一個支援語音功能之 Web 節目發行及訂閱系統
A Web Program Distribution and Subscription System Supporting Audio Media

蔡尚榮 董仲愷 許欽貴 李昱
Shang-Rong Tsai Chung-Kie Tung Chin-Kwei Shu Yu Lee

國立成功大學電機工程研究所
Department of Electrical Engineering, National Cheng-Kung University, Tainan, Taiwan, R.O.C.
{tsai,tung,ckshu,lily}@turtle.ee.ncku.edu.tw

摘要

在本論文中我們提出一個支援 RTSP Proxy 功能的 Web 節目發行及訂閱系統。這個系統結合了 Server Push 與 Proxy 技術，透過這個系統，Web 資訊的提供者能方便地將自己的站台資料以節目方式提供給使用者訂閱，使用者訂閱這些網頁節目資料後，系統會將所訂閱的節目內容預取到本地的 proxy server 上並自動做更新的工作。使用者瀏覽網頁內容時，不需花費時間在等待網路資料傳輸所產生的延遲上。

Abstract

In this paper, we present a web program distribution and subscription system supporting audio media based on server push and proxy techniques. With our system, the content provider can distribute their data to users via user ‘s subscriptions. After users subscribe to the programs with our system, it will cache the data to the local proxy for the users and refresh the data automatically. The user won’t waste time on waiting for the data to transfer when browsing the pages.

1. 序論

網際網路在近年日漸普遍，尤其 WWW (World Wide Web) 的出現，其上文字、聲音、圖形和影像等多種資訊，使得資訊的傳遞更加的生動活潑，一般大眾已經很習慣利用 WWW 來取得資訊，但是現今 WWW 的運作還是有其不方便的地方。提供資訊者沒有很方便的管道宣傳其網站，使用者也不能很方便的找到所需要的資訊。而且因為網路頻寬不足的關係，使用者需等待很長的時間才能獲得資料導致使用上的不方便。此外，對於多媒體的傳輸播放也沒有標準的方式，通常是將多媒體檔案完全下載到 client 端再播放，若是一段很長的資料則使用者需等待很久才能開始播放。

現在有很多技術來解決這些問題，包括使用 Proxy Server 和伺服器推送技術[5,6,7] 等等，但是都有其不足之處。Proxy 可能回應給使用者過期的資訊，導致使用者無法或的最新的資料，或者使用者得利用瀏覽器的 RELOAD (重新整理) 功能去取得最新的資料，但這樣有時反而讓使用者更不方便。目前的各種伺服器推送技術通常需要在使用者端加裝軟體，而且當使用者沒有開機上網就無法運作。

針對聲音等多媒體資料需將檔案下載再播放這個問題，Netscape 和 Progressive Network 這兩家公司共同制訂了一種新的 URL：RTSP (Real Time Streaming Protocol)[8]。支援此 RTSP 的 Browser 或 Player 可以即時地撥放這些多媒體資料而無須先進行下載的動作。另外 RTSP 定義了一些不同於 HTTP[1][2] 的 Method，使得它可以做到快轉、倒帶的動作，甚至利用

Multi-Session[9] 達到 Video Conference，線上教學等目的。但是目前的 proxy server 均不支援 RTSP，使得 RTSP client 端在播放多媒體資料時均是連回原資料所在站台。當資料站台與使用者距離過遠或是兩者間頻寬不足時便可能導致無法獲得令人滿意的播放效果。

在本論文中我們結合 server push 和 proxy server，設計並實作一個支援 RTSP proxy 功能的 Web 節目訂閱系統，以期能解決上面的問題。透過這個系統，Web 資訊的提供者能方便地將自己的站台資料以節目方式提供給使用者訂閱，使用者訂閱這些網頁節目資料後，系統會進行抓取與自動更新的工作。使用者瀏覽網頁內容時，不需花費時間在等待網路資料傳輸延遲上。在播放多媒體資料時，經由系統的 RTSP proxy 功能，可避免因為頻寬不足影響到即時播放多媒體的品質。

