行動式多媒體推播與跨終端機換手系統設計實作

Design and Implementation of a Multimedia Push and Inter-device Handoff System for Mobile Devices[1]

蔡奇峰 史永健 曾建超 何文楨

Chi-Feng Tsai[+] Yung-Chien Shih[‡] Chien-Chao Tseng[‡] Wen-Jen Ho[±]

chi@iis.sinica.edu.tw {ycshih, cctseng}@csie.nctu.edu.tw wenjen@iii.org.tw

Institute of Information Science, Academia Sinica[+]

Department of Computer Science and Information Engineering, National Chiao-Tung University[‡]

Institute for Information Industry, Multimedia Technologies Laboratory[±]

摘要

在這一篇論文裡，我們設計並實做一個多媒體遠端學習系統讓觀看者和演講者都可以在遠端參與教學活動。觀看者在這個系統架構之下除了可以看到目前正在進行的演講、教學之外，也可以看一些預先錄製好的教學節目。此外，這個系統也支援多媒體串流的跨終端機換手，讓觀看者可以很方便地將多媒體 session 在不同設備間轉換。而且在進行跨終端機換手的時候，系統可以根據使用者的設備特性來調整多媒體串流的 bit-rates。最後，我們也提出一個新的方法將即時的教學投影片影像傳送給觀看者。使用這些經過設計的方法來傳送投影片影像時，不用消耗太多頻寬就可以讓觀看者看到清晰的投影片影像。

ABSTRACT

In this paper, we present the design and implement a multimedia distance learning system that could enable both the audience and speakers to partiate in learning or teaching activities from the distance. In addition to the teaching or speech programs that have previously been recorded, the audience in this system could but could attend the on-going programs that are currently taking place on the networks. Bedises, the presented system also support inter-device multi-stream hand-off for an audience to transfer multimedia sections from a device previously used to a new devices in hand. Furthermore, the inter-device function can also adjust the bit-rates of the multimedia streaming according to the characteristics of the devices in use while performing an inter-device handoff. Finally, we also propose a novel approach to delivering real-time teaching contents to the audience. The approach has been carefully designed so that the contents delivered are clear to the audience without consuming much bandwidth.

關鍵詞：Remote presentation, Distance learning, Multimedia streaming, Inter-device handoff

一、研究動機

近幾年來，由於客戶端的連線技術不斷地進步，使用者連上網際網路的速度也不斷提升。目前透過 ADSL、Cable modem 等等方式，下行連線速率至少可達 512 KBits 以上，上行速率也可達 64 KBits 以上。對於一般使用者來說，下行速率的提升，不僅只是節省上網的時間，還可以獲得以前無法完成的各種服務，而這些正是需要較高頻寬的多媒體服務，例如網路電視、網路電影、網路電台...等等。但是這類多媒體影音的資料量相對於一般網頁來講還是大得很多，光有寬頻還無法滿足使用者的需求，因為我們不可能等到所有影音資料下載到電腦之後才開始播放，所以就必須透過串流技術（streaming technology）讓使用者可以一邊觀賞節目，一邊進行下載，以縮短等待的時

[1] This research was supported by the National Science Council, ROC, under grants NSC 92-2219-E-009-010- and NSC 92-2219-E-009-011-.

間。

在這裡，我們希望可以將這些網路與多媒體串流的技術應用在遠端學習上，不過「遠端」這個部分通常著重在於觀看者可以在遠端透過電腦設備與網路連上伺服器觀看學習資料。我們覺得這樣是不太夠的，因為演講者仍然必須到一個特定的地方（例如會議室、教室...等等）才能開始進行簡報。如果演講者也可以利用這些技術，在遠端就開始進行簡報，那就增加了更多的方便性了。

另一方面，一個使用者可能同時擁有桌上型電腦與筆記型電腦，上網的地方可能是在電腦桌，也可能是在任何一個地方透過無線網路（wireless network）與網際網路接駁。使用者這一分鐘可能在桌子前使用電腦，但下一分鐘可能帶著筆記型電腦去聽演講。使用者如果在觀看多媒體串流的時候，想要換到另外一個設備，可能必須再手動進行一次連線以便取得剛剛觀看的串流，而且還必須自行控制播放串流的應用程式，讓應用程式從剛剛中斷的地方開始播放，這樣對於使用者來講實在不太方便。

使用者上網不再局限於固定的地方，固定的設備。這樣的使用習慣和以往是不同的，而新的技術與環境可以造就新的應用。例如個人化的多媒體服務〔5〕，透過無線網路或者行動電話允許個人遠端遙控家庭設備、攝影機，預約錄製節目，並於任何時間可以讓使用者在遠端觀看錄製下來的節目；以及一個分散式的會議系統〔6〕，讓使用者可以從遠端參與會議，系統並可以錄製會議過程以便讓無法參加的人可以在事後觀看完整的內容。除了這些之外，相信還有很多實驗性或者已經是實用階段的各種應用。這些應用的共同特點都是讓人可以在遠端參與事務，並利用網路與多媒體技術增加臨場感，盡量接近使用者在現場的感覺。

