快取適性化之網頁應用程式設計
Cache-aware Design of Web Applications

張燕光，蔣坤霖
中華大學資訊管理學系
{ykchang,mi86039}@mi.chu.edu.tw

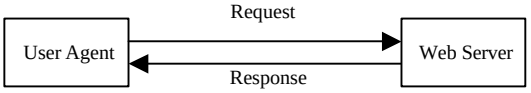
摘要

隨著近年來網路技術的進步與電子商務的興起，網頁應用程式（Web Application）已成為現今網路服務系統撰寫方式的主流，雖然其對人們所帶來的利益是無可記數的，但其所產生的網頁卻不適於為網路上的快取（Cache）機制所暫存，造成網頁伺服器（Web Server）的寶貴資源被重複的浪費，也使已日顯壅塞的網路頻寬更顯缺乏，本文即欲提出一網頁應用程式的撰寫流程方法，使得由網頁應用程式所產生的網頁，能儘可能的被暫存於網路上任何擁有快取機制的平台上，如代理伺服器（Proxy Server），以期能令網路頻寬的壅塞現象能獲得有效的舒緩，並使網頁伺服器能大量減少重複處理相同要求的情況，讓網頁伺服器能持續保有最佳效率以處理龐大的服務要求。

關鍵字：網頁應用程式（Web Application）、快取（Cache）、代理伺服器（Proxy Server）

壹、序論

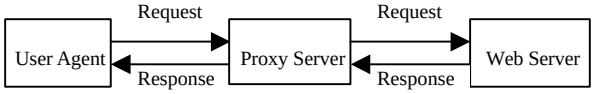
近幾年來，網路已成為現代人不可或缺的一部份，由於網路具有無遠弗屆與易於傳達資訊的特質，人們可藉由網路與遠方親友聯絡感情，也可在網路上散佈或瀏覽不同的資訊，而欲使用以上的功能，在現今也有許多不同的方式可供選擇使用，其中最為普遍且簡便的方法，當屬使用瀏覽器（Browser）搭配利用超文字標記語言（Hypertext Markup Language）所撰寫出來的網頁（Web Page）來達成，其所使用的網路協定為超文件傳輸協定（HyperText Transmission Protocol），負責整個網頁傳輸的流程與機制，如下簡圖一：



圖一、不含代理伺服器之 HTTP 傳輸流程

每當使用者想要瀏覽網頁伺服器上所存放的網頁時，使用者便可使用用戶代理器（User Agent）軟體，也就是瀏覽器，向網頁伺服器提出要求（Request），當網頁伺服器收到用戶代理器的 Request 時，便會將所要求的網頁回應（Response）給使用者，並在將網頁回應之前先回應一個 HTTP 狀態碼，與搭配這一個 HTTP 狀態碼的簡短文字聲明給使用者，以告知網頁伺服器對使用者所提出的 Request 處理的狀態，藉由以上的方式，使用者就可自由瀏覽所選擇的資訊。

不過隨著網路的快速成長，網路上的交通卻日顯壅塞，在其中常有相同的網頁被要求、傳輸，於是這造成寶貴的網路頻寬被浪費，若是能將這些相同的網頁被集中、收集在鄰近的某台伺服器上，便可減少網路壅塞的情況，使網頁能快速的為使用者所觀看，於是便有快取（Cache）機制的產生，如下簡圖二：



圖二、含有代理伺服器之 HTTP 傳輸流程

用戶代理器向鄰近的代理伺服器（Proxy Server）提出要求，此時代理伺服器便會檢查自己是否暫存有所被 Request 的網頁，若有，則立即將所暫存的網頁 Response 予用戶代理器，若無，則向網頁伺服器提出 Request，將網頁伺服器所 Response 的網頁再 Response 予用戶代理器，並將該網頁暫儲起來，待下次若有用戶代理器提出相同的 Request，便將其所暫存的網頁立即的 Response 予用戶代理器，於是，在這種快取的機制之下，網路壅塞的情況便可獲得解決，而使用者也可較為快速的瀏

覽網頁。

現今的網路世界中，網頁應用程式被大量的開發與應用已是一項不爭的事實，在本文之中，我們欲在現有的快取機制之下，發展一適性於快取機制下的網頁應用程式撰寫流程方法，一開始除了先前所簡述的 HTTP 傳輸協定外，也將探討現有之快取機制不足以適用於網頁應用程式的因素所在，並以網頁應用程式方面為考量，逐步的闡述網頁應用程式需有之功能架構與處理流程，令所產出之結果網頁能為擁有快取機制之網路平台所暫存，之後再以一實驗來得知使用靜態 Html 檔案與使用網頁應用程式直接回應之間的效率差別程度，最後統整出一結論。

