國立臺北護理健康大學資訊管理研究所碩士論文

National Taipei University of Nursing and Health Sciences

Graduate Institute of Information Management

指導教授：黃衍文 博士

*Advisor: Ean-Wen Huang Ph.D.*

以雲端運算平台建置電子病歷檢視器系統之研究

The Study of Building Cloud Computing Platform for the System of electronic medical records

研究生：徐明睿 撰

*Name: Ming-Ray Hsu*

中華民國一○二年五月三十一日

May 31, 2013

以雲端運算平台建置電子病歷檢視器系統之研究

研究所組別：資訊管理研究所

指導教授：黃衍文 教授

研究生：徐明睿

時間：民國102年06月

# 論文摘要

關鍵字： 電子病歷、臨床文件架構、雲端運算、檢視器、超文本標記語言第五版

# Abstract

Keywords: Electronic Medical Record(EMR), Clinical Document Architecture(CDA), Cloud Computing, Viewer, Hyper-Text Markup Language 5(HTML5)

# 目錄

論文摘要 I

Abstract I

目　　錄 II

圖目錄 IV

表目錄 V

第壹章 緒論 1

第一節 研究背景 1

壹 國內電子病歷推行狀況 1

貳 雲端服務的普及度 2

參 行動裝置與平板電腦的崛起，Web 3.0的時代來臨 3

第二節 研究動機 3

第三節 研究目的 4

第貳章 文獻探討 5

第一節 電子病歷 5

第二節 資訊技術與服務 6

壹 基礎構想 6

貳 服務管理 7

第三節 XML 9

第四節 Web 3.0 10

壹 HTML5 11

貳 自適應網頁設計 12

第五節 雲端運算 13

第六節 多國語系應用程式 14

第七節 其他相關研究 14

第參章 研究方法 16

第一節 研究流程與步驟 16

壹 系統建置流程 17

第二節 系統需求分析與設計 19

壹 需求分析 19

貳 系統分析與設計 21

第三節 相關技術及開發工具 25

第肆章 研究結果與討論 26

第一節 研究結果 26

壹 樣板管理模組 26

貳 文件讀取及解析模組 27

參 資料內嵌處理模組 30

肆 樣板處理與頁面構成模組 33

伍 多國語系介面 35

第二節 討論 37

壹 樣板檔編輯 38

貳 與經過HIS轉換電子病歷比較 39

第三節 研究限制 41

第伍章 結論與建議 43

第一節 結論 43

第二節 未來努力方向 43

參考文獻 44

# 圖目錄

圖 1 研究流與步驟程圖 16

圖 2 雲端部署架構示意圖 19

圖 3單張代碼有預設樣板檔的查詢命令 26

圖 4單張代碼有預設樣板檔的回應結果 26

圖 5單張代碼沒有預設樣板檔的查詢命令 27

圖 6單張代碼沒有預設樣板檔的回應結果 27

圖 7 鼠標回應：未按附加按鍵 28

圖 8 鼠標回應：有按下附加按鍵 29

圖 9 提示訊息：不正確的格式 30

圖 10 資料內嵌處理膜組的系統畫面 31

圖 11 內嵌資料於頁面中 32

圖 12 內嵌CDA文件的Header與Body資料 33

圖 13 樣板處理與頁面構成模組的系統畫面 34

圖 14 包含附件時的畫面呈現範例 35

圖 15 中文語系介面 36

圖 16 英文語系介面 37

圖 17 語法使用示意圖 39

圖 18 系統載入所耗用的時間 40

圖 19 樣板檔套用的資料要求耗用時間 41

# 表目錄

表 1 電子病歷推動計劃補助款項及申請醫院家次統計 2

# 緒論

## 研究背景

國內衛生署推行的電子病歷的資料交換格式是由HL7制定的臨床文件架構所衍生的一種XML格式的文件，這文件格式現在國內已經有數百家醫院使用，各個醫院必須透過各自的EEC(E.M.R. Exchange Center, Electronic Medical Record Exchange Center) Gateway，經過健保局提供的VPN(Virtual Private Ntework)網路，與衛生署電子病歷交換平台(EEC)交換電子病歷資料。各醫院的醫師看診時，若要看其他醫院所產出的電子病歷，則必須在有病患的健保卡與醫師的醫事憑證卡，連線至EEC平台申請相關的電子病歷資料調閱，隨後等待資料回傳後，即可直接在HIS(Hospital Information System)中直接瀏覽資料。但目前交換電子病歷資料的對等醫院皆必須要有HIS系統能夠編、解碼交換的電子病歷文件，否則交換後的資料無法解讀給醫生瀏覽。至目前為止，國內所有的電子病歷都必須要各家醫療院所自行開發電子病歷相關的編、解碼程式，然後經過衛生署認證後，才能上線至EEC平台開始做電子病歷交換的流程。

### 國內電子病歷推行狀況

自民國九十八年開始，衛生署根據經驗上優先選取了常用的108份表單，並使用HL7 CDA格式制定了相關的資料欄位成為國內電子病歷資料交換的標準。並在隔年的『醫院實施電子病歷及互通補助計畫』中公告並開始使用四張醫院的病歷單張與三張中西牙醫門診單張交換使用，其餘百餘張表單在這幾年內皆尚未開始正式使用。而目前電子病歷推動計劃相關的補助以及申請醫院家次統計如表 1所示。

表 1 電子病歷推動計劃補助款項及申請醫院家次統計

|  |  |  |  |
| --- | --- | --- | --- |
| **項目/年度** | **99年度** | **100年度** | **101年度** |
| 補助預算 | 3.89億元 | 2.2億元 | 5.8千萬元 |
| 已申請之醫院家次 | 164家 | - | - |
| 已申請檢查之醫院家次 | - | 120家 | - |
| 已申請查驗之醫院家次 | - | 57家 | >170家 |
| 已申請之補助金額 | 3.27億元 | 2.2億元 | 5.8千萬元 |
| 補助款餘額 | 6.17千萬元 | 0元 | 0元 |

(參考於衛生署網站[1])

### 雲端服務的普及度

從目前國內衛生署正在積極推行「全民健康雲」，同時政府也在積極推行政府雲端應用加值服務中看出，不只政府推行雲端服務，甚至連國內外廠商也都陸續開始競爭。根據2012年11月，VMware委託Forrester Consulting進行的市場調查[[2](#_ENREF_2)]中，台灣是首次納入調查範圍內，其中共有473位來自跨國企業、地區或區域企業、以及公部門等組織的資深IT工作者作為受訪者。在該調查內發現台灣的企業雲端的普及率為44%，略高於亞太地區的平均值2%，且受訪者中有27%表示尚未使用雲端解決方案，但未來有打算使用雲端方案，而在計畫採用雲端方案的受訪者中，有56%表示會在未來的18個月內落實。

