採用 WebSocket 改善 SPA 應用程式效能之研究

呂庭宇, 丁建文*
國立高雄應用科技大學資訊管理系
E-mail: b790113g@gmail.com, *jwding@kuas.edu.tw

摘要

近年來，隨著網際網路技術的發展，網站已經成為現代人生活中不可或缺的，也隨著瀏覽器技術的進步，具備有更完善的支援性及相容性，使得網頁程式開發逐漸從伺服器運算轉移到用戶端運算，由傳統的 AJAX、REST 等等開發技術成長到單一頁面應用程式（Single-page Application，SPAs），透過 AJAX 技術將網站大部分的運算在使用者端的瀏覽器上處理，使用者可以在單一頁面上操作如同使用桌面應用程式般。

本研究將深入討論如何加快單一網頁應用程式的初始加載速度，並模擬一個電子商務網站進行初始頁面載入時間測試，比較傳統網站與 SPAs 網站使用 AJAX 技術的初始頁面載入時間，研究中將使用 HTML5 新制定規範中的 WebSocket API，來改善使用傳統技術的延遲，比較效能並評估其安全性，對此讓使用者有更佳的互動性與使用體驗。

關鍵詞：單一頁面應用程式、HTML5、WebSocket

Abstract

With the development of Internet technology in recent years, the websites have become the integral part of life. Along with advances in the browser technology, the websites have better supportability and compatibility. Web Programs development has transferred from the server side to the client side computing. Traditional AJAX, REST and other development technologies have grown into the Single-page Application (SPAs). By the AJAX technology, most of the site computing can process on the user side of the browser. Users can operate on a single page, such as using a desktop application.

This study will explore how to enhance the single page application for loading speed,and we simulate tests for an e-commerce website the load time of initial page, compared with traditional websites and SPAs website that uses AJAX for load time of initial page. The study will use the HTML5 to formulate WebSocket API to explore how to improve the delay from using traditional technology, and obtain comparable performance and security. Finally, we hope this study can give users better interactivity and experience.

Keywords: Single-page application, HTML5, WebSocket

1. 前言

隨著資訊科技的發展與網際網路的普及，全球資訊網中的網頁應用不斷的增加，HTML 是其描述語言，在發展最初的版本，只提供以內容為主的網頁，隨著技術的進步，使用者的需求也越來越多樣化，促使許多新的概念與功能逐漸加入 HTML 中，

2000 年後，動態網頁逐漸開始被許多商業公司採用，開發的方向越來越套裝化，為了滿足越來越複雜的需求，網頁框架的產生，MVC 設計模式的提倡，各種技術逐漸推出，例 Microsoft Sliverlight 與 Adobe Flash，可以增加介面反應速度，變化多的視覺設計，缺點是須另外安裝外掛，目前最流行的是可相容傳統網站，快速 Mashup 網站資源，節省運算資源的 SPAs 技術，為目前主流的開發方式，Google+、Facebook 等，都採用此技術開發。[5, 9, 13]

2010 年 HTML 更新到第五版，主要加強目前的功能，增加了許多實用的 API，除了標準更加統一，減少瀏覽器需安裝外掛的依賴，HTML5 是未來發展非常重要的技術，除瀏覽器外，智慧型裝置也都依標準逐漸支援。[11]

其中 WebSocket API 提供了全雙工通訊的網路技術，為全新制定的應用程式介面，能使用文件、二進位陣列等進行傳輸，建立低延遲的雙向資料交換，改善使用傳統 HTTP 標準通訊協定的速度與效能。

1.1 研究動機

SPAs 的開發方式已經成為目前的主流，但現在的網頁不像過去這麼單純，由單純分享資源的網站逐漸演變到高互動性的網站，社群網站的興起，例如 Facebook 都是更接近人與人之間的互動，介面中的組件、元件也越來越多，運作的邏輯也越來越複雜。這些組件與伺服器要求資料的方式，皆是採用 AJAX 對伺服器發送要求，若網頁需要的資料非常多與多樣化，則會對伺服器持續要求，造成網頁非常嚴重的延遲，初始頁面等待載入的時間過長。