貳、問題定義與文獻探討

上述的網頁快取的運作流程與機制雖可稱完善，但仍有其不足之處，其中之一就是無法快取由網頁應用程式所產生出來的結果網頁，近年來，隨著網頁技術的進步與電子商務的興起，與人們期盼能更進一步與網路世界產生互動的希望，由網頁應用程式所產生的結果網頁已逐步取代現有只由單純 Html 語法所構成的網頁，舉凡網路上常見的功能，如資料的搜尋、股票價格的檢索、網路訂購商品與信件的收發等等，皆為由程式設計者所設計出的網頁應用程式所提供的功能而致，程式設計者利用網頁程式語言，如 CGI、ASP、PHP、JSP 等等，構成縝密而複雜的處理流程，搭配上網頁伺服器，使得人們的需求獲得解決，因為這些網頁應用程式具有傳遞與處理用戶代理器所送出參數與其相對應之數值的功能，可將使用者所做的選擇或輸入的字句作適當的處理，通常這些參數與其相對應的數值是夾帶在用戶代理器的 Request 中送出，可置入 Request 的 URL（Uniform Resource Locator）中之後，如 http://site/example.php?k1=v1&k2=v2，或是隨著 Request 時的表頭（Header）送出，前者稱為 GET 方式，後者稱為 POST 方式，而無論是何種方式，代理伺服器皆預設為不將其暫存於其中，也正因為如此，使得網路壅塞的情況不能有效的獲得抒解，同時也造成了網頁伺服器沈重的負擔，舉例來說，在一廣受歡迎的網頁伺服器上提供著股價資訊，使用者可自由選擇觀看不同的股價，而這股價資訊在一分鐘之內並不會變動，若在這一分鐘之內有著一千位使用者同時選擇觀看，於是在這分鐘內，網頁伺服器必須重複相同動作、送出同樣的資料，於是由此可見網頁伺服器與網路的資源有相當多的部分是被浪費了，不過程式設計者仍可以藉著送出 HTTP 傳輸協定裡所制訂的 Cache-Control HTTP Header，來控制網頁應用程式所產出的結果網頁可否被用戶代理器或代理伺服器暫存與被暫存的時間，但這些表頭在許多常見的網頁應用程式中並未被廣泛應用，同時光是使用這些表頭，對網頁應用程式來說也是不足夠的，網頁應用程式設計者必須更積極有效的讓由自己所設計出來的網頁更適於被暫存，以讓使用者能更加快速與正確的得到網頁應用程式所產生的結果網頁。

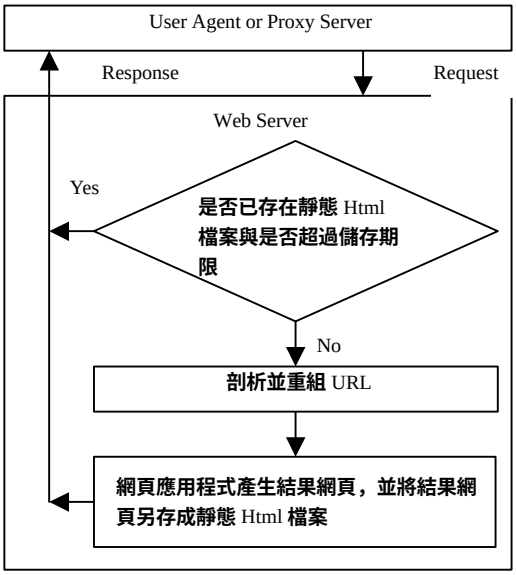
參、現存快取軟體套件之探討

為因應以上的問題，現今正有許多的擁有快取機制的軟體套件被設計開發，以利用 PHP 網頁程式語言撰寫的軟體套件而言，就有 Cacher 、 phpCache 、 jpCache 、 PEAR Cache[http://php.weblogs.com/php_debugger_cache#htmlcache]等等，雖說種類與數量可謂不少，但其主要的快取機制之作法則是大同小異，共同的特徵在於當網頁伺服器接收到 Request 時，便馬上需轉交這些快取套件來做處理，皆須有一 PHP 程式檔案來判別是否快取套件本身已有所要求的結果網頁暫存檔案，以及判別這些暫存檔案是否以超出暫存的期限，若皆無，則將暫存檔案取出，透過網頁伺服器 Response 至原 Request 端，不然則需另外再產生網頁伺服器的處理行程（Process），待網頁應用程式將結果網頁產出後再將其暫存下來，再以這些暫存檔案 Response 至原 Request 端，對由 Request 端至網頁伺服器端路徑上的代理伺服器或使用者本身的用戶代理器而言，皆可將其暫存，只要 Request 時不要使用如 http://site/example.php?k1=v1 之類格式的 URL 便可，故的確可節省網路資源用於傳輸相同資料的浪費，但對網頁伺服器而言，每次皆須使用一 PHP 程式檔案來進行判別，每次的 Request 皆為一新的 Request，即每次的 request 由網頁伺服器所 Response 的 HTTP 狀態碼永遠是 200（OK：表示網頁伺服器已成功處理請求，而且請求中的文件也已經附加進來了），也就是說皆須浪費網頁伺服器的處理資源來重新判別處理一遍，而不是在理想的