該調查內顯示，絕大多數的受訪者都能瞭解雲端方案對企業的益處，有76%受訪者表示雲端技術能有助於提升營運效能，79%認為能降低成本；另外有28%表示這對企業轉型帶來重要影響。另外，該調查也歸納出企業評估投入雲端前的重要因素，前三項包括：

1. 資料隱私、所在地，或「無法掌握」 (loss of control) (71%)
2. 成本 (69%)
3. 規範/管理/法規遵循議題 (67%)

### 行動裝置與平板電腦的崛起，Web 3.0的時代來臨

在過去，Web 2.0所使用的元件、概念，現在完全顛覆。自適應網頁設計(RWD, Responsive Web Design)的理念已經逐漸成為目前的主流，讓網頁可以自行適應於各種不同的裝置與設備上可以自動調整顯示的方式；若再以傳統概念設計系統，則無法完全發揮原先預期設計理念的成果。在過去，開發系統時僅需要一般電腦使用者能夠正常使用即可，但由於行動裝置與平板電腦的輕便，已經逐漸的讓使用者轉型。現在的使用者不再需要固定的電腦主機即可使用各式各樣的系統來完成工作，行動應用程式的崛起也創造不少商機，更多的醫療機構也逐漸陸續的投資使用平板電腦作為日常工作所需之設備。現在，大多數的網站設計都已經導入了RWD理念，這理念不僅能節省開發成本，專注於開發與增加應用程式更多方面的應用。

## 研究動機

在國際上，雲端服務與雲端醫療的爭相競爭之下，病歷資料的交換互通與即時處理顯示已經成為了未來的趨勢。尤其是國內的電子病歷交換之後，面臨到的是系統是否能夠即時顯示而不需要經過HIS的轉換就能夠呈現，或者能夠即時的轉換成不同的樣貌來顯示資料或者圖表。

在目前國內政府力推雲端醫療，以及國際上電子病歷的盛行之下，電子病歷的交換與使用是不可或缺的，而衛生署在為了推行電子病歷，並提出了推動補助款，加速國內電子病歷的普及率。

## 研究目的

為了可以增加政府推行電子病歷的成效，以現有技術考量，建立一套得以使各家醫療院所產出的電子病歷能夠互通檢視的雲端平台，以及能夠透過各家醫療院所客製化的表單至其他單位供檢視使用。若有使用遠距照護等系統，並且能產出電子病歷的話，則也可使用此一平台來協助病患自行檢視，迅速瞭解自己的遠距照護資料。

本系統提供給醫療院所、醫事人員及病患使用使用，並具有下列所述之目的：

1. 此一平台提供各家醫院能夠使用自行設計之客制化的樣板檔來顯示各種電子病歷單張
2. 提供上述各醫院設計的電子病歷單張樣板檔上傳至雲端平台的功能
3. 依據雲端平台的樣板檔，可在檢視病歷時自動套用原始單位所設計的樣板檔
4. 不需要經由HIS系統的轉換即可使用電子病歷內的資料呈現成報表或者表單
5. 本系統可以依據使用者要求而產生Green CDA規範下的XML文件以增加可用性
6. 符合資訊系統共享及永續經營理念，本系統所有功能皆可以客制化定義及設計不同的顯示方式

# 文獻探討

本章節依照研究目的，首先探討電子病歷的歷史與規格及國內交換的方式及相關驗證部分，並且依序探討資訊技術與服務為什麼會在本研究中使用，以及探討其他在本研究所開發之系統中所使用到的技術。

## 電子病歷

電子病歷(Electronic Medical Record, EMR)是一種由電子化方式擷取、傳送、儲存、取回、連結與處理的多媒體資料，電子病歷主要的用途為協助醫療或其相關服務。其資料內容包括病患的個人資料、問題、SOAP(Subjective, 主觀陳訴, Objective, 客觀陳訴, Assessment, 評估, Plan, 計畫)、病程紀錄、護理計畫、護理記錄、生命徵象記錄、藥物使用記錄、過去病史、家族病史、預防接種記錄，與其相關的醫療實驗室的檢驗資料與報告，和相關的檢查資料與報告，凡是相關病情必要之資訊，如相對於疫情的旅遊史，則亦可成為電子病歷的一環而成為電子病歷的資料內容[[3](#_ENREF_3)]。

目前在國內的電子病歷則是使用HL7 CDA v3的規格所制定的，在資料交換時，必須經由健保局提供的VPN網路，透過電子病歷交換中心閘道器(EEC Gateway, Electronic Medical Record Exchange Center Gateway)交換資料。EEC Gateway是一個必須在各個醫院設立的閘道器主機，目的在於EEC平台可以透過此一閘道器作為一個中繼點，使用健保局提供的VPN當作管道，平時醫院可以將需要上傳的資料先儲存於該主機內，等待EEC平台至該主機內定期的抓取上傳的資料。若是作為資料調閱時，可以透過EEC平台查詢資料索引，並且通知存有資料的EEC Gateway將完整資料回傳至EEC提供調閱，並隨後轉送至需要該調閱資料的單位的EEC Gateway內，HIS取得該資料後才能將資料轉換並呈現給醫事人員瀏覽。

## 資訊技術與服務

ITIL(Information Technology Infrastructure Library)針對的是使用者的感受，強調終端與終端的服務，與本研究開發之系統的關聯在於本研究強調於使用者可以自行設計樣本檔並且提供給其他使用者使用。

### 基礎構想

資訊科技基礎構想(ITIL)源起於1980年代的英國政府商務部(OGC, Office of Government Commerce)的前身組織 -- 顧客信用貿易協會(CCTA, Consumer Credit Trade Association)所主導進行的「如何提升政府資訊科技與管理」研究專案。又被稱為資訊服務管理的最佳實務(Best Practice)。資訊服務管理的ISO標準(ISO/IEC 20000)主要參考架構所發展制定。

ITIL之發展有三個階段：

* 1. 針對IT技術提供者(功能基礎)
  2. 針對服務提供者(處理流程為導向)
  3. 針對策略夥伴(生命週期管理)

ITIL並不是個正式的標準，而是目前普遍執行於「事實」上的準則。而整個ITIL的核心重點在於人與處理流程的爭議必須第一個被提到，其次就是要有「質」感的客戶服務，也就是必須無時無刻的去迎合客戶的需求和期望。

ITIL第二階段以前，主要分成兩個部份：服務支援 (Service Support) 與服務遞送 (Service Delivery)。ITIL 最新版本(Version 3)則將安全管理(Security Management)納入考量，所以ITIL包含七個模組：

* 服務管理(Service Management)
* 服務支援(Service Support)
* 服務遞送(Service Delivery)
* ICT基礎架構管理(ICT Infrastructure Management)
* 規劃到建置服務管理(Planning to Implement Service Management)
* 應用管理(Application Management)
* 企業遠景(The Business Perspective)
* 安全管理(Security Management)