本系統提供一套有效架構及工具供資訊提供者在 Internet 上提供常態姓具語音功能之節目，也方便資訊讀者訂閱瀏覽。在本論文的其他部份裏，第二節中我們將介紹一些相關方面的研究，第三節會對系統架構及實作考量作詳細的說明，第四節為測試數據，最後第五節是結論。

2. 相關研究

本論文的目的在於研究設計一個架構，在 Web 環境建構一個利用 Server Push 和 Proxy 技術的節目訂閱系統，以下列舉 Microsoft Internet Explorer、Netscape Netcaster 和 PointCast Network 等伺服器推送的相關研究，說明其運作模式及其不足之處。

2.1 Microsoft Internet Explorer 4.0 的 Webcast 技術[5]

Explorer 4.0 對每個可供訂閱的節目稱之為頻道(Channel)，每個頻道作者必須為它製作一個 CDF（Channel Definition Format）檔，將這個檔案放在 Web Server 上，CDF 是 Microsoft 所制訂的檔案格式用來描述頻道，包括頻道所含的 URL、頻道的名稱、頻道的更新排程等等資訊，讓 IE 獲知頻道的資訊。當使用者使用 IE 點選一個 CDF 檔的鍊結(link)表示使用者要訂閱此 Channel，IE 會下載此 CDF 檔，依據設定的排程不斷的下載此頻道的 CDF 檔，以及下載有被更動過的網頁，讓新的資訊不必透過使用者的操作就已經下載到使用者的電腦上，達成伺服器推送的效果。

此種方式，只有使用 IE 的使用者才能利用，使用其他瀏覽器的使用者就無法使用。而且當使用者的電腦沒有開機或連上網路，就沒有辦法達到伺服器推送的效果。IE 是利用不斷的下載 CDF 檔和下載 Channel 的網頁來達到伺服器推送的效果，並不是真正的伺服器推送。

2.2 Netscape 的 Push 技術[6]

Netscape 新成員 Netcaster 也是利用 Channel 的觀念，擴充原本的 JavaScript 功能，替其添加了 "netcaster" 元件及 "channel"物件。Channel 的作者在訂閱的網頁裡需用 JavaScript 寫著此 Channel 所具備的設定。當使用者讀取此網頁時表示要訂閱此 Channel，netcaster 根據 script 描述的位置、時間去下載 Channel 的內容。

Netscape 的作法同樣也只有使用 Netscape 的使用者才能利用，使用者電腦沒有開機或連上網路就無法運作。而且 Netcaster 和 Navigator 的 Cache 是分開的沒有共享，可能會造成磁碟空間浪費或發生兩者 cache 不一致的情形。

2.3 PointCast Network

PointCast Network 是目前網路上最受歡迎的新聞推送(push)軟體，使用者安裝了這個軟體，裡面會列出網路上支援 PointCast 的 Channel，讓使用者挑選訂閱。新的資料會下載到使用者的電腦上，利用 PointCast 本身的瀏覽器或其他如 Netscape navigator 和 IE 來展現。要使用 PointCast Network 使用者必須安裝 PointCast 的 client，學會如何去操作各種設定，對於使用者而言會增加額外的負擔。而且當使用者沒有開機上網路時，新的資訊沒有辦法下載到使用者的電腦上。

2.4 DRP (The HTTP Distribution and Replication Protocol)

DRP[3]設計來有效的複製 Web Server 上的一個階層式架構的檔案集合到許多的 client。DRP client 和 server 的溝通是在 HTTP 上增加一些 header 來達成。使用 DRP 的 client 可以只下載有變更過的檔案，甚至只下載和原有檔案差異的部分。DRP 的目的主要在於改善由 HTTP 來分散資料的效率和可靠性。本論文中的節目訂閱系統，需要複製傳播節目內容，利用 DRP 協定來傳輸節目內容可以讓資訊傳播更有效率。

2.5 Squid Proxy Server

Squid[12] 提供高效率的 Proxy Cache，它支援 FTP、Gopher 和 HTTP 的 Request，是目前網路上使用最廣泛的 Proxy Server。Squid Cache 的特色在於各個 Squid Cache 之間是可以互相合作的，構成一階層式(hierarchy)的架構，利用這種架構，可以更有效的取得資訊，減少使用者的等待時間以及降低網路頻寬的使用。