快取機制中，網頁伺服器所應有 Response 的 HTTP 狀態碼：304（Not Modified：表示請求中的文件自從指定的日期之後，並沒有被修改過，客戶端應使用其快取區中的副本），故以網頁伺服器本身的系統資源而言，處理的時間永遠是浪費的。

肆、適性於快取機制之網頁應用程式處理流程

於是在此提出一種網頁應用程式撰寫方式，期望能對上述的問題有著正面的幫助，整體概念運作流程如下圖三：



圖三、適性於快取機制之網頁應用程式之概念處理流程圖

在整個流程中，主要與傳統網頁應用程式撰寫流程方式相異之處，在於以下幾點：

一、選適於被暫存的結果網頁

在開始著手將網頁應用程式所產生的結果網頁適性於暫存之前，網頁程式設計者必須先評估有哪些結果網頁是適合被快取機制所暫存的，考慮的因素有以下三點：

1. 結果網頁的主要用途：絕大多數的結果網頁用途皆可適宜於將其暫存，但若其涉及隱私性、安全性，或當中使用有 session 或 cookie，而如其用途為使用者認證、電子交易等，或內容含有個人身份資料、機密文件資料等，即不適宜將其暫存，以免造成資料外洩，使網站信譽受損。
2. 結果網頁所含內容的變動頻率：網頁程式設計者應需瞭解結果網頁所提供之內容，並評估其內容變動頻率，以便決定該結果網頁是否適於於暫存，或是其被暫存的期間長短，如結果網頁中含有股價資訊，而其股價在一分鐘之內並不會有所變動，則該網網頁便適於被暫存，同時可被暫存的時間為一分鐘。
3. 結果網頁可否預先產出：包含上述二點考慮因素，網頁程式設計者尚須檢視結果網頁可否預先產出，可由結果網頁的使用對象是否有針對性開始考慮起，若結果網頁的內容對所有的使用者而言皆為相同，如股價資訊對所有人而言皆為相同，則該結果網頁便適於預先產出，若結果網頁的內容對不同的使用者而有所變動時，則便不適於將其預先產出，不過這並不是絕對的，如在一 Web Mail 的網頁應用程式當中，使用者之間的信件匣列表並不相同，但一般而言，使用者本身對已自行分類建立的信件匣並不會經常加以變動，此時網頁程式設計者可選擇將其列表網頁預先產出，以加快結果網頁的瀏覽速度，也可減輕網頁伺服器的負擔。

由以上三點評估因素，我們便可製作一結果網頁暫存評估表，如表一。

二、傳遞參數值的方式需改變

依標準的作法，用戶代理器或代理伺服器若需將參數與其相對應之數值傳遞至網頁伺服器上的網頁應用程式，可選擇使用 GET 或是 POST 模式，在 GET 模式中，參數值的傳遞是依附在 URL 之後，如 http://site/example.php?k1=v1&k2=v2，只要在 URL 之後多加一"?"，並將參數 k1 與 k2 及其值 v1 與 v2 添上並以"&"作為區隔即可，但正由於中間有一"?"，使得多數的用戶代理器或代理伺服器並不將之暫存（Cache），使得若為含有同樣 URL 的 Request 送達至網頁伺服器，皆會再重新處理一遍，使得整體處理效率無法提高，故在本文作法中，利用 URL 傳遞參數的方式必須改變，使得用戶代理器或代理伺服器皆能將網頁應用程式所產生的結果網頁暫存於本身之中，依上述範例，在本方法中，應改為如下之 URL 模式：http://site/example.php_k1=v1_k2=v2.html，當有著如此的 Request 送達至網頁伺服器時，搭配上網頁伺服器所具備有剖析與重組 URL 的能力，如 Apache 網頁伺服器的 mod_rewrite