因為雲端醫療的電子病歷的專業領域與一般IT有所不同，其服務的項目是各個醫療院所所使用的醫師，而不是一般使用者；所以服務內容是透過各個不同的醫療單位所提供的資訊，並不是固定內容的資訊系統。因此本研究僅針對ITIL的服務管理做探討，其他模組則不在本研究的範圍。

### 服務管理

本研究僅使用到前面所述之ITIL的第一個模組服務管理(ITSM, IT Service Management)內細分的服務支援(Service Support)及服務遞送(Service Delivery)的服務傳遞部分，其詳細說明如下。

服務遞送有以下五種管理：容量管理(Capacity Management)、服務水準管理(Service Level Management)、可用性管理(Availability Management)、服務永續性管理(Service Continuity Management)、服務的財務管理(Financial Management for Service)。本研究中使用到可用性管理與服務永續性管理兩部分。

可用性管理(Availability Management)有兩個處理流程階段：

* 透過適當資源、方法和技術的部署，從而確保與顧客約定之服務應用(SA)可用性目標得以實現的流程。
* 主要涉及維護活動的最佳化規劃，以及設計相關措施，將事件發生的次數降至最低。

階段一：規劃(Planning)：由於此部分服務應用軟體(內製、委外、夥伴)及IT的規劃，所以服務經理必須參加三種不同特性的規劃，不僅反映客服需求，同時也要求各單位(包含內部、委外、夥伴)設計「可用性」及「復原」的監督績效評核準則，才有辦法制定客服管理方法。

* 決定可用性需求
* 可用性的設計
* 復原的設計

階段二：監督(Monitoring)

* 監督和趨勢分析
* 檢閱服務(服務應用/IT)及設備的可用性

服務永續管理(Service Continuity Management)：

* 找出所需要的和協議的商業營運最低層級，以及系統、設備和服務需求所涵蓋的需求定義
* 檢查對這些需求的風險及威脅
* 開發新系統的風險降低或是減輕計劃

持續性管理之目標：

* 持續性管理之主題在於管理"風險"
  + 確保組織的服務應用軟體基礎架構，在發生不可能發生的事件或非預期的事件時可以繼續提供服務
  + 即確保當發生正常可用性解決方案失敗的事件時，仍能夠提供價值給客戶
* 透過商業處理程序的分析，對於服務應用軟體基礎架構弱點進行風險控管
* 持續性管理應結合於企業整體永續經營計畫中

## XML

可延伸標記式語言(XML, eXtensible Markup Language)是一種標記式語言。標記指電腦所能理解的訊息符號，使用此種標記，電腦與電腦之間可以互相處理包含各種訊息的資料。XML是從標準通用標記式語言(SGML, Standard Generalized Markup Language)中簡化修改出來的。它主要用到的有可延伸標記式語言、可延伸樣式語言(XSL, eXtensible Stylesheet Language)、可擴展商業報告語言(XBRL, eXtensible Business Reporting Language)和XML路徑語言(XPath, XML Path Language)等[[4](#_ENREF_4)]。

XML是設計用來傳輸及儲存資料資訊，不是用來顯示或呈現資料，相對的HTML(Hyper-Text Markup Language)則是用來呈現資料，所以XML的主要用途如下[[4](#_ENREF_4)]：

1. 豐富文件(Rich Documents)自定義檔案的描述並使其具有更豐富的內容
   * 屬於以檔案為主的應用
   * 標記是用來定義資料應該如何呈現
2. 中繼資料(Metadata)描述其它檔案或網路資訊
   * 屬於以資料為主的應用
   * 標記是用來說明資料的意義
3. 配置檔(Configuration Files)描述軟體設定的參數
   * 屬於以屬性為主的應用
   * 標記是用來配置資料或應用程式的設定

## Web 3.0

Web 3.0是一個概念，他的定義在2006年11月的Technet峰會上首次被提出。Netflix創始人Reed Hastings闡述了定義Web術語的簡單公式：

「Web 1.0是撥號上網，50K平均頻寬，Web 2.0是1M平均頻寬，那Web 3.0就該是10M頻寬，全視頻的網路，這才感覺像Web 3.0。」

而隔年的8月7日，Google首席執行官埃里克•施密特出席首爾數字論壇

「Web 2.0只是一個行銷術語，而你剛才正好發明了Web 3.0這個行銷術語。」

隨後他談及了自己的具體看法

「建應用程序的方法將不同。到目前為止Web 2.0一詞的出現主要是回應某種叫做「AJAX」的概念，而對Web 3.0，我的預測將是拼湊在一起的應用程序，帶有一些主要特徵：

* 應用程式相對較小
* 資料處於Cloud中
* 應用程式可以在任何設備上運行（電腦或者行動裝置）
* 應用程式的速度非常快並能進行客製化
* 此外應用程式像病毒一樣地擴散（社交網路，電子郵件等）

」，而要如何達到Web 3.0的概念呢？其實我們只要替使用者提供更加豐富的相關使用者經驗，並且讓每個使用者可以有獨特的網路使用模式，如此一來，這就達到了Web 3.0的概念。

### HTML5

超文本標記語言第五版(HTML5, Hyper-Text Markup Language 5)草案的前身名為Web Applications 1.0，是在2004年由WHATWG提出，再於2007年獲W3C接納，並成立了新的HTML工作團隊。它是HTML下一個主要的版本，現在仍處於發展階段。目標是要取代1999年所制定的HTML 4.01和XHTML 1.0標準，以期望能在網際網路應用迅速發展的時候，使網路標準符合現代的網路需求。廣義的說HTML5，實際指的就是包括HTML、CSS和JavaScript在內的一套技術集合。它希望能夠減少瀏覽器需要外掛程式的豐富性網路應用服務(RIA, plug-in-based Rich Internet Application)，如Adobe Flash、Microsoft Silverlight，與Oracle JavaFX的需求，並且能夠提供更多有效增強網路應用的標準[[5](#_ENREF_5)]。以下是本研究中所使用到的HTML5的功能。

* 離線儲存資料庫(離線網路應用程式)[[6](#_ENREF_6)]：可以讓網頁不需要網路的狀態下儲存資料或存取資料，等有網路的時候再依照設計的模型決定是否要與伺服器同步資料
* 編輯：讓網頁直接編輯，不再需要透過編輯器或所見即得編輯器
* 拖放：使檔案的選擇不需要經由系統的對話視窗選擇即可使用拖放的方式將檔案放置在網頁中讓應用程式讀取資料
* 瀏覽歷史管理
* MIME和協議處理程式時表頭登記
* 檔案API：處理檔案上傳和操縱檔案[[7](#_ENREF_7)]
* 目錄和檔案系統：這個API是為了滿足客戶端在沒有好的資料庫支援情況下的檔案儲存要求[[8](#_ENREF_8)]
* 檔案寫入：從網路應用程式向檔案裡寫資料內容[[9](#_ENREF_9)]

### 自適應網頁設計

自適應網頁設計(RWD, Responsive Web Design)是一種網頁設計的概念，這種設計的方式可以使網站可以適應到多種瀏覽的裝置上(從電腦到行動電話或者行動裝置)閱讀，同時可以減少縮放和捲動。