根據網站統計，有 47%的使用者希望初始頁面載入時間在 2 秒或更少，有 25%的人不希望初始頁面載入時間多於 4 秒，如圖 1 所示，有 40%的人放棄載入一個網站超過 3 秒。[12]

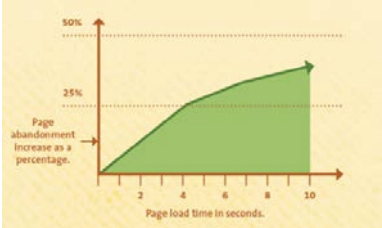

圖 1. 初始頁面載入時間統計

1.2 研究目的

本研究將針對網頁向伺服器存取資料之技術，提出改善機制，主要以 HTML5 中的 WebSocket 為基礎，透過全雙工通訊網路技術的連線，減少傳統網站為了實現推送技術，產生不必要的傳輸流量，藉此優化初始頁面加載時間。

伺服器也能主動傳送資料給使用者端，方便開發者能簡易實作聊天室等需要低延遲的網路服務，減輕網頁開發人員的負擔，增加使用者互動性與更佳的體驗，並擁有更高的資料安全性，資料皆使用數據幀（Data frame）方式傳輸，可以解決資料量大時，資料大小導致長度標誌位數不足的狀況，也與 HTTP 的分塊傳輸相同，提高傳輸效率。

2. 相關研究及技術原理

2.1 Asynchronous JavaScript and XML

Jesse James Garrett 所提出的概念，他指出這不是一種技術，而是多種技術構成，傳統的網頁應用中，使用者觸發一個請求到伺服器，伺服器回應一個完整的 HTML，這樣不能提供好的用戶體驗，而且重複加載了固定的資料，在 AJAX 中，瀏覽器會載入一個 Ajax 引擎，資料存取與響應界面由它完成，能在不更新整個畫面下改變網頁內容。[3, 10]

假設需進行頻繁的溝通，例如聊天室等接近零網路延遲的應用，則會難以實作，最古老的方式，是使用 Polling 來實作，利用計數器，每隔一段時間就對伺服器發送一次請求，雖簡單，但因大部分要求的標頭與內容都相同，也無法做資料快取，導致非常的沒有效率，也會浪費很大的頻寬。[1]

另一種方式是 Streaming，雖可以解決 Polling 所造成大量的頻寬，但卻因為瀏覽器支援的關係可用性非常的低，有人提出改良的 Comet，也是目前 Plurk、Facebook 實現動態更新的方法，為目前 SPAs 主流開發方式，但還是有佔用連線及單向溝通的問題。 [1, 4, 8]

2.2 Single-page application

在傳統的動態網頁與使用者的互動建立於多頁面的情況下，為每個請求進行整個頁面的刷新，頁面的產生完全透過伺服器運算，這樣的方式導致網頁回應的時間過長，整個畫面重整時，頻繁造成畫面的閃爍，對使用者操作上的不流暢，也造成操作介面缺乏親和性與人性化。[13]

在 2004 年之後，也是同時 Google 釋出 Gmail 的 Beta 版本，網頁技術的演算，由瀏覽器支援度的大幅提升，逐漸從伺服器轉移到瀏覽器本身，這種技術被稱為 AJAX [3]。在 SPAs 中，網頁內容可以透過建立組件來取代，促使整個頁面不會因為每個請求而重新加載，有助於提高與瀏覽者的互動性、回應速度與用戶滿意度。

SPAs 中，初始頁面加載時間是非常重要的關鍵指標，初始頁面加載時間是從使用者在瀏覽器中輸入網址，直到呈現畫面時的時間計算，有研究表示，如果使用超過 4 秒鐘來加載頁面，25%的使用者會直接關閉網頁，並在分析中顯示，這些網頁都非常冷門。[12]

SPAs 在加載頁面的時候分為下列八個階段：「使用者輸入 URL」、「網頁被下載」、「JavaScript 下載」、「JavaScript 解析」、「應用程序初始化」、「模塊加載」、「數據庫中載入數據」、「呈現頁面」，但在傳統的網站只包含其中的三個步驟，因此傳統網站在頁面載入時間上會有較好的表現，除了保持檔案大小最小，也需減少下載與解析的時間，減少每個階段所執行的時間是非常重要的。[6]