表一、結果網頁暫存評估表

程式	用途	變動頻率	可否暫存	暫存期間	預先產出
Mail_Writer_Html_1.php	供使用者寫信之用	無	是	1 週	可
Mail_Box_List.php	供使用者瀏覽信件匣列表之用	低	是	3 天	可
Outer_Mail_Box_Edit_1.php	供使用者編輯外部信箱之用，當中含有外部信箱登入名稱與密碼	中	否	—	否
View_Mail_Control.php	供使用者選擇觀看覆信件之用	高	否	—	否

模組，首先網頁伺服器先檢視是否有相同檔名的靜態 Html 檔案的存在，若存在，則直接將該檔案 Response 予原 Request 端，若檢視結果為不存在，則將該 URL 拆解並重組成傳統標準 URL 的模式，並交與網頁應用程式做接下來的處理，至於 POST 模式則應盡量避免使用，除了使用 POST 模式傳遞的資料並不能被用戶代理器或代理伺服器所暫存外，另外因為在 GET 模式中，資料隨著 URL 而傳送，資料的大小就受限於用戶代理器 URL 長度的限制，因此可經由 URL 傳送的資料著實有限，同時由於資料附於 URL 當中，相當容易為他人所窺見，故除非參數的內容中含有較為涉及機密或隱私的資訊時，或是需傳遞大量資料時，否則應多使用前述模式取代，讓結果網頁能盡量被用戶代理器或代理伺服器所暫存，以使網路與網頁伺服器的資源浪費能減少至最低程度。

三、網頁應用程式需將網頁輸出成靜態 Html 檔案

在本文所想達成的目的中，其一就是盡可能的減少系統處理的資源浪費，讓擁有相同內容的結果網頁不需再重新處理，故網頁應用程式需將所產生的 Web Page 輸出成一靜態的 Html 檔案保存於網頁伺服器中，當中若有著對網頁應用程式本身的鍊結（Link），需依上一點所提及的作法產生，使網頁應用程式內的所有程式皆能依本方法來為用戶代理器或代理伺服器所讀取與暫存，另外在所產生的靜態 Html 檔案中，需註明此靜態 Html 檔案所產生的時間令使用者知悉，並加上含有標準 URL（如 http://site/example.php?k1=v1&k2=v2）的鍊結方式，供使用者若欲讀取最新資料內容或欲令網頁應用程式重新處理產出結果時使用，以減低一致性（consistency）的問題發生，而檔案名稱需依上一點方式命名，如 example.php_k1=v1_k2=v2.html，待下次含有相同參數的 Request 來要求網頁伺服器回應時，便可直接將該靜態 Html 檔案 Response 予原 Request 端，不需再耗費系統資源來處理。

四、背景預先產出靜態 Html 檔案

雖然在上一點中指出將結果網頁輸出成靜態 Html 檔案，可使整體網頁系統回應效率提升，但靜態 Html 檔案產出的時機僅於當有 Request 送達至網頁伺服器時，使得使用者永遠僅於須待下次送出相同 Request 時才可受惠，否則尚須忍受因網頁應用程式即時處理所帶來的瀏覽時間延遲，也就是說到目前為止，整體處理流程皆為被動的，於是在本文作法中，網頁應用程式可定期或於某一特定時機，如當網頁伺服器未處於高使用率的狀態下，進行背景作業，預先將先前所評估過可預先產出的結果網頁輸出成靜態 Html 檔案，使得首位提出 Request 的使用者也可感受到本文作法所帶來的整體瀏覽效率提升，故使本文作法較其餘作法更多了一項主動性。

五、制靜態 Html 檔案可被暫存的期間

網頁應用程式也需提供定期清除靜態 Html 檔案的功能，讓已存在多時的靜態 Html 檔案能被自動刪除，以讓使用者能取得較新的結果網頁，也必須同時提供控制靜態 Html 檔案為用戶代理器或代理伺服器所暫存期間的功能，以減少發生網頁伺服器現有與用戶代理器或代理伺服器所暫存之靜態 Html 檔案有不一致的情況發生，在本文作法之中，可使用 Apache 網頁伺服器軟體所提供之功能與其 mod_expires 與 mod_headers 模組，在存放靜態 Html 檔案的目錄中，放入.htaccess 檔案，以可分別控制，其內容可為下例：

```
### activate mod_expires
ExpiresActive On
### Expire .gif's 1 month from when they're accessed
ExpiresByType image/gif A2592000
### Expire everything else 1 day from when it's last modified
### (this uses the Alternative syntax)
```

```
ExpiresDefault "modification plus 1 day"
### Apply a Cache-Control header to index.html
<Files index.html>
Header append Cache-Control "public, must-revalidate"
</Files>
```