採用RWD設計的網站會使用到CSS3的媒體查詢，即是對CSS的 @media 規則的擴充，以適應不同大小的裝置。但是這在比較就版本的瀏覽器就無法使用。

## 雲端運算

雲端運算是一種基於網際網路的運算方式，這種方式是繼1980年代的大型電腦到主從式架構的大轉變之後的再次巨變。雲端運算通常可以認為包括以下三個層次的服務：基礎設施即服務(IaaS, Infrastructure as a service)，即是使用「基礎運算資源」，如處理能力、儲存空間、網路元件或中介軟體，使用者能掌握作業系統、儲存空間及已部署的應用程式和網路元件(如防火牆、負載平衡器等)，但並不掌握雲端基礎架構。例如: Amazon AWS；平台即服務(PaaS, Platform as a service)，即是使用主機操作應用程式，使用者有掌控運作應用程式的環境(也擁有主機部分掌控權)，但並不掌控作業系統、硬體或運作的網路基礎架構。使用的平台通常是應用程式基礎架構。例如: Google App Engine；軟體即服務(SaaS, Software as a service)，即是使用應用程式，但並不擁有作業系統、硬體或運作的網路基礎架構，是一種服務的觀念基礎，軟體服務的供應商以租賃的方式提供客戶服務，而非購買，比較常見的模式是提供帳號密碼來操作使用。例如: Microsoft CRM。這些雲端運算服務通常透過瀏覽器存取，讓軟體和資料可在資料中心儲存和使用。

根據美國國家標準和技術研究院的定義[[10](#_ENREF_10)]，雲端運算服務應該具備以下幾條特徵：

* 需要時即可使用的自助服務：不需要人的操作即可以自動化的將需要的資源啟用
* 寬頻網路存取：使用任何的網路或裝置存取
* 資源池：可以將資源集合在一起，並且可以提供多人共享使用
* 可快速重新佈署：當有需求的時候可以自動化的重製來擴展服務寬度
* 可被監控與量測的服務：讓系統可以很清楚的知道使用的狀況，並且可以控制和優化使用的資源

本研究中，主要為資源池的使用方式來提供使用者服務，而且可以快速地重新部署系統至各地方醫療機構使用。

## 多國語系應用程式

在國際上，很多的東西與物件的名稱都有不同的稱呼方式，而為了讓使用者可以更容易瞭解物品的名稱，因此必須要標註不同語言的名稱在物品上，而多國語系的應用程式也是如此。在一個應用程式的介面上，以不同的文字來呈現相同的功能，由於是針對不同語系的人所設計的文字，所以即使使用者是來自不同語系的國家也能正常的使用這個應用程式。若應用程式要國際化發展的話，則勢必要有不同的文字翻譯，如此這個應用程式將能國際化發展。

## 其他相關研究

目前與電子病歷相關的系統有Microsoft HealthVault與Google Health，但是目前Google已經於2012年的1月1日將Google Health終止服務了，因此這裡只探討Microsoft HealthVault的功能。

Microsoft HealthVault是一個個人健康履歷的網站，目前僅提供美國與英國地區的服務，其中包含個人健康歷、生理徵象、檢驗檢查、醫學影像等資料，皆能在此一服務上存取，並且可以匯出成電子病歷資料提供醫師檢視使用。

# 研究方法

本研究為實驗設計，但並不是一般商業軟體開發，故本系統建置流程不適合於一般商業軟體系統開發建置流程，所以較為簡化。

## 研究流程與步驟



圖 1 研究流與步驟程圖

本研究會設計一個電子病歷的檢視平台，並且提供各醫療院所及民眾使用。依據研究動機來決定並確認其研究目的之範圍，並進行國內、外的文獻、規範、書籍做探討，並且彙整後先瞭解技術的可行性，以及能否符合預期設計和目標，最後實作系統，該系統主要提供醫療院所做為一個電子病歷資料交換後的檢視平台，並且可以在線上開發各種單張所需的檢視模板。最後檢查實際是否能夠達到預期目標。如圖 1所示本研究的研究流程與步驟：

### 系統建置流程

在整個系統建置的過程裡，首先做文件分析對CDA等相關的文件分析，然後做系統的規劃與設計，最後開始系統開發與部署至雲端平台上進行系統測試。

#### 文件分析

首先，先研究CDA文件架構，研究CDA文件內的架構與模型、參考方式、資料屬性及詞彙等，研究文件的階層及儲存位置還有形態等內容。第二，分析電子病歷文件單張，為了區分出每個單張的不同，以及辨識不同單張所需的樣板檔，必須先瞭解電子病歷文件中所記錄的代碼為何種意義，以便瞭解哪個代碼的特性為何，並且針對該代碼來作為區分用途。第三，研究XPath使用特性，在電子病歷文件中接使用XML來儲存內容，若要萃取出資料則必須要有選擇，而在XML文件中要萃取出需要的資料則是要先瞭解XPath內的特性以及使用方式。

#### 系統設計

首先的第一步驟為系統規劃。系統規劃是根據整個CDA的使用目的來設計本系統的相關功能，而原本電子病歷的使用目的是希望CDA能在各家醫院的HIS系統編、解碼使用，但是因為現有的醫院有很多不是沒有開發解碼電子病歷的功能，那更不用說編碼了。所以說，系統的需求與設計，首先注重於各醫院，不論是否已經或尚未開發電子病歷的編、解碼程式，本系統將協助醫事單位表單開發設計人員完成電子病歷的編、解碼作業。同時，也提供一般醫事人員與民眾解讀電子病歷文件的內容。

根據系統規劃使用者的需求，首先完成系統架構圖，然後開始系統開發，而開發過程中將會依序分別為：建置系統框架、檔案解析介面、樣板檔管理等三個系統模組。

* 1. 建置系統框架

建置系統框架，提供使用者直覺的檢視資料。其中必須提供檢視工具，並且能在檢視器的畫面中呈現出如同真實紙本資料的畫面，可以直接顯示圖表，或者可程式化的互動介面。使用者可以在畫面使用拖拉的方式將檔案置入檔案解析介面中開始解析檔案。

* 1. 檔案解析介面

讓使用者可以將檔案匯入至頁面中解析資料，並且可以結合樣板檔互相搭配呈現畫面。其中會依據電子病歷檔案內的屬性來判斷該檔案是為哪一單張的資料結構，隨之自動的配合相對應的單張樣板檔來呈現該單張的顯示介面以及互動介面。

* 1. 樣板檔管理

管理已經設計好的樣板檔，並且可以編輯或者引用其他設計者所設計好的介面，另外提供版本歷程的記錄，以避免版本變更後的不預期問題發生。

#### 雲端部署

因為國內將要推行雲端醫療平台，故本研究將會以雲端平台的方式，將系統部署到雲端上，且採用開放原始碼的Cloud Foundry平台架構進行部署。但由於此種架構下的雲端平台的硬體需求必須達到最少128GB的記憶體[[11](#_ENREF_11)]，因此本研究直接採用其他使用Cloud Foundry架構的PaaS平台來部署本系統，所以本階段需要利用部署工具將本系統部署至平台上即可使用(如圖 2所示)。