2.3 HTML5 WebSocket

WebSocket 為 HTML5 新制定的原生應用程式介面，主要是提供瀏覽器與伺服器之間的溝通，並以全雙工的資料交換模式下運行，資料可以在瀏覽器與伺服器之間直接進行傳輸，此通訊協定在 2011 年被 IETF 定為標準 RFC 6455，資料傳輸格式上也有特別定義，在傳輸上也更加安全。[2, 7]

WebSocket 採用以數據幀的方式進行傳輸，可以將數據分為多個幀（Frame），並且按照順序進行傳輸，解決當資料量大時，資料大小導致長度標誌位數不足的狀況，也與 HTTP 的分塊傳輸相同，可以一邊產生數據一邊傳送資料，提高傳輸效率。

3. 研究設計

針對 SPAs 的開發中，會將網頁分為多個區塊，我們將區塊稱之「視圖組件」，網頁中包含多個視圖組件，以電子商務網站、社群網站為例，組件必定是基本網頁的好幾倍，這些視圖組件還必須向伺服器隨時要求最新的資料，要求資料以 AJAX 最為普遍，然而此方法只支援從瀏覽器發出請求至伺服器端要求資料，除此之外，若以一個視圖組件在伺服器必須最少撰寫一隻存取資料的應用程式介面（API），則必須建立多次連線才能將資料完整

抓回，造成存取時間過長。

故本研究提出一套改善機制，以 HTML5 WebSocket 為基礎，建立持續性的連線，不需要重複不斷的建立連線，以改善原有使用 AJAX 重複建立連線機制之不足，改善 Polling 的方法，也可以由伺服器端主動傳送資料，改善 AJAX 無法主動傳輸的限制，減少往來溝通之間的延遲。

本研究將模擬一個電子商務網站，使用傳統動態網頁程式設計為基礎開發，其中包含 HTML 與 CSS 檔案，並建立網站測試內容儲存於 MySQL 資料庫內，再延伸分別使用 AJAX 技術與 WebSocket 技術進行開發測試，則傳統動態網頁不應該包含 JavaScript 檔案，單一網頁應用程式（SPAs）將會包含 JavaScript 檔案，為了保證研究實驗公平與準確性，AJAX 與 WebSocket 技術所建立的測試程式碼將會同時包含在一個 JavaScript 檔案中，實驗時以觸發不同副程式進行時間測試。

3.1 傳統電子商務網站

本研究將使用 PHP 動態網頁程式語言，模擬傳統電子商務網站，首頁內將只包含介面設計所產生的 HTML 與 CSS 樣式檔案，為確保評估一致性，該兩個檔案將在後續實驗中沿用，圖 2 為傳統電子商務網站模擬運作流程圖。

圖 2. 初始頁面載入時間統計

實驗測試載入時間採用 Google Chrome 中的 Developer Tool 為依據，下圖 3 為 Google Chrome 中的 Developer Tool 運作範例圖，實驗將運行於本機並進行實驗，減少因網路傳輸延遲所造成測量的誤差，測試伺服器主機的硬體配備為：處理器 i5-4250U Processor、OS X Mavericks 10.9.4、伺服器軟體：MAMP 3.0.5、Google Chrome：35.0.1916.153。

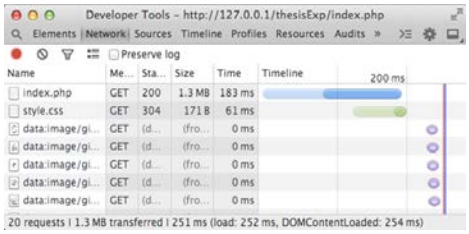

圖 3. Developer Tools 測試範例

首頁依照不同功能區塊，模擬 15 次向資料庫進行資料存取，只取得純文字的資料分別有「商品分類」、「熱門分類」、「優惠資訊」、「熱門品牌」、「購物車數量」，取得混合資料分別有「最新資訊」、「新品上架」、「優惠商品」、「暢銷商品」、每個熱門分類中的「分類商品」與「商家廣告」。