在我們的 Web 節目訂閱系統中，需要以 proxy server 來 cache 被使用者訂閱的節目內容。由於 squid 不能提供將特定網頁鎖定在 cache 中的功能，且不支援對 RTSP 資料的 cache，因此我們另行實作 proxy server 來 cache 被訂閱的網頁資料與 RTSP 資料，但是對於其他的網頁資料，我們是在後端搭配 squid proxy server 來達到 cache 的功能。

3. 系統介紹

3.1 系統架構

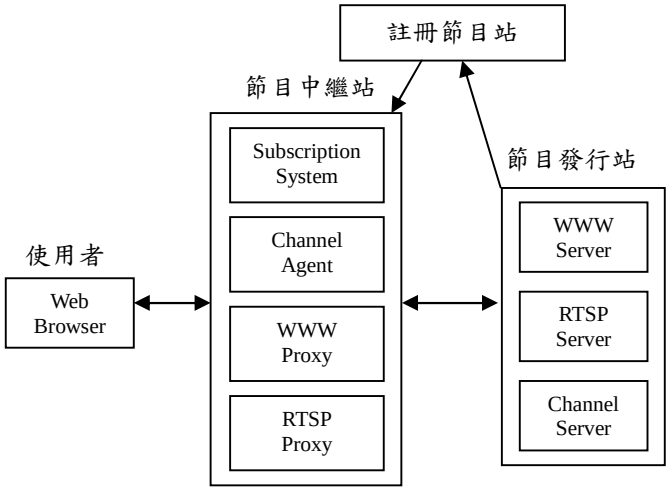

圖 3-1:系統架構

整個系統架構如圖 3-1 所示，可以分為四個角色：使用者、中繼站、節目發行站和節目註冊伺服器。各個角色所具備的元件與功能說明如下：

1.使用者

使用者只要具備一般的瀏覽器，不必在加裝任何軟體就可以使用此訂閱系統。使用者在使用這個系統時必須將其瀏覽器的 Proxy Server 設成中繼站上的 Proxy Server，所以使用者的要求會送給中繼站的 Proxy Server。而使用者使用訂閱系統的方式是利用瀏覽器連上中繼站的訂閱系統，中繼站會提供選單讓使用者點選。

目前各種瀏覽器尚未支援 RTSP，所以必須提供 RTSP client 給使用者來播放 RTSP 的節目內容。

RTSP client 也可以設定 proxy server 為中繼站的 RTSP proxy server。

2.節目中繼站

中繼站包括了三個元件：訂閱系統、Channel Agent 和 Proxy Server。訂閱系統提供使用介面給使用者來訂閱或取消訂閱節目，所以必須具備編排節目選單的功能和儲存使用者訂閱狀況的相關資料，為了讓使用者便於操作，訂閱系統會建立在 Web server 上，使用者只要透過一般的瀏覽器就可以使用。

為了讓使用者能夠很方便的使用此系統，中繼站所提供的節目選單必須很豐富，所以中繼站要能夠蒐集節目資訊。使用者訂閱節目的動作是向中繼站來訂閱或者取消訂閱，中繼站必須能替使用者向節目發行站訂閱和取消訂閱。而節目內容有更動的時候，中繼站必須能更新 proxy cache 的節目內容。為了以上這些目的在中繼站此系統會提供 Channel Agent 這個元件負責這些工作。

中繼站的 proxy server 是做為節目內容的 cache，使用者的瀏覽器透過中繼站的 proxy 來取得資訊，可以避免網路傳輸延遲所造成的長時間等待。這個 proxy server 主要分為兩個部份，一個用來處理被訂閱的節目中一般性的網頁內容，其後端搭配 squid 以處理非訂閱網頁的存取。另一個是 RTSP proxy，目前此 RTSP 僅支援語音資料的處理，它用來處理節目中以 RTSP 方式製作的語音多媒體資料

