，測試使用者帳號頁面顯示。完成整個在 XP + IE 與 MAC OS + Safari 上的一致性測試。

圖 104：MAC + Safari：測試使用者帳號頁面顯示

陸、 結論

6.1 總結

本研究依論文提及四大特性：跨平台跨瀏覽器、操作互動流暢、易上手、編輯方便快速，也考慮到使用者的編輯習慣、人機介面設計、編輯功能與樣板的擴充性、系統架構彈性與安全性，設計及實做出一套 RIA based EzShow 編輯系統。並設計及實做三種類型傳統 Web 應用程式不易做出與呈現的互動式樣版：特殊效果、圖片轉換、趣味遊戲等。經由在微軟 XP 作業系統配合 IE8 瀏覽器測試以及 MAC OS X Intel 配合 Safari 瀏覽器測試功能操作與顯示皆相同，證明本系統在有執行環境的情況下可達到跨平台與跨瀏覽器，降低經常會在不同時間地點接觸不同台電腦會遇到的一些不方便。

系統特色詳細敘述如下：

〔1〕 藉由 Rich Interactive Application (RIA) Microsoft Silverlight 技術解決在異質作業系統下應用程式人機互動方面的問題，減少程式開發與移植所花費的時間及成本，讓編輯器同時擁有桌機應用程式的高操作互動性與 Web 軟體免下載安裝特性，編輯者可隨時隨地網路上做編輯。

〔2〕 在編輯模式方面導入了樣板替換機制取代元件排版機制，協助使用者不用經過長時間學習專業美工排版，依然可以快速做出豐富的多媒體檔案。

〔3〕 採用智慧型標籤機制，協助編輯者在編輯的時候不用花時間去找尋所需設定元件的屬性設定介面位置在編輯器的何處。協助編輯者操作方便又簡單，提升持續使用本編輯器的意願。

〔4〕 在使用者需要的時候，再將所需檔案或功能下載至編輯者端，用動態分散下載方式降低編輯者等待時間，提升使用意願。

〔5〕 本研究在開發系統時，特別考慮到編輯器未來的擴充性與彈性，採用 CAL 程式架構，將系統功能模組化、資料替換編輯邏輯與編輯器獨立分開、資料替換編輯邏輯與樣板合併實做、樣板設計成獨立模組，其好處為各功能或樣板可由各團隊獨立開發，之後視需求再整合在一起，提升開發的效率與可維護性，也提升樣版的多樣性。

〔6〕 也考慮到編輯系統未來的擴充性與安全性，採用 SOA 系統架構，將整套系統的功能依需求分為 Server 端與 Client 端，需高權限動作

的功能放至 Server 端，其餘功能放至 Client 端，降低 Server 與 Client 之間所需傳遞的資料量，提升編輯過程的流暢性。

6.2 未來發展方向

在此我們將提出未來可能發展的方向及尚未解決的問題。本研究採用 SOA 系統架構與 CLA 程式架構實做 EzShow 系統，提升系統功能安全性、擴充彈性與樣板設計多樣性，未來可更加容易將以下功能持續導入系統。

〔1〕 本研究採用樣板的編輯模式來編輯多媒體檔案，未來可持續擴充各類型的樣版，增加系統樣板多樣性，例如名片、卡片、新聞、特殊應用等

〔2〕 提供編輯者更容易發佈檔案的機制，將編輯系統與現有的網路平台做結合，例如：樣板國度, Xuite, Wretch, Blogger。協助編輯者可以非常方便快速直接將完成的多媒體檔案發佈至各式平台。

〔3〕 編輯者透過網路平台分享知識，瀏覽者亦可透過網路輕易取得該多媒體檔案來閱讀、複製、列印、甚至是編輯，這對於保護文件原始作者的智慧財產來說是一項重大打擊，因此未來希望能結合 DRM 機制，維護原始作者該有的權利。

〔4〕 編輯者並不可能隨時都在有良好網路品質的環境下，所以思考如何能提供 Client 端編輯器離線存取功能是非常重要的。

〔5〕 將編輯系統與雲端計算、雲端服務做結合。

參考文獻或資料

〔1〕 Internet World Stats，"INTERNET USAGE STATISTICS"，[On-line].Available: http://www.internetworldstats.com/stats.htm，March 2009