「PHP Data API」，將上述取得資料的需求，寫成對應的方法（PHP Function），並使用 PHP 內建的 MySQLi 方法對資料庫進行資料存取，回傳資料使用資料量小的 JSON 格式，除了在此實驗中使用，後續實驗將持續沿用此介面，以下為資料應用程式介面的詳細介紹：

- getShoppingCartQuantity：購物車商品數量
- getProductCategoryList：產品分類清單
- getProductTopCategoriesList：熱門分類清單
- getInformation：包含優惠資訊、最新訊息、商家廣告
- getProductTopBrandsList：熱門品牌清單
- getProductsNewArrivals：最新上架商品
- getProductSpecials：最新商品優惠
- getProductBestSellers：暢銷商品
- getCategoriesProduct：依分類取得商品

本研究中，資料庫的部份，採用與 PHP 相同且支援度高的 MySQL 資料庫，並將上述應用程式介面所用到的資料包含圖片存至資料庫，並且資料筆數固定，維持測試資料與評估的一致性，以下為資料庫中資料表的詳細介紹：

- Cart：購物車項目，包含 7 筆資料，欄位有「編號」、「產品編號」、「數量」欄位，資料大小共 16 KB
- Category：商品分類，包含五筆主分類，每個分類有三個子分類，共 15 筆資料，欄位有「編號」、「所屬分類」、「分類名稱」、「瀏覽次數」，資料大小共 16 KB
- Info：一般資訊，包含優惠資訊、最新訊息、商家廣告各五筆資料，共 15 筆資料，欄位有「編號」、「二進位圖片」、「本文」、「更新時間」、「種類」，資料大小共 320 KB
- Product：商品資訊，每個子分類有五項商品，共 75 筆資料，欄位有「編號」、「商品分類編號」、「商品名稱」、「圖片」、「品牌」、「購買次數」、「是否優惠中」、「上架時間」，資料大小共 2.5 MB

本研究產生的 HTML 與 CSS 檔案合計為 518 Bytes，以下圖 4 與圖 5 為動態網頁模擬傳統電子商務網站未載入資料前及資料完整載入後的畫面。

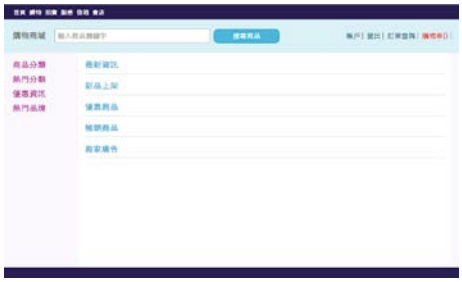

圖 4. 傳統電子商務網站（純介面）

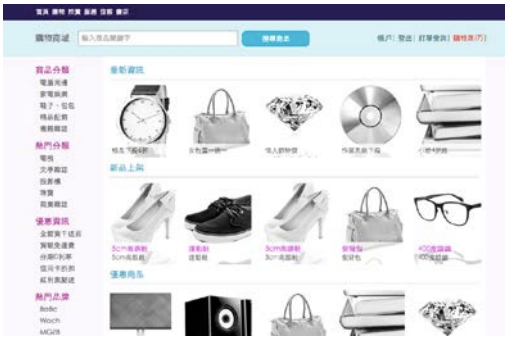

圖 5. 傳統電子商務網站（帶入資料）

實驗採用 Developer Tool 在內部網路檢測初始頁面載入時間為 250 ms 上下，若從外部網路連線則初始頁面載入時間為 500 ms 上下，如圖 6、7 所示。