現在使用者連上網際網路的方式不外乎在用專線直接連上，或利用電話撥接網路透過 ISP 公司連上網際網路以及經由有線電視網路(CATV)，透過頭端(head end)來連接。電話撥接的 ISP 公司或 CATV 的頭端都很適合提供中繼站的服務，因為使用者的要求都會透過這些公司傳出去，在這些點設立中繼站可以很有效的發揮中繼站的功能。

3.節目發行站

節目發行站的元件包括 Web server、RTSP server 和 Channel server。節目提供者要將其 Web server 上原本所提供的資訊變成本系統中的節目不需改變其架構，只須為這個節目編寫一節目單。若有 RTSP 的資訊則需放在 RTSP server 上。

節目發行站必須讓中繼站訂閱和取消訂閱，所以節目發行站必須有一套機制能夠處理訂閱的工作，將有訂閱的中繼站的 host name 記錄下來，以便在節目有更新時能夠更新這些中繼站所 cache 的節目內容。這些工作將由 Channel Server 這個元件來負責。

4.節目註冊站

在我們所設計的架構下中繼站必須具有蒐集節目資訊的功能，但是網路上每個人都可以架設網站提供節目，要中繼站主動在網路上蒐集節目資訊需要花費很大的代價，可能需一 Robot 在網路上漫遊分析是否某個網站是此系統的節目發行站，在將這些網站整理成選單讓使用者挑選，如此不但浪費網路頻寬而且結果不一定準確。在我們的系統裡將設計一節目註冊站，讓節目發行站的管理者在此註冊其節目資訊，而個中繼站在到此 well-known 的節目註冊站抓取節目資訊。

綜合以上所述，節目發行站是位在各個網頁資料提供者的 Web server 上，而節目中繼站則是位在 user 所在的區域網路中，節目註冊站為一 well known server，提供節目發行站與節目註冊站一個連繫的管道。

3.2 實作上的一些問題考量

1.節目單的格式

在 Web 上文件是以 URL 來描述，同一個 Web Server 上可能有很多種資料彼此並不相關，為了

使中繼站和節目發行站能夠溝通一個節目包括哪些 URL，所以對每個節目需有一節目單，節目單中需描述節目內容的 URL，以及其他有關的資訊比如說節目的類型或節目的簡介。目前並沒有標準的格式，Microsoft 為其 push 技術訂定 CDF(Channel Definition Format)[4]。在我們的系統中將引用此 CDF 格式做為節目單的格式。

2.節目內容的傳遞

在我們的系統裡需要將節目內容從節目發行站傳遞到中繼站上，因為並不是要直接傳給使用者所以並不用限定利用 HTTP 來傳送。HTTP 在傳遞資料上效率較差，可以改用 FTP 或 DRP 等協定來傳送資料。考慮利用 HTTP 可以和現在的 WWW 環境較為相容，目前我們採用 HTTP 協定來傳送資料。

3.達成伺服器推送的方式

在我們的系統中，對節目發行站而言，節目中繼站便是其 client，我們在中繼站上執行一個叫 channel agent 的 process，當節目發行站有資料要推送到中繼站時，事實上是通知中繼站上的 channel agent，channel agent 再以 Web client 的身份向節目發行站取得資料。

4.避免節目內容被 cache flush 的方法

利用 proxy 來 cache 節目內容必須考慮 proxy cache 的 replacement，當 proxy cache 容量超過時可能將節目從 cache 中移除，如此便不能讓使用者得到便利，所以必須讓 proxy 具有 lock 特定 page 的功能。我們的作法是實作另一個 proxy server 來 cache 這些被訂閱的網頁，而一般性的網頁則是交由後端 squid proxy server 處理。

5.RTSP proxy server 的實現

由於目前的 proxy server 均不支援 RTSP，使得 RTSP client 端在播放多媒體資料時需要連回原資料所在站台。為避免當資料站台與 client 距離過遠或是兩者間頻寬不足時可能導致不良的播放效果, 在我們的系統中提供了對 RTSP proxy 的支援。此 RTSP proxy server 主要是由一個 RTSPget (RTSP client)，RTSP server module 和一個 cache manager 所組合而成.