六、完整處理流程

接下來開始說明本文作法的詳盡流程說明，如圖四，當收到來自用戶代理器或代理伺服器的 Request 時，網頁伺服器便開始進行 Request 中的 URL 剖析，首先判斷是否為 http://site/example.php?k1=v1&k2=v2 之類格式（以下以 Type A URL 格式代稱之），若是，則轉交於網頁應用程式開始進行動態處理以準備產生結果網頁，若結果網頁中含有指向網頁應用程式本身內適於暫存之其他結果網頁的 URL 時，則皆須將其以 http://site/example.php_k1=v1_k2=v2.html 之類格式輸出（以下以 Type B URL 格式代稱之），以讓由用戶代理器至網頁伺服器之路徑上的所有擁有快取機制的網路平台皆可將 URL 所指的網頁暫存，接下來則開始判斷此結果網頁是否適於被暫存，網頁程式設計者可依先前所完成之結果網頁暫存評估表來決定，若結果網頁不適於被暫存，則應立即將結果網頁 Response 至原 Request 端，若適於被暫存，則將原先 Type A URL 剖析重組為 Type B URL 格式，並將結果網頁另存檔成靜態 Html 檔案，存檔時的名稱應以 Type B URL 格式中的網頁檔案名稱為命名方式，若已有相同檔名之靜態檔案存在，則將之以新產生之靜態 Html 檔案取代，同時設定結果網頁可被暫存的期間，暫存的期間也可依先前所完成之結果網頁暫存評估表來決定，而後立即回傳至原用戶代理器或代理伺服器。

若 Request 為 Type B URL 格式，則網頁伺服器便以此檢視是否該 Type B URL 所指之靜態 Html 檔案是否存在，若並不存在，則再將 Type B URL 剖析重組為 Type A URL 格式，再重新以 Type A URL 進行處理流程。若該靜態 Html 檔案存在，則還需判斷該靜態 Html 檔案是否已存在逾期了，若尚在結果網頁暫存評估表裡所決定的暫存期間之內，則將該靜態 Html 檔案回傳至原用戶代理器或代理伺服器，若已超出暫存期間，則首先將原檔案刪除，並將 Type B URL 剖析重組為 Type A URL 格式，再重新以 Type A URL 格式進行處理流程。若 Request 中之 URL 不為 Type A 或 Type B URL 格式，則表其所要求之網頁為單純 Html 網頁，網頁伺服器應立即將之回傳至原用戶代理器或代理伺服器。

在上述的處理流程中，URL 的剖析與重組為關鍵之所在，雖然乍看之下有其繁複度在，但實際上當網頁伺服器本身具備此項能力，流程內的絕大多數處理與判別皆可在單一的網頁伺服器處理行程（Process）中完成，除非真須網頁應用程式產生結果網頁時，否則並不需仰賴其他的處理資源，故伺服器的實體處理資源並不會受到太大的影響，整體處理的速度與效率必定會較為使用本處理流程前提升不少，同時所產生的靜態 Html 檔案相當的適於為快取機制所暫存，故也可大幅減少相同的網頁資料重複於網路上傳輸，使得網路壅塞的情況也可藉此獲得相當的改善。

七、效能評估

本節將推估在使用本文作法前後所能帶來的利益成長，在此以一現有之 Web Mail 系統為例，首先記錄統整個別網頁程式的 Request 數與所產生之結果網頁的傳輸位元組，如表二：