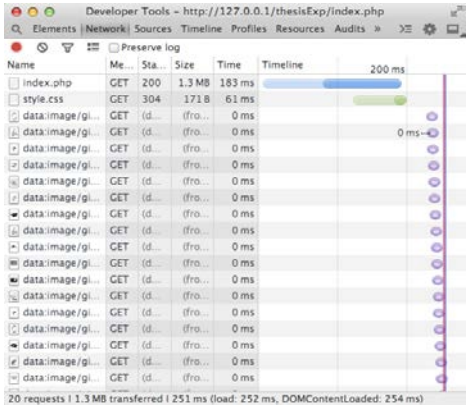

圖 6. 傳統電子商務網站初始頁面載入時間（內部網路）

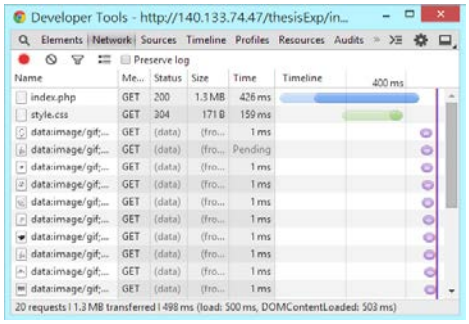

圖 7. 傳統電子商務網站初始頁面載入時間（外部網路）

3.2 SPAs 使用 AJAX 技術

在 SPAs 下，沿用前 3.1 章節所產生的 HTML、CSS 檔案，並建立一個 HTML 檔案為測試進入點（index.html），檔案內包含一個 AJAX 存取資料介面所需的 JavaScript 檔，為了保證評估一致，檔案同時包含 AJAX 與 WebSocket 的程式，利用觸發不同副程式（Function）進行區別測試。

除此之外，會增加一個 PHP Adapter 的檔案，因為 AJAX 使用的是 HTTP 方法傳輸，WebSocket 則連接 Socket 進行傳輸，此檔案用於接收是由哪種方法進行要求資料，並沿用 PHP Data API 進行資料存取，下圖 8 為模擬電子商務網站採用 SPAs 架構開發使用 AJAX 方法運作流程圖。

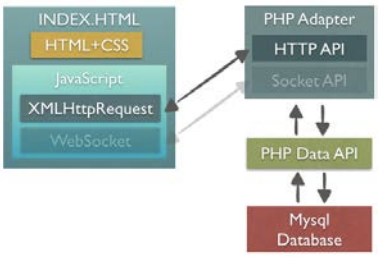

圖 8. SPAs 使用 AJAX 開發流程圖

依本研究設計架構，使用 AJAX 技術載入網頁時，會先等待 HTML、CSS 與 JavaScript 完全載入，並解析 JavaScript 檔案，接著觸發 JavaScript 中的 AJAX 方法，由 PHP Adapter 接收後，判斷是 AJAX 則使用 HTTP API 進行呼叫 PHP Data API，最後回應資料庫中的資料，並由一開始的 AJAX 方法解析資料並進行介面上顯示，依本研究實驗將會進行 15 次的 XMLHttpRequest 觸發。

考慮到開發者針對存取資料介面可能採用不同方式進行實作，本研究設計將會針對以下兩種方式進行實作與時間測試：

Sequential AJAX：整個網頁中只包含一個 XMLHttpRequest 物件進行存取 PHP Data API，網頁載入時，會先發送第一個 API 請求到伺服器，當要求完成並接收資料後，才會發送下一個請求。

Parallel AJAX：整個網頁中包含多個 XMLHttpRequest 物件進行存取 PHP Data API，網頁載入時，會同時向伺服器發送所有請求，本研究實驗將會建立 15 個 XMLHttpRequest 物件進行資料存取。

3.3 SPAs 使用 WebSocket 技術

在 SPAs 採 WebSocket 技術下，WebSocket 只要向伺服器進行一次性的連接，就可以進行雙向資料傳輸，但在伺服器，則必須使用 Socket 方式來進行溝通，本研究採用 PHP 內建的 Socket 方法建立一個可支援多連線的伺服器端，並透過命令提示字元執行 PHP Adapter 啟動 Socket API 伺服器，並沿用 PHP Data API 進行資料存取，下圖 9 為模擬電子商務網站採用 SPAs 架構開發使用 WebSocket 方法運作流程圖。

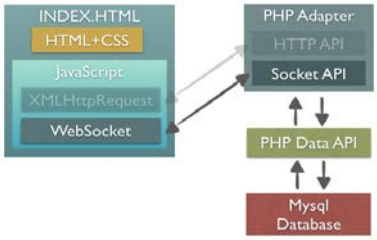

圖 9. SPAs 使用 WebSocket 開發流程圖

依本研究設計架構，使用 WebSocket 技術載入網頁時，會先等待 HTML、CSS 與 JavaScript 完全載入，並解析 JavaScript 檔案，接著觸發 JavaScript 中的 WebSocket 方法，由 PHP Adapter 接收後，判斷是 WebSocket 則使用 Socket API 進行呼叫 PHP Data API，最後回應資料庫中的資料，並由一開始的 WebSocket 方法解析資料並進行介面上顯示，由於只進行一次與伺服器連結，有效縮短與伺服器連結的延遲。