圖 2 雲端部署架構示意圖

## 系統需求分析與設計

根據系統建置流程中所提及的系統設計規劃中，將三個系統模組需要的功能及流程依序分析及設計。

### 需求分析

依照使用者分析其代表的角色將會有如下列所示：

* 醫事單位表單開發設計人員：負責文件樣板的設計以及開發，讓醫事人員以及一般民眾可以很容易的使用本系統瀏覽由醫院所輸出的電子病歷資料，以及和其他系統的整合應用
* 醫事人員：使用本系統檢視電子病歷資料的時候可以經由本系統導入資料至醫事單位所開發整合的系統中使用
* 一般民眾：僅只有電子病歷的檢視功能，並無其他功能可以使用

根據系統建置步驟來分析每個步驟中需要的功能以及處理的目標為下：

* 1. 系統框架

系統的功能必須要有以下功能：

* + 必須可以將畫面分割顯示於不同的分頁之中
  + 能夠顯示頁碼，以及頁首與頁尾
  + 必須能夠提供檔案的讀取及寫出的能力
  1. 檔案解析介面

檔案解析介面的功能必須要有以下功能：

* + 必須檢查檔案是否為CDA文件
  + 將CDA的Header與Body讀取出來，並加以區分
  + 能夠將樣板檔選取的XPath資料以元件的方式呈現
  + 能夠列舉XPath所選取的資料
  + 能夠有資料對應表來制作代碼區域對應的呈現
  1. 樣板檔管理

樣板檔管理的功能必須要有以下功能：

* + 必須自動的在找不到對應樣板檔的時候退回至該單張的預設樣板檔
  + 必須自動的在找不到對應的預設樣板檔時，使用系統範例樣板檔
  + 提供樣板檔上傳介面，讓設計者可以提供樣板檔給對應的電子病歷單張

### 系統分析與設計

依照需求中所提及的系統建置步驟可以歸納出本研究之系統中，主要有五個系統模組來形成電子病歷檢視器，分別為(1)文件讀取及解析模組;(2)資料內嵌處理模組;(3)樣板處理與頁面構成模組;(4)樣板管理模組;(5)多國語系介面，其中以開發界面最為重要，因為包含了形成檢視的檢視畫面以外，還必須有設計的功能，並且還要結合樣板檔資料庫，以便開發中引用其他樣板做進一步開發。以下將對這幾部分加以探討：

#### 樣板管理模組

此模組提供系統查詢與上傳系統或者使用者提供的樣板檔，以便使用者在使用系統時可以提供顯示對應的單張樣板頁面。

在其他模組要求樣板檔的時候，會使用md5 hash演算法作為資料識別ID，並且使用以下規則傳回適當的樣板檔以供使用：

1. 以單張代碼及醫療機構代碼查詢樣板檔
2. 若上述規則無樣板檔，則以單張代碼返回其使用的公用樣板檔
3. 若上述規則無樣板檔，則直接返回通用樣板檔

而樣板檔中必須定義各樣版的名稱於根元素內，定義的時候必須以固定的格式定義識別名稱，其規則為下：

1. 若為醫療機構客制的單張，則以此種方式命名：cda\_\_\_單張代碼\_\_\_醫療機構代碼
2. 若為單張公用樣板檔，則以此種方式命名：cda\_\_\_單張代碼
3. 若為通用樣板檔，則以此種方式命名：cdaDemo

最後，樣板檔的設計人員將樣板檔設計完成之後，透過此模組可以將樣板檔上傳至本系統中，系統將會依據樣板檔內宣告的命名來進行存檔作業，並且進入版本控管。若原始檔案已經有相同的檔名存在的時候，將會在檔名之後加上版本流水編號以利區分，日後也可取得過去各種版本的樣板檔，提供其他使用者使用。

#### 文件讀取及解析模組

電子病歷文件需要透過使用者驅動瀏覽器讀取資料，本系統會將檔案使用Drag & Drop的方式讓使用者輸入檔案，過程中必須要使用者手動的將要讀取的一個或多個檔案使用滑鼠拖曳至瀏覽器的畫面中，接收到OnDrag Event之後系統就會將一個或多個檔案依序讀取出來，並且檢查是否為系統可接受的檔案格式，並且在通過檢查之後將此一動作交由下一個系統模組處理。

在資料讀取的時候，為了要確定使用者輸入的檔案確實為符合臨床文件架構的電子病歷資料，故本模組會有以下規則來檢查是否為正確的文件格式：

1. 確認是否有XML的文件宣告
2. 檢查是否有ClinicalDocument的Root Element，並且有Element結尾
3. 檢查是否有xmlns宣告為urn:hl7-org:v3的schema
4. 其使用的RegExp檢查式為：/<\?xml[^?>]+\?>\s\*(<\?xml-stylesheet[^\?]+\?>)?\s\*(<\?[^xml-][^\?]+\?>\s\*){0,}<([\w]\*:)?ClinicalDocument[^>]+>([\s\w\W]\*)<\/([\w]\*:)?ClinicalDocument>([\s\w\W]\*)/
5. 但本系統不會根據宣告的schema去驗證XML是否符合schema的宣告

#### 資料內嵌處理模組

此一系統模組中，承接上一模組的事件，隨後將資料內嵌至系統的畫面中，等待處理模組處理完成之後再交由下一模組處理。

經由上一模組判斷為正確的檔案格式之後，就會進入本模組開始將檔案區分為兩個部分：CDA Header與CDA Body。其區分的方式為下列XPath表示：

* 取得CDA Header：\*:ClinicalDocument/(\* except self::\*/\*:component)
* 取得CDA Body：\*:ClinicalDocument/\*:component

在區分之後，將會分別以cdaHeader與cdaBody的元素物件包裝起來，放入cda2g這個元素物件中，並且使用Web Components的API處理在這個元素內的資料與讀取。在資料處理的時候，會根據樣板檔的描述，去決定選擇header的資料或者是body的資料，並且使用XPath的方式提供資料的選取。在選取的過程中，可以指定輸出的方式為具有格式化的資料或者純資料選取的方式輸出，若選取的資料不存在，則選取的時候不會有任何的資料輸出。

#### 樣板處理與頁面構成模組

資料處理後，將結果依照樣板檔的描述，輸出成最終的頁面結構，以及依照使用者在文件讀取階段所下的指令來決定是否要輸出樣板檔所描述相對應的輸出目標結構來產生XML文件輸出。