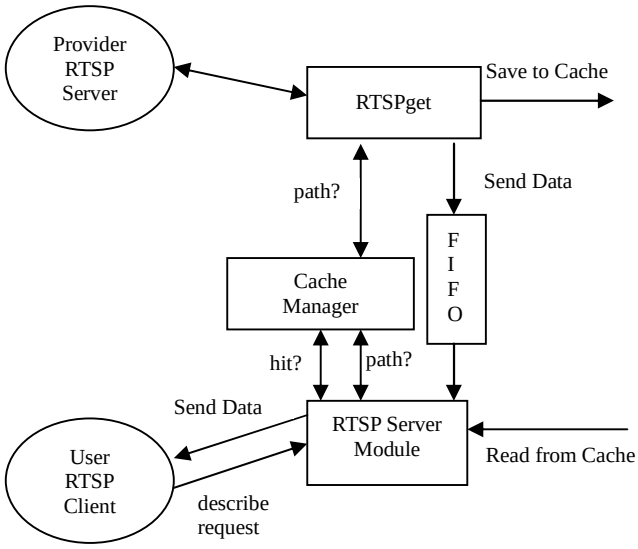

圖 3-2 RTSP proxy server 運作圖

如圖 3-2, RTSP proxy 在接受 user RTSP client 的 describe request 後，RTSP Server module 會詢問 Cache manager 是否這個 request URL 已經存在於 cache 中。如果存在 cache 中(hit)，則 Cache manager 將告知 RTSP Server module 這個 request URL 在 cache 中的存放路徑。接下來的運作 RTSP server module 會把相對於這個 request URL 的資料傳回給 user RTSP client。

如果 request URL 並不存在於 cache 中，此時 RTSP Server module 便會呼叫 RTSPget， 請 RTSPget 抓取 request URL。

RTSPget 被呼叫之後，會以 Client 的角色向 Providee RTSP Server(也就是節目提供者的 RTSP server)要求 request URL。並開啟 FIFO 將所有收到的訊息都傳給 RTSP Server module。RTSP Server module 再原封不動地傳回給 user

RTSP client，因此 user RTSP client 感覺起來便會像直接向 Provider RTSP Server 發出要求一樣。而且，user RTSP client 不必等到 RTSPget 將所有的資料都儲存在 cache 中後才能獲得 RTSP Server 的回應。

同時，RTSPget 會詢問 cache manager 相對應於 request URL 的 cache 路徑，然後將 request URL 的資料儲存在指定的路徑中。當儲存完畢後，RTSPget 會再通知 Cache manager，將 request URL 所對映的 hash key 插入 cache 的 hash table 中，以供查詢。

6. RTSP proxy 與系統的整合

RTSP 服務與現有 Web Server 之間的結合，需要靠一種副檔名為 rtsp 的檔案。比如說使用者在某個 Web Page 上看到一個指向 http://www_server/wave1.rtsp 的鏈結。當使用者點選了這個鏈結，Browser 便向 Web server 要求這個 URL 的內容。而 wave1.rtsp 的內容只有一行：rtsp://rtsp_server/wave1.wav 因為 Browser 中 MIME 設定對 *.rtsp 的處理是呼叫外部程式 RTSP Player，所以 wave1.rtsp 便會交由 RTSP Player 處理。RTSP player 便會開始播放 rtsp://rtsp_server/wave1.wav 這個 URL。

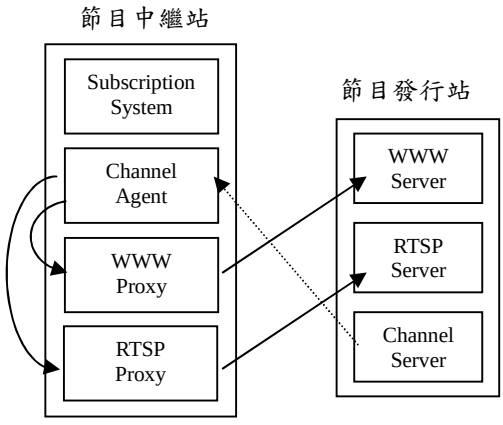

圖 3-3 Channel Agent 透過不同 proxy 抓資料

當節目發行站的內容有更新時，網頁節目製作者透過 Channel Server 通知所有曾經訂閱此節目的 Channel Agent。各中繼站的 Channel Agent 事實上是扮演一虛擬的使用者，瀏覽節目內容，它會分辨 URL 是屬於 HTTP 或是 RTSP。若屬於 HTTP，則向 WWW Proxy 發出請求；若屬於 RTSP，則向 RTSP Proxy 發出請求。Proxy Server 收到 Channel Agent 的請求之後，便會將 Channel Agent 要求的內容存放到 Cache 當中以供使用者讀取。差別只在於使用的 protocol 不同。