4. 效能評估

本章節主要探討本研究對 SPAs 提出採用 WebSocket 技術改善使用 AJAX 技術，縮短初始頁面載入時間的評估與效能比較，本研究時間效能比較將在內部網路以及外部網路環境進行評估，內部網路因無網路環境因素，評估數據將會比外部網路準確，實驗在相同環境下，使用 Google Chrome 瀏覽器內的開發工具，運用其工具的網路效能分析器測量本研究提出的實驗所需要的初始載入時間，並比較原有技術方法下的初始頁面載入時間，以下為內部網路環境的分析詳細結果。

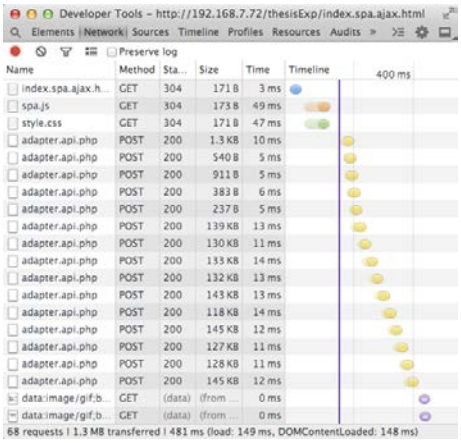

圖 10. SPAs 採用 Sequential AJAX 初始頁面載入所使用的時間（內部網路）

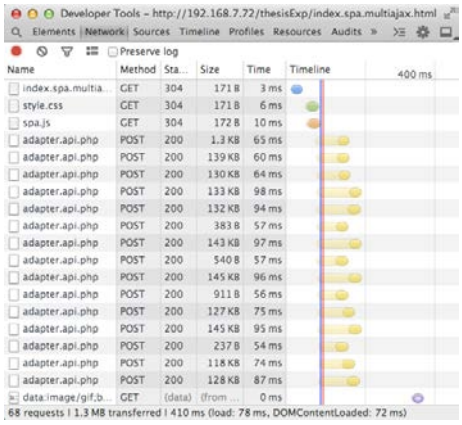

圖 11. SPAs 採用 Parallel AJAX 初始頁面載入所使用的時間（內部網路）

依本研究架構設計的實驗，圖 10 為 SPAs 採用 Parallel AJAX 方法的初始載入時間在內部網路大約為 480 毫秒；圖 11 為 SPAs 採用 Sequential AJAX 方法的初始載入時間在內部網路大約為 410 毫秒；使用傳統動態網頁載入在內部網路大約為 250 毫秒，由此實驗結果可以了解，SPAs 採用 AJAX 技術進行開發實作時，光是在內部網路進行測驗，初始頁面載入時間就比傳統動態網頁等待時間更長，以下實驗為不在內部網路進行評估並考慮實際網路環境的分析詳細結果，圖 12 為本研究實驗考慮實際網路環境測試架構圖。