本模組會讀取其樣板中的單張代碼，以及醫療機構代碼，並且依照這兩個代碼所組成的資料形成唯一值，根據這個唯一值對本系統要求電子病歷樣板檔回傳至前端系統，並且使用Web Components結合這些樣板檔使用於文件呈現、資料處理的模組處理，畫面的顯示與輸出處理主要會以一個單張資料為一頁，若該單張的資料過多，則會在超過範圍的分頁處截斷至下一頁處理，而被截斷的頁面屬性將會延伸到截斷後的新頁面上。

由於電子病歷資料可以內嵌其他附件資料(例如：圖片、文件)，所以這些資料都會儲存於電子病歷內的observationMedia段，並且會以base64的方式儲存資料的內容。而本模組會使用HTML5的File API來產生檔案的轉換及下載輸出，在檔案轉換輸出的時候會有以下流程：

1. Base64使用瀏覽器atob function轉換為binary string資料
2. 將binary string資料轉換回正確的binary資料
3. 使用Blob class將binary資料轉換成瀏覽器的blob資料
4. 建立blob URL
5. 輸出為圖片帶超連結或者文件資訊帶超連結

在檔案轉換輸出後，會根據檔案的形態來決定是否要內嵌於畫面當中。如果是圖片的格式，則會限制最大不超過頁面大小，並且內嵌於畫面上，也可以直接將圖形儲存於電腦上，檔名則是根據電子病歷文件內設定的檔案名稱來自動存檔。若不是圖片格式，則本系統會於畫面中顯示其檔名，以及檔案的大小於畫面中，並且具有超聯結可供使用者從系統內部快取中下載該檔案並儲存於使用者電腦上的任意位置。

#### 多國語系介面

系統會依照使用者的瀏覽器設定自動的設定到對應的介面與語言顯示，讓使用者可以直接進到系統就是使用者所熟悉的語言環境與介面。

## 相關技術及開發工具

本研究準備使用的開發環境、開發工具、開發程式語言如下：

* 開發環境： Macintosh OS X Mountain Lion 10.8.3、Microsoft Window 8 Pro
* 開發工具： Sublime Text 2、SourceTree、Google Chrome、Mou
* 開發語言： HTML 5、JavaScript、jQuery、jQuery UI、CSS、php、mySQL、Markdown

# 研究結果與討論

## 研究結果

以系統中各個模組處理的過程與結果來分別敘述內容如下。

### 樣板管理模組

模組對應的時候會經由md5編碼產生一段由單張代碼與機構代碼所組成的路徑，當查詢樣板檔時，如果沒有對應的樣板檔，則有下面兩種狀況處理：

1. 單張代碼有預設的樣板檔
2. 完全無法對應

這兩種狀況我們分別以curl公用程式執行指令查詢伺服器回應會有以下結果：

1. 單張代碼有預設的樣板檔

Macintosh HD:Users:chaochan:Dropbox:Institute:CDA2graph:documents:image:121_curl_bash.png

圖 3單張代碼有預設樣板檔的查詢命令

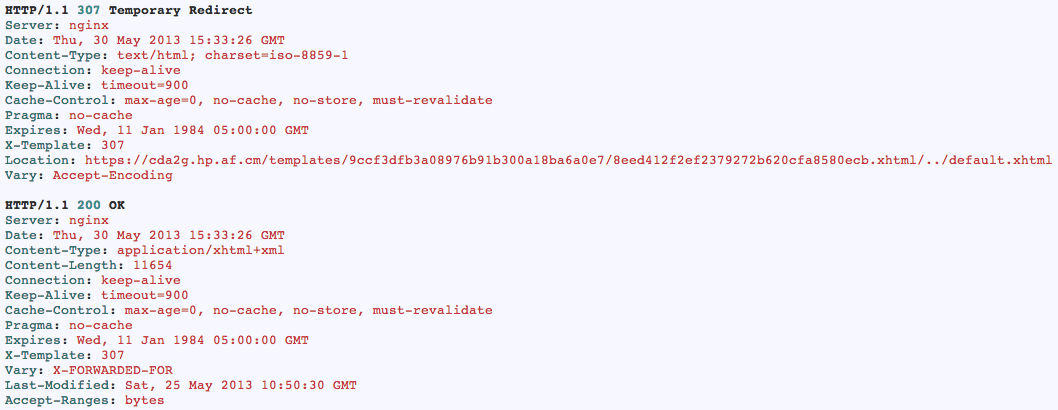


圖 4單張代碼有預設樣板檔的回應結果

1. 完全無法對應

Macintosh HD:Users:chaochan:Dropbox:Institute:CDA2graph:documents:image:115_curl_bash.png

圖 5單張代碼沒有預設樣板檔的查詢命令

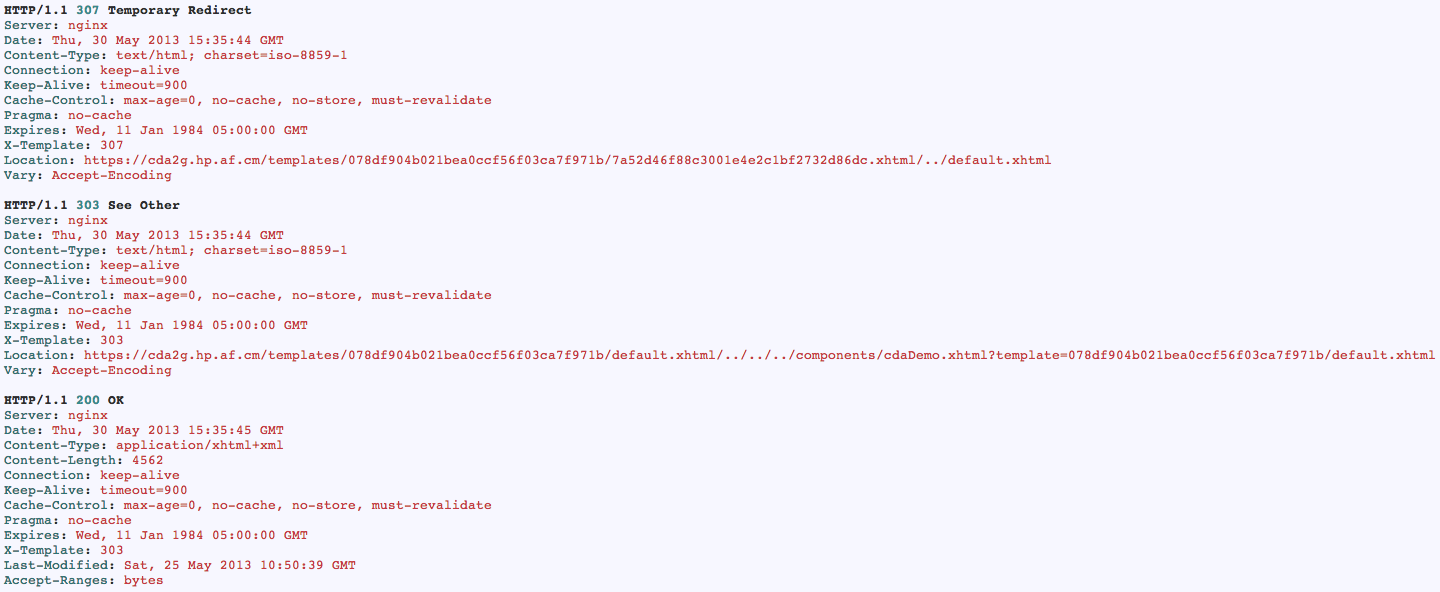


圖 6單張代碼沒有預設樣板檔的回應結果

### 文件讀取及解析模組

在觸發讀取文件的拖放事件的時候，系統會依據使用者附加按鍵會有下面兩種不同的回應鼠標(參照圖 7與圖 8)，然後當使用者放入了不正確的文件的時候則會出現如圖 9所述的錯誤訊息。