〔2〕 張筱靈，"移植多媒體講解呈現播放器於 Linux 平台上的製作 - 內容協調與呈現技術"，交大碩士論文，2008

〔3〕 W3C，"Web Services Architecture"，[On-line].Available: http://www.w3.org/TR/2003/WD-ws-arch-20030514/#service_oriented_architecture，W3C Working Draft，14 May 2003

〔4〕 Newcomer, Eric; Lomow, Greg，"Understanding SOA with Web Services. "，Addison Wesley. ISBN 0-321-18086-0.，2005

〔5〕 Microsoft MSDN，"Model-View-ViewModel in Silverlight2 application"，[On-line].Available: http://msdn.microsoft.com/

〔6〕 Microsoft Silverlight，[On-line].Available: http://silverlight.net/

〔7〕 Adobe Flex，[On-line].Available: http://www.adobe.com/products/flex/

〔8〕 Sun JavaFx，[On-line].Available: http://java.sun.com/

〔9〕 Microsoft MSDN，"Composite Web Apps With Prism"，[On-line].Available: http://msdn.microsoft.com

〔10〕 財團法人資訊工業策進會，"多媒體產業專題研究報告"，財團法人資訊工業策進會資訊市場情報中心 (MIC)，2003

〔11〕 Christodoulou S. P., Styliaras G. D. and Papatheodorou T. S., "Evaluation of Hypermedia Application Development and Management Systems". 9th ACM conference on Hypertext and Hypermedia, ACM Press, Pittsburgh, 1998,pp. 1 - 10, ISBN:0-89791-972-6.

〔12〕 Duhl J.，"Rich Internet Applications"，IDC white papers，2003，[On-line]. Available: http://www.idc.com

〔13〕 W3C，"Hyper Text Markup Language (HTML) ", [On-line].Available: http://www.w3.org/MarkUp

〔14〕Jesse James Garrett，"Adaptive path"，[On-line].Available: http://www.adaptivepath.com/

〔15〕吳信輝，"網頁技術的新趨勢—RIA（Rich Internet Application）"，中央研究院計算機中心，[On-line]. Available: http://www.ascc.sinica.edu.tw/nl/93/2019/02.txt，2003

〔16〕Bozzon, A., Comai, S., Fraternali, P., Toffetti Carughi,G.，"Conceptual Modeling and Code Generation for Rich Internet Applications"，In Proc. of the 6th Int. Conf. on Web Engineering, pp. 353-360，ACM，2006

〔17〕Murugesan, S.，"Understanding Web 2.0. "，IT Professional Journal, vol.9, no.4, pp.34-41，2007

〔18〕Loosley, C.，"Rich Internet Applications: Design Measurement and Management Challenges"，Keynote Industry Brief，2006

〔19〕Driver M., Valdes R., Phifer G. "Rich Internet Applications are the next evolution of the Web"，Technical Report, Gartner，2005

〔20〕Lawton, G.，"New Ways to Build Rich Internet Applications"，Computer Volume 41, Issue 8, Aug. pp. 10-12，2008

〔21〕Melia, S.; Gomez, J.; Perez, S.; Diaz, O.，"A Model-Driven Development for GWT-Based Rich Internet Applications with OOH4RIA"，ICWE '08. Eighth Conf. on Web Engineering, pp. 13-23, IEEE CNF，2008

〔22〕W3C，"Document Object Model (DOM) "　，[On-line].Available: http://www.w3.org/DOM/

〔23〕InfoWorld，[On-line].Available: http://www.InfoWorld.com/

〔24〕余筱薇，"SCORM 編序規則的學習策略樣板產生器之設計製作與視覺化多媒體教材樣板之編輯套用系統"，交大碩士論文，2006

〔25〕AUGUSTO CELENTANO ,OMBRETTA GAGGI, "TEMPLATE-BASED GENERATION OF MULTIMEDIA PRESENTATIONS", International Journal of Software Engineering and Knowledge Engineering，2003

〔26〕江書瑩，"互動式多媒體的視覺化劇情編輯機制應用於多媒體試題樣板套用系統的實作"，交大碩士論文，2005

〔27〕Bertalanffy L. Von.，"General System Theory: Foundations

Development Applications"，New York: George Braziller，1968

〔28〕Flagle, C.D., William, H. & Ray, R.H.，"Operations research is system engineering"，Baltimore, MD: John Hopkins，1960

〔29〕Levie, W. H. & Dickie, K. E.，"The analysis and application of media"，In Second Handbook of Research on Teaching. Edited by R. M. W. Travers, Chicago, IL: Rand Mcnally，1973