表二、網頁程式使用率記錄表

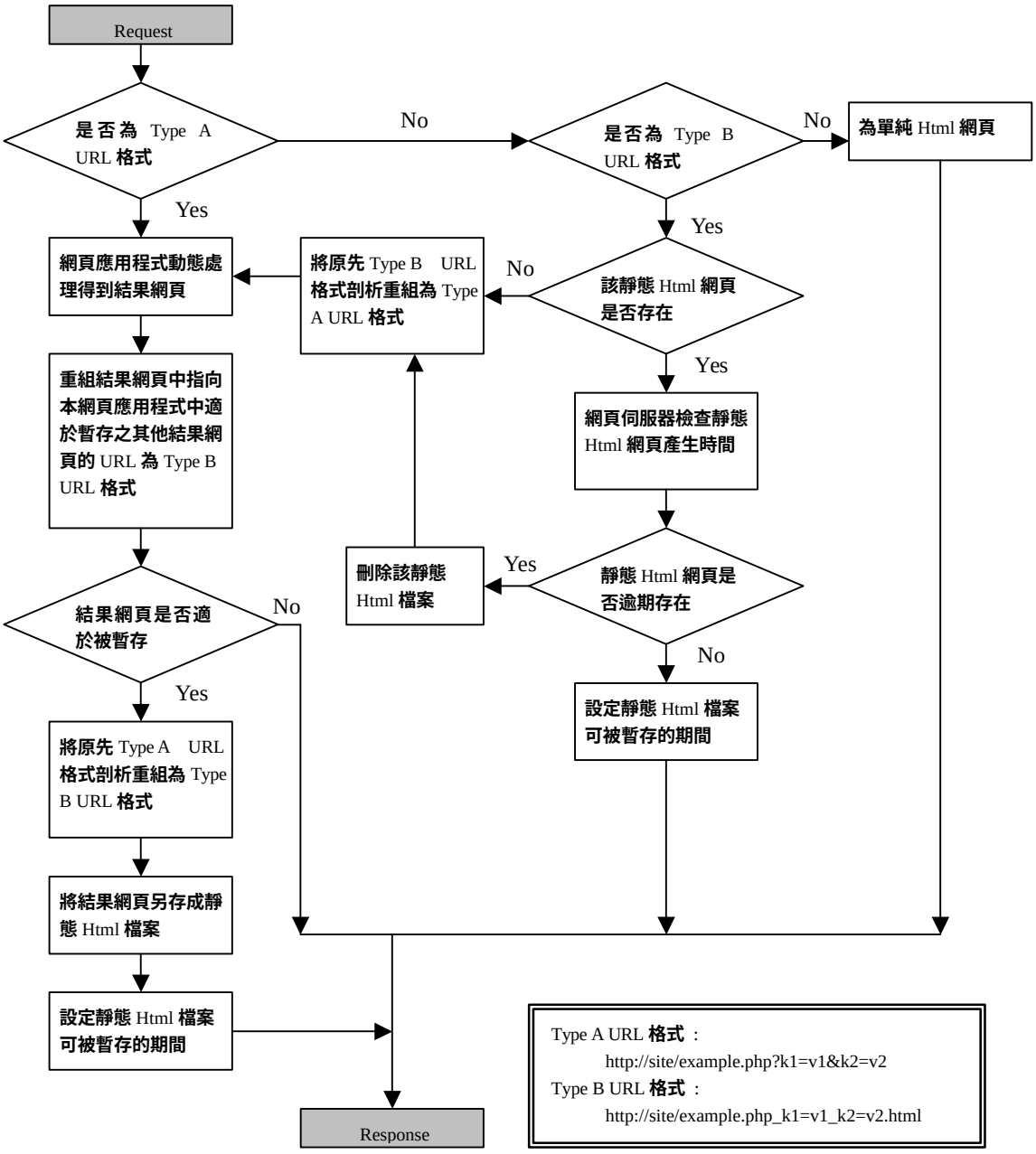
程式	% request	% byte
List.php	17.02%	68.91%
View_Mail_Main.php	8.15%	0.50%
View_Mail_Control.php	7.42%	2.90%
…	…	…
Calendar_Day.php	4.84%	14.53%

接下來根據結果網頁暫存評估表與上列記錄表，可再製作一統計表，如表三：

表三、網頁程式適用本文作法統計表

	%總 request	%總 byte
適用本文作法之網頁程式	67.34%	75.23%
不適用本文作法之網頁程式	32.66%	24.77%

由表三可得知，若使用本文作法，網頁伺服器可節省處理於重複相同之 Request 的可達 67.34%，而因為將結果網頁存成靜態 Html 檔案而能暫存在用戶代理器或代理伺服器上，所能節省網路傳輸相同內容網頁內容之部分更可達 75.23%。



圖四、適性於快取機制之網頁應用程式之詳細處理流程圖

伍、實驗設計與結果分析

為了能實際確知若能將網頁應用程式所輸出的結果以靜態 Html 檔案存於網頁伺服器上用以輸出所能帶來的優點，與實際測知使用靜態 Html 檔案與使用網頁應用程式直接輸出結果網頁所帶來效率上的差距，故進行了此項實驗。