圖 12. 實際網路環境測試架構圖

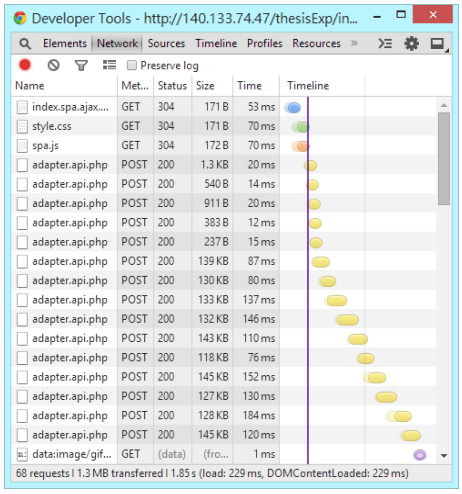

圖 13. SPAs 採用 Sequential AJAX 初始頁面載入所使用的時間（外部網路）

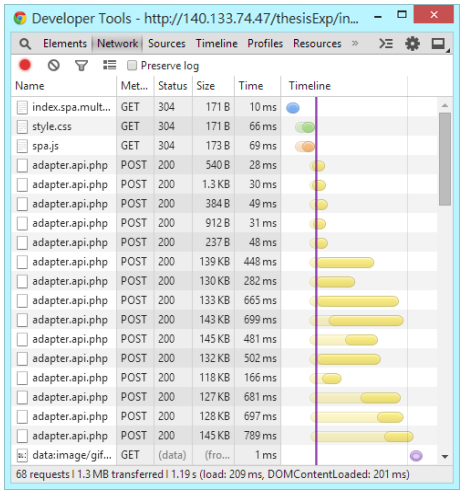

圖 14. SPAs 採用 Sequential AJAX 初始頁面載入所使用的時間（外部網路）

實驗在考慮實際網路環境時進行測試，圖 13 為 SPAs 採用 Sequential AJAX 方法的初始載入時間在外部網路大約為 1850 毫秒；圖 14 為 SPAs 採用 Parallel AJAX 方法的初始載入時間在外部網路大約為 1200 毫秒；使用傳統動態網頁載入在外部網路大約為 500 毫秒，由以上結果可以知道，無論是內部網路或是外部網路進行測試，傳統動態網頁的初始頁面載入時間皆優於 AJAX 技術。

本研究提出採用 WebSocket 進行資料存取，縮短初始頁面載入時間，如圖 15 所示，在內部網路測試初始載入時間約為 300 毫秒，若考慮網路因素，如圖 16 所示，初始載入時間約為 1000 毫秒，與先前的實驗數據進行比較，可以知道採用 WebSocket 可以大幅縮短 AJAX 技術存取數據所花費的時間，圖 17 為本研究實驗初始頁面載入時間整體評估效能圖。

本實驗經過多次測試，每次所產生的初始頁面載入時間，會因每次重新整理而不同，在內部網路環境下，誤差不會超過時間的正負 10%，在考慮網路環境因素下將會變的不穩定，但實驗數據皆比內部網路所測試的時間還高。

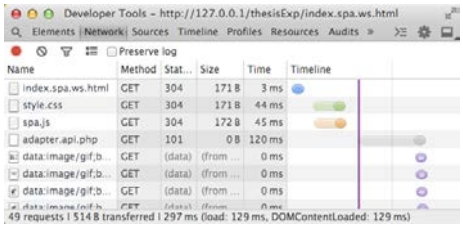

圖 15. SPAs 採用 WebSocket 初始頁面載入所使用的時間（內部網路）

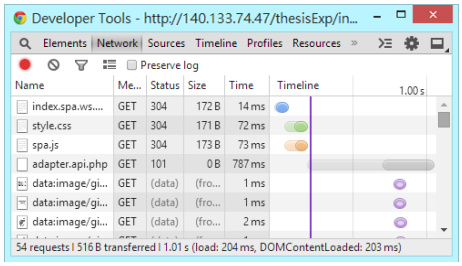

圖 16. SPAs 採用 WebSocket 初始頁面載入所使用的時間（外部網路）

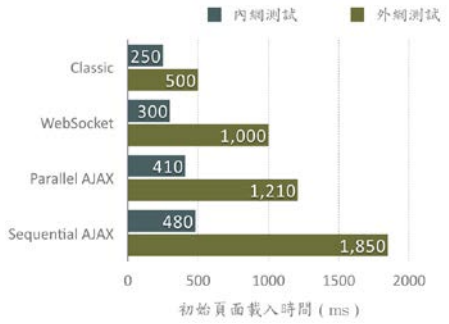

圖 17. 本研究實驗整體評估效能圖

5. 其他技術議題評估

5.1 瀏覽器支援

AJAX 在瀏覽器上的支援一直未統一，使用前必須依照不同的瀏覽器取得 XMLHttpRequest 物件，例如 IE 就必須使用 ActiveXObject，而在資料格式支援上，例如 Opera 不支援 XSL 與 XSLT，開發時必須針對瀏覽器支援進行評估， WebSocket 則無此問題。