4. 測試及評估

本實驗以 RTSP client 透過系統中的 RTSP proxy server 讀取一個遠端的 RTSP Server 上的 audio file，計算 Proxy Server 可處理的 RTSP client 個數上限以及比較在有 Proxy Server 和沒有 Proxy Server 的條件下，client 端開始聽到聲音的時間。實驗用的機器如下：

RTSP Server：P133，64MB RAM，SCSI Disk，100Mb/s Ethernet，IP： 140.116.25.10，成大生物系，OS：FreeBSD Unix。

RTSP Proxy Server：P200mmx，32MB RAM，IDE Disk，100Mb/s Ethernet，IP：140.116.72.124，成大電機系，OS：FreeBSD Unix。

RTSP Client：P200，128MB RAM，SCSI Disk，10Mb/s Ethernet，IP：140.116.72.72，成大電機系，OS：FreeBSD Unix。

4.1 最大 Client 數

測試方式是啟動數個 RTSP Client 向 RTSP Proxy Server 要求同一個 audio file，同時以一個 windows 95 上的 RTSP Client 程式發出同樣的要求，測試播放效果是否順暢。檔案大小為 754642

bytes，播放長度為 34.22 秒，計算可得所需的最小頻寬為 176400 bits/s。所有的 RTSP Clients 都位於 10Mb/s Ethernet 網路上，因此在理論上，應可以同時支援 53 個 RTSP Client。測試結果顯示，最大 Client 數可達 50 個。

4.2 對傳輸速度的改善

對一個 Proxy Server 而言，最重要的就是使用者感覺到的 response time。以 Proxy Server 扮演的角色來說，因為它實際距離通常會比真正的 Server 來得近,所以只要 data 被 cache 在 disk 裡，一般來說,response time 一定都會比沒有用 Proxy Server 來得少。

首先，我們以 ping 指令向 RTSP Server 及 RTSP Proxy Server 發出 request 各 200 次得出平均回應時間。數據如下：

	Ping 回應平均時間
Client to RTSP Server	5.78 ms
RTSP Proxy Server to RTSP Server	3.40 ms
Client to RTSP Proxy Server	0.97 ms

接著我們執行 RTSP Client 模擬程式，這個在 Unix 系統上的模擬程式不會發出聲音。它的動作是，向 Server 發出 request，開始計時，直到 Client 端收到第一筆 audio data 為止，這中間所花的時間視為 RTSP Server 對 Client 的 response time。以下是得到的實驗數據：

	平均回應時間
沒有透過 Proxy	631.20 ms
透過 Proxy 且資料在 cache 中	599.72 ms
透過 Proxy 但資料沒有在 cache 中	808.87 ms

Cache hit 時減少了 0.03 sec（4.98% ）的回應時間。但在 cache miss 時，多出的 overhead 卻有 0.2 sec（28.14% ）。這樣的結果原因在於測試中的 RTSP server 距離 client 很近，使得網路速度的影響遠不及程式處理速度，但如果 RTSP Server 位在遠端網路上，那麼 Proxy Server 的 overhead 將會相對地少。

4-3 系統未來之改進及擴充方向

目前本雛形系統已能達到方便節目發行、讓使用者易於訂閱之目的, 並透過將資料 push 到中繼站 proxy server 以減少網路延遲, 降低頻寬浪費. 但仍有著待改進的地方:

1. Proxy 效能的改進
節目中繼站的 proxy 只 cache 節目內容，非節目內容的要求會交由後端的 squid proxy 處理而造成延遲。將網頁鎖定功能整合進 squid proxy 應是較具效率的方法。

2. Multicast 的運用
在此系統裡同一份節目內容傳到各個中繼站，需要傳輸多次才能完成，假如能運用多點傳輸技術(multicast)，讓同一份內容只需傳遞一次，如此可以減少網路頻寬的浪費，也可以降低節目發行站的負載。

3. 階層式架構的設計
考慮各個中繼站互相知道彼此的位置，當中繼站接受使用者訂閱時，判斷與節目提供者的位置關係，若不適合直接訂閱，則向其他的中繼站選一個距離節目提供者最近的中繼站，要求代理訂閱。如此可以減少 Time Out 的機會，且可有效減少網路的交通量。

5. 結論

雖然目前有許多伺服器推送的軟體,可以將 Web Server 上的資料傳送給使用者，但是通常必須配合特定的 Browser，或使用者端需加裝其他的軟

體。將資訊推送到使用者的電腦上雖然可以節省使用者的時間，但是大部分的使用者並沒有 24 小時的把機器開機上網。使用者所感興趣的資訊很多，每個更新的時間又不統一，使用者若沒有一直開機等待會漏失很多資訊。而且由使用者直接去 Web Server 下載資訊會浪費網路頻寬，若透過 proxy 又有可能會得到過時的資料。

在本論文中我們結合 Server Push 和 Proxy 技術建立 Web 節目訂閱系統，此系統提供網頁製作人一個方便的資訊發行管道，而使用者也能夠方便地以訂閱的方式來取得所要的資訊。這樣的系統尤其適用在教學節目或雜誌等這些資料量較大的資訊傳播上。本論文所提出的 Web 節目訂閱系統具有以下特點：

1. 不必更動現有的 Web server, Web Browser 與 proxy server 運作.
2. 透過節目註冊站與節目中繼站的配合, 提供網頁製作者方便且具效率的資訊發行管道
3. 節目資料被 push 到各中繼站的 proxy server 上, 而非直接到使用者電腦. 使用者不必 24 小時開機上線等待新的資訊, 而且不同的使用者可以共享同一個中繼站上的節目資料
4. 支援對 RTSP 的 proxy 功能, 使用者在撥放 RTSP 資料時不必連回資料提供站台, 可以降低網路延遲時間, 減少網路的交通量, 同時獲得較佳之播放品質.
5. 以獨立的 proxy server 存放被訂閱的節目資料, 避免訂閱節目資料被其他網頁給擠出 cache 的情形.

參考文獻

[1] R. Fielding，J. Gettys，Mogul，H.Frystyk，and T.Berners-Lee，"Hypertext transfer protocol -HTTP/1.1," RFC 2068，Jan 1997

[2] D.Kristol and L. Montulli，"HTTP state management mechanism," RFC 2109，Feb.1997.

[3] "The HTTP Distribution and Replication Protocol"，http://www.w3.org/TR/NOTE-drp-19970825.html

[4] "Channel Definition Format"，http://www.w3.org/TR/NOTE-CDFsubmit.html

[5] "Webcasting in Microsoft Internet Explorer 4.0 White Paper"，http://www.microsoft.com/ie/press/whitepaper/pushwp.htm

[6] "Netcaste r Developer's Guide"，http://developer.netscape.com/library/documentation/netcast/devguide/

[7] Drummond Reed，Kevin Jones，"Pushing Push : Advancing the Features of Channel Communications"，Intermind Corporation，September 8，1997.

[8] H. Schulzrinne，A. Roa and R. Lanphier，"Real Time Streaming Protocol (RTSP)"，RFC 2326，Apr 1998.

[9] M. Handley，"SDP: Session description protocol," RFC 2327，Apr 1998.

[10] H.Schulzrinne，"RTP profile for audio and video conferences with minimal control," RFC 1890，Jan 1996.

[11] H. Schulzrinne，S. Casner，R. Frederick，and V. Jacobson，"RTP: a transport protocol for real-time applications"，RFC1889，Internet Enginnering Task Force， January 1996.

[12] http://squid.nlanr.net/Squid/，"Squid Internet Object Cache"