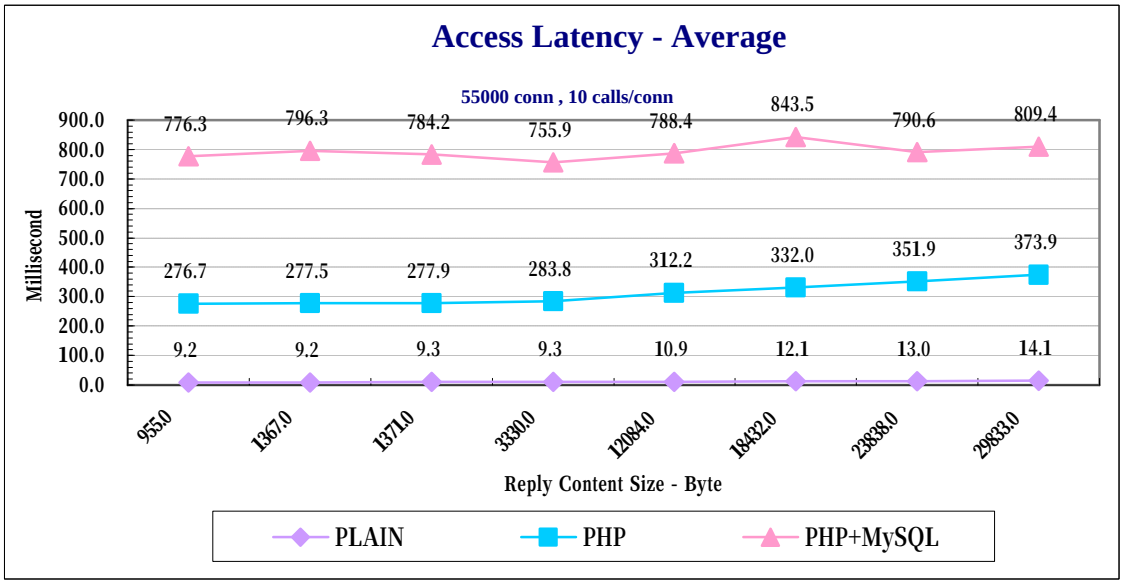
一、實驗環境與設計

實驗所用之量測工具為 httperf（HTTP Performance Measurement Tool）[http://www.hpl.hp.com/personal/David_Mosberger/httperf.html]，使用之版本為本文撰寫時的最新 0.8 版，此量測軟體提供了多樣化的參數選擇，如可設定 httperf 每秒內對網頁伺服器提出 Request 的數目（num-call）與 Request 總共提出的數目（num-conns），以供對網頁伺服器產生不同的需求工作量，並產生各方面詳細的數據，藉此量測網頁伺服器的服務表現情形。

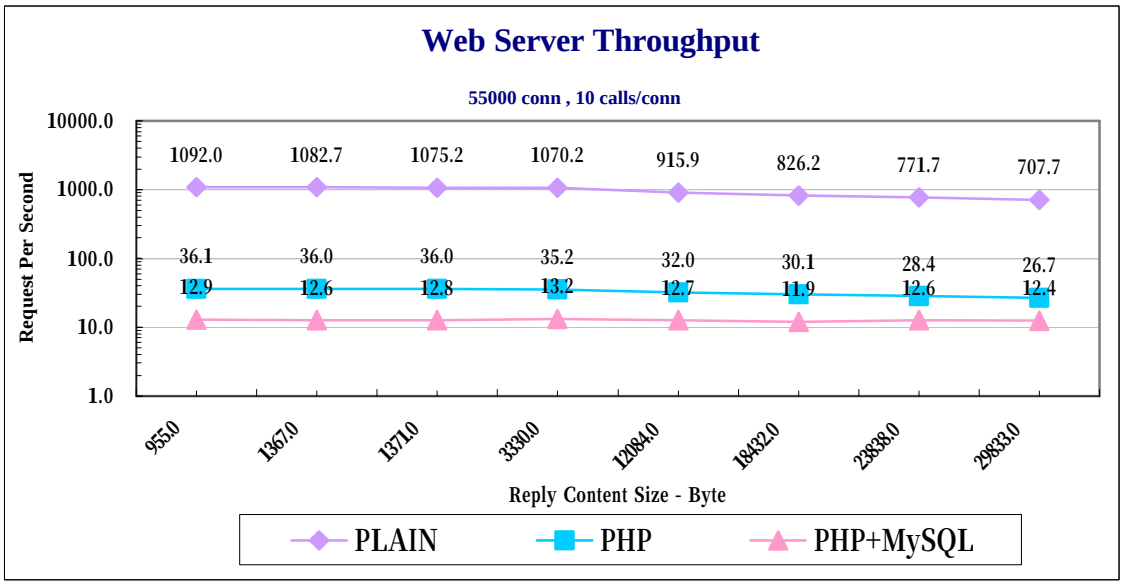
在此實驗中，共可分為三大類，分別為 PLAIN HTML，PHP，PHP + MySQL，PLAIN HTML 即為靜態 Html 檔案，在檔案

中直接存放有所有的資料，不需另外存取，PHP 為表直接利用 PHP 語言處理使用者的 Request，而所需的資料則存放於一文字檔案之中，當 PHP 處理時會利用一程式迴圈依序將文字檔案中的資料取出並將其所得結果直接輸出至 Request 端，而 PHP + MySQL 則表利用 PHP 語言處理使用者的 Request，而所需的資料則存放於 MySQL 資料庫中，當 PHP 處理時會利用一程式迴圈依序將 MySQL 中的資料取並將其所得結果直接輸出至 Request 端，每一類所產出的網頁可見內容皆為相同，在 httperf 的實際量測下也證實擁有相同的 Reply Content Size，故主要差異點僅為產生結果網頁的方法不同。