圖 7 鼠標回應：未按附加按鍵



圖 8 鼠標回應：有按下附加按鍵



圖 9 提示訊息：不正確的格式

### 資料內嵌處理模組

在資料讀取完成之後，會將資料內嵌至頁面中等待處理(進入此模組時，系統畫面會如圖 10所示)，而其內嵌的結構與CDA格式相同，但僅只有將CDA Header與CDA Body分開至兩個物件內包裝，其中會包含使用Web Components的技術，而Web Components的描述結構皆在#document-fragment的物件內。(如圖 11與圖 12所示)



圖 10 資料內嵌處理膜組的系統畫面

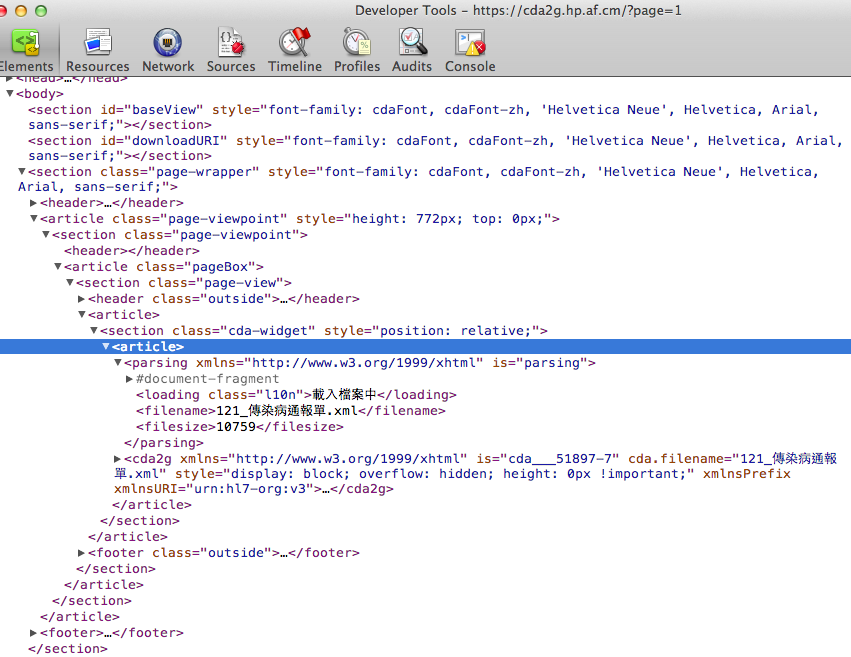


圖 11 內嵌資料於頁面中

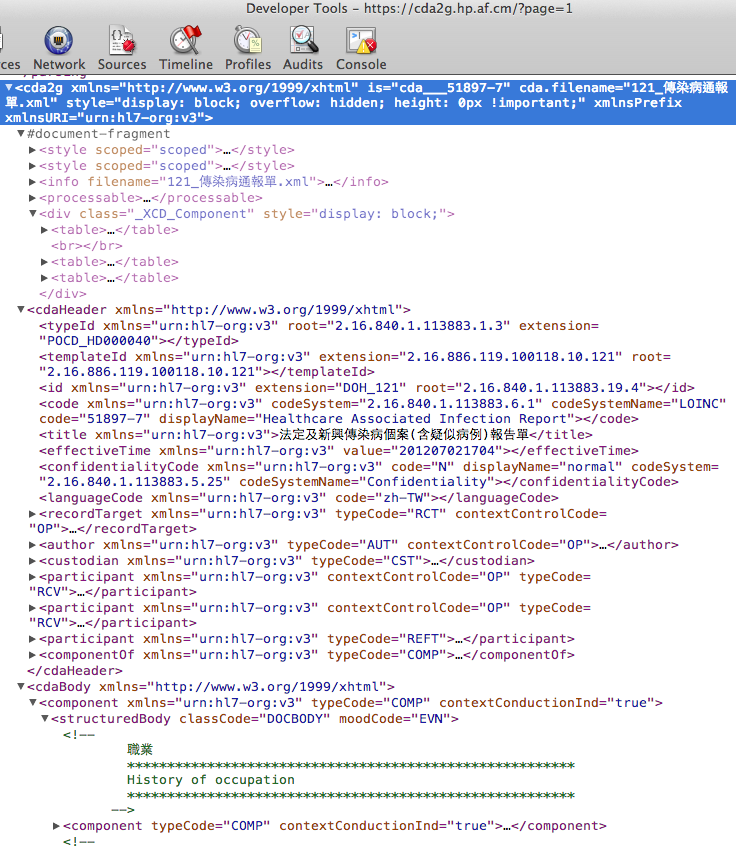


圖 12 內嵌CDA文件的Header與Body資料

### 樣板處理與頁面構成模組

本模組將內嵌的資料取出(進入此模組時，系統畫面會如圖 13所示)，並且將Web Components的結果取出並取代原本內嵌的結構，最終變成如圖 14所示的文件結構。



圖 13 樣板處理與頁面構成模組的系統畫面



圖 14 包含附件時的畫面呈現範例

### 多國語系介面

依照使用者瀏覽器的語言不同，本系統會自動的調整顯示的語言，以檔案格式錯誤微粒，中文與英文的界面將會如圖 15與圖 16所示。



圖 15 中文語系介面



圖 16 英文語系介面

## 討論

本研究為了可以有更多的擴充性與靈活度，採用Web Application的方式寫作，並且結合MVC(Modal View Controller)的設計方式，以及結合RWD理念，所以本系統中隨時可以根據樣板檔內的設計不同而提供不同的擴充功能，因此僅需要修改樣板檔，即可馬上對應到任何有相關的電子病歷單張的檢視畫面。

### 樣板檔編輯

本研究由於是可行性研究，故目前如果要對樣板檔編輯，則必須要使用支援XHTML的編輯器編輯，編輯時僅需要將檔案中命名的屬性修改後即可變成專用樣板檔。在製作樣板檔客制化時，僅需要有HTML的設計編寫能力即可客制化樣板檔。