5.2 相同來源政策

AJAX 針對不同網域之間的互動有所限制，為相同來源政策（Same-origin policy），跨來源寫(Cross-origin write)與跨來源嵌入(Cross-origin embedding)是被允許的。跨來源讀取(Cross-origin read)不被允許，在不同網域下，是無法使用 AJAX 讀取其他伺服器資料，WebSocket 則無此限制。

5.3 資料傳輸格式

以 AJAX 來說，資料傳輸的加密，必須由開發者自行設計，使用者可以使用網頁檢閱器，例如瀏覽器中的「開發人員工具」，查看每個連線的詳細內容，就算以 JSON 格式化，資料都是以明碼展示，敏感的資料就無法以 AJAX 進行存取，本研究採用 WebSocket，在資料傳輸時，協議規定 Client 傳送給 Server 的資料必須要進行 Mask， Mask 則是採

用 RFC5234 定義，傳輸都必須遵從這個協議，安全性上，需要執行解碼才可以看見，會安全許多。

表 1. AJAX 與 WebSocket 支援度綜合比較

	AJAX	WebSocket
Object Name of Browser	ActiveXObject XMLHttpRequest Msxml2.XMLHttp	WebSocket
Same-origin policy	Write(O) embedding(O) read(X)	O
Data Mask	X	O(RFC5234)

6. 結論

隨著網頁技術的興起與蓬勃發展，除了網頁的應用越來越多元化外，更是與人們的生活息息相關，因此使用者對網頁初始頁面載入時間的縮短，勢必也變得更加重要。

SPAs 借用了使用者電腦的資源進行運算，改善了傳統動態網頁完全由伺服器進行運算，降低了伺服器的成本，由於伺服器只負責資料的傳輸，也減少網路傳輸所使用的頻寬，能更快回應使用者要求的頁面項目，更彈性也更動態的網頁操作，有助於提升網站與使用者之間的互動性。

本論文提出使用 WebSocket 來改善 SPAs 採用 AJAX 技術的初始頁面載入時間，瀏覽器只需向伺服器進行一次性的連接，就可以進行雙向資料傳輸，大幅減少使用 AJAX 多次連接時所產生的連接延遲與標頭的浪費，並大幅縮短初始頁面載入時間，對使用者而言，能有更佳的使用體驗與網站互動性，針對開發者而言，除了能輕易實作低延遲網站應用與提供更好的資料安全性，且不需針對瀏覽器支援多加考慮，也能設計更人性化的介面，希望能為網站初始頁面載入時間的優化盡一份心力。

參考文獻

[1] Bozdag, E., A. Mesbah, and A. Van Deursen. A comparison of push and pull techniques for ajax. in Web Site Evolution, 2007. WSE 2007. 9th IEEE International Workshop on. 2007. IEEE.

[2] Fette, I. and A. Melnikov, The websocket protocol. 2011.

[3] Garrett, J.J., Ajax: A new approach to web applications. 2005.

[4] Gutwin, C.A., M. Lippold, and T. Graham. Real-time groupware in the browser: testing the performance of web-based networking. in Proceedings of the ACM 2011 conference on Computer supported cooperative work. 2011. ACM.

[5] Magel, K., Is it too late to put the user back into HTML? Computer, 1997. 30(12): p. 131-132.

[6] Petersson, J., Designing and implementing an architecture for single-page applications in Javascript and HTML5. 2012.

[7] Rakhunde, S.M., Real Time Data Communication over Full Duplex Network Using Websocket.

[8] Russell, A., Comet: Low latency data for browsers. alex. dojotoolkit. org, 2006.

[9] Taivalsaari, A. and T. Mikkonen. The web as an application platform: The saga continues. in Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on. 2011. IEEE.

[10] van Wijngaarden, T., Asynchronous JavaScript and XML.

[11] Vaughan-Nichols, S.J., Will HTML 5 restandardize the web? Computer, 2010. 43: p. 13-15.

[12] Work, S. How Loading Time Affects Your Bottom Line. Available from: http://blog.kissmetrics.com/loading-time/.

[13] 郭惠煌 and 蕭裕正, AJAX 網頁設計技術於校務行政資訊系統之應用. 網際網路技術學刊, 2007. 8(2): p. 185-190.