實驗的環境皆為在與實際網路隔離的本地端完成（localhost），即為 Request 端與 Response 端皆為同一處，實驗時並不將實驗機器與網路連結，以去除網路裡不確定因素的干擾，並將系統內大部分無關的處理行程砍除，以減少系統內任何因素的干擾，同時也可將系統效能提升。



圖五、平均網頁讀取時間延遲



圖六、網頁伺服器每秒可處理的 Request 數

二、實驗的軟硬體設備

表四、實驗所用軟硬體設備

Hardware		Software	
CPU	PIII 450	OS	RedHat Linux 7.0
RAM	196 MB	Web	Apache 1.3.14
HDD	Maxtor 10G 7200rpm	PHP	PHP 4.0.3pl1
NIC	Intel 82559	SQL	MySQL 3.23.28
		Measurement Tool	httperf-0.8

表五、實驗所用評估軟體參數設定值

httperf environment			
client	0/1	recv-buffer	16384
server	localhost	num-conns	55000
port	80	num-call	10
send-buffer	4096		

三、實驗結果：

如圖五、圖六，各圖所測量的目的為：圖五：在於測量整個 TCP 連結（TCP connections）於起始（initiated）至終結（closed）所需花費的平均時間，以得知平均網頁讀取時間的延遲程度；圖六：在於測量每秒內平均共有多少的Request能獲得網頁伺服器的回應。

四、實驗結論：

可以發現的是，在擁有相同內容（也可說是擁有相同 Reply Content Size）時，使用靜態 Html 檔案在各方面會遠比使用 PHP 語言或是 PHP 語言加上 MySQL 資料庫來產生或傳遞網頁要來的穩定、快速與有效率，故由上列各圖中皆可得到一共同結論，使用靜態 Html 檔案的確可使網頁伺服器的負擔大幅度的減輕，使得大量的 Request 皆能較純粹使用網頁應用程式更能在最短時間內令 Request 端得到網路伺服器的 Response。

陸、結論

一個成功的網頁服務站台，除了必須提供豐富的內容外，完善的網頁應用程式功能更是吸引人潮光顧所不可或缺的基本條件，現今多數的網頁應用程式在設計之初，僅僅只考慮到功能上的完整性與實用性，卻忽略了使用者也同時要求能在最短的時間內即能取得結果網頁，於是如何縮短網頁下載時間、減少頻寬使用量與減輕網頁伺服器的負擔，便為現今網頁程式設計者所必須同時注重的幾個關鍵問題。

在本文之中，我們發展了一網頁應用程式的撰寫流程方法，使其能適性化於快取機制之下，藉由製作結果網頁暫存評估表，到關鍵的 URL 剖析與重組與將結果網頁輸出成靜態 Html 檔案，以及背景預先產出靜態 Html 檔案和控制靜態 Html 檔案可被暫存的期間等幾個主要處理方法，構成一連串完整並富有效率的處理流程，以最務實及看似繁瑣卻簡易的作法，使多數由網頁應用程式所產生的結果網頁皆能為代理伺服器或用戶代理器所暫存，令使用者能在最短的時間內取得結果網頁，並有效的減少網頁伺服器處理的資源與網路頻寬的使用，讓網頁伺服器能持續保有最佳的處理效率，也使網路壅塞的情況獲得舒緩，故整體的服務品質便能獲得提升，使用者對網站的向心力便會提高，所帶來的有形與無形的利益是無可記數的。

參考文獻

[蒲俊男，2000] 蒲俊男、張燕光，2000，”整合性 Proxy 設計與實作研究”，中華大學碩士論文，新竹，台灣.

[Challenger，1999] Jim Challenger、Paul Dantzing，”A Scalable System for Consistently Caching Dynamic Web Data”，IBM Research，T. J. Watson Research Center.

[Iyengar，1997] Arun Iyengar、Jim Challenger，1997，”Improving Web Server Performance by Caching Dynamic Data”，IBM Research，T. J. Watson Research Center.

[Nottingham，1999] Mark Nottingham，1999，”Caching Tutorial for Web Authors and Web Master”，http://www.mnot.net/cache_docs/ .

[Zhu，2001] Huican Zhu、Tao Yang，2001，”Class-based Cache Management for Dynamic Web Content”，Dept. of Computer Science，University of California.