目前樣板檔中所使用的功能元素(Functional Element)有如圖 17作為說明：

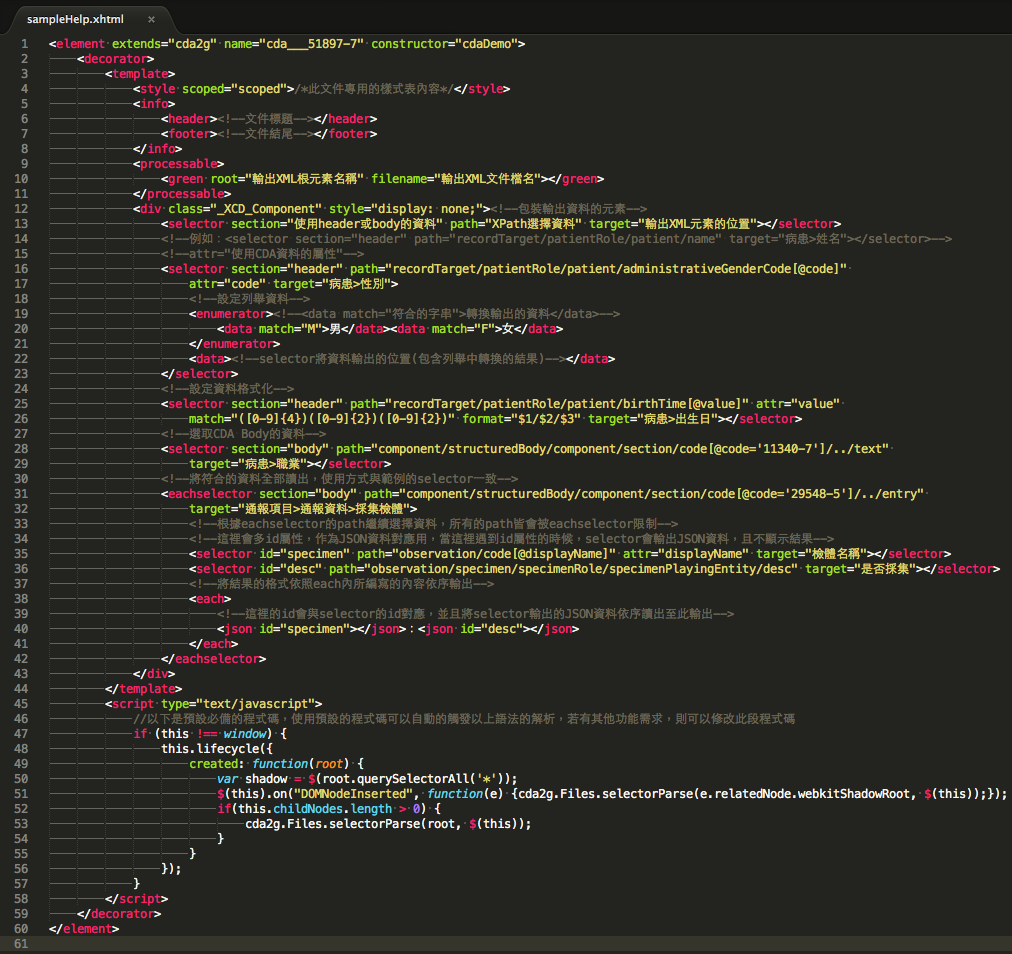


圖 17 語法使用示意圖

### 與經過HIS轉換電子病歷比較

本研究之系統提供簡單的功能，令使用者僅需要簡單的動作即可套用樣板檔產生結果，HIS轉換的電子病歷結果需要靠系統排程的時間逕行轉換，而本系統在有對應的樣板檔的時候，系統載入與套用樣板檔僅需要約不到30秒的時間即可完成一個單張的樣板的套用並顯示結果(如圖 18與圖 19所示)。



圖 18 系統載入所耗用的時間



圖 19 樣板檔套用的資料要求耗用時間

## 研究限制

本研究的範圍，將會限制在系統的可行性研究上，會以主流的網頁應用程式設計的方式導入到電子病歷交換的層面，觀察能否應用於醫療領域；但下列各項將不列入本研究範圍：

* 電電子病歷呈現用的樣板檔編輯器
* 樣板擋存回電子病歷中增加可攜性之研究
* 電子病歷的制作與交換
* 電子病歷的格式與數位簽章之驗證及檢查
* 資料安全與防止漏洞隱碼之研究
* 雲端平台相關建制之研究
* 國內其他非正式交換用的電子病歷規範之研究

# 結論與建議

## 結論

本系統目前已經可以呈現現行已經公告出來進行交換用的四個門診單張，以及疾病通報單張。同時，本系統操作簡單，使用者僅需要將檔案拉入瀏覽器的畫面中即可開始解析電子病歷的內容，並且可以匯出成XML文件。本研究之系統目前同時也能夠達到永續經營之目的，隨時可以提供資料維護的作業，並且利用雲端平台的服務能力，提供不間斷的服務。

本研究完成的系統可以將病歷資料取出，並及時對應出適當的單張範本，自動結合並且解析，隨之呈現於瀏覽器畫面中。使用者在使用的時候並不需要瞭解電子病歷的規範與格式，僅需要做最簡單的檔案拖拉即可呈現電子病歷的內容，如此可以讓各醫療院所節省電子病歷轉換解碼的程序及減少開發時所遇到的問題。

## 未來努力方向

針對本研究系統之建議如下，可提供未來研究者參考：

1. 本系統僅只有實作單張樣板之對應，並無結合代碼轉換之功能。
2. 未來可與多家醫院結合使用，使之可以不用建置電子病歷解碼程式。

# 參考文獻

[1] 行政院衛生署電子病歷推動專區. *歷年補助案*. Available: <http://emr.doh.gov.tw/allowance.aspx>

[2] VMware. *VMware 雲端成熟度指標：台灣雲端運算普及率 44%*. Available: https://<http://www.vmware.com/tw/company/news/releases/VMware-cloudindex2012tw-112012.html>

[3] Wikipedia. *電子病歷*. Available: <http://zh.wikipedia.org/zh-tw/%E7%94%B5%E5%AD%90%E7%97%85%E5%8E%86>

[4] Wikipedia. *XML*. Available: <http://zh.wikipedia.org/wiki/XML>

[5] Wikipedia. *HTML5*. Available: <http://zh.wikipedia.org/wiki/HTML5>

[6] I. Hickson, "Web Storage," in *W3C Candidate Recommendation*, ed, 2011.

[7] A. Ranganathan and J. Sicking, "File API," in *W3C Working Draft*, ed, 2012.

[8] E. Uhrhane, "File API: Directories and System," in *W3C Working Draft*, ed, 2012.

[9] E. Uhrhane, "File API: Writer," in *W3C Working Draft*, ed, 2012.

[10] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," in *National Institute of Standards and Technology Special Publication* ed, 2011, pp. 800-145.

[11] C. Foundry. *Deploying Cloud Foundry on vSphere - Hardware Requirement*. Available: <http://cloudfoundry.github.com/docs/running/deploying-cf/vsphere/hardware_spec.html>