有了這些技術以及應用例子，我們在這篇論文中也將嘗試提出一個可以改進前述不足之處的系統架構。在這個架構之下不僅是觀看者可以在遠端直接觀看伺服器上的多媒體串流，演講者也可以在遠端就進行簡報教學。並且觀看者在觀看串流的時候可以很方便地進行跨終端機的換手動作（inter-device handoff）。

二、系統架構簡介

(一) 系統架構

在我們構想的這個架構之下有三個不同的設備，分別是 Media server、Remote pusher、Client player（如圖 1），各簡稱為 MServer、RPusher、CPlayer，以下文章我們也都使用這些簡稱。

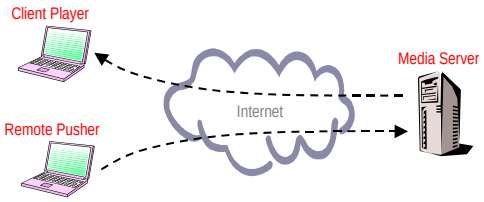

圖 1 系統架構

這三個設備各自要負責的工作如下：

- MServer

  可以解析事先錄製好的多媒體檔案，並將多媒體檔案轉化成串流經由網路提供 CPlayer 觀看。在遠端學習的需求上，這些事先錄製好的節目可以是一些教學影片。

  接受 RPusher 在遠端推播來的多媒體簡報串流，經過適當的緩衝（buffering）之後，可以提供給 CPlayer。那這部分就是動態產生的即時簡報節目。

- RPusher

  提供給演講者在遠端進行簡報教學。RPusher 會將演講者的簡報畫面與聲音推播到 MServer。

- CPlayer

  觀看者可以透過 CPlayer 觀看 MServer 送來的多媒體串流，包含了事先錄製好的多媒體檔案，或者目前正在進行的多媒體簡報。觀看者在觀看串流的時候，可以在各個 CPlayer 之間切換，也就是進行跨終端機換手。

在 RPusher 與 CPlayer 之間存在一個 MServer 有兩項好處：

- RPusher 不需要負責把多媒體簡報串流傳送給所有連上線的 CPlayer，只要推播到 MServer 即可。這樣 RPusher 不需要考慮大量 CPlayer 連線的問題。因此，只要簡單的機器就可以進行簡報。
- CPlayer 不需要知道 RPusher 在什麼位址，只要知道 MServer 的位址即可。這樣一來，就算 RPusher 處於移動的狀態，甚至在簡報過程中更動本身的網路位址都無所謂。

(二) 使用行為

這裡我們要以三種不同的使用行為為例子，說明這個推播簡報系統的應用。首先，假設我們正在舉行一場研討會，會中的一位演講者因為臨時有事無法前來會場，但因事發突然

無法事先通知與會人員，現場都在等待即將開始的演講。為了要讓研討會可以繼續，我們請這位講師在遠端以電腦連上推播系統的伺服器（MServer），然後使用 Microsoft PowerPoint 簡報軟體開始簡報。講師使用的這一台電腦稱之為遠端推播者（RPusher）。在會場這一端，我們準備一台電腦（CPlayer）也連上 MServer，然後開始接收由 MServer 送過來的多媒體簡報串流，並將螢幕投影到會場螢幕，以及將聲音接到會場擴大機由喇叭播音。另外一些無法及時趕到會場的觀眾，也可以在遠端透過 CPlayer 連上 MServer 同時觀看這場簡報教學。則不論是演講者或者觀看者，都可以在任何地方共同參與一場簡報教學活動。整個應用如圖 2 所示。

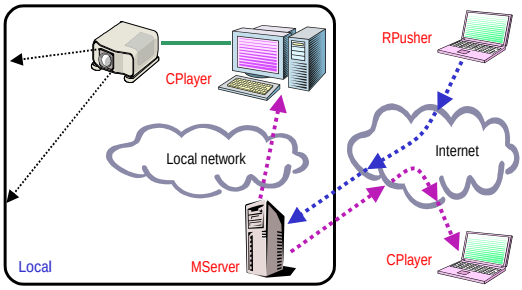

圖 2 應用範例一

在這個例子中，講師那一端不用到特定的地方也不必準備特別的設備，只需要一台電腦、Microsoft PowerPoint 簡報軟體與一支麥克風就可以進行簡報。甚至，透過其他技術的配合，例如 MobileIP〔4〕，講師在簡報的同時可以是持續在移動的狀態。

第二個例子假設在學校的實驗室中，實驗室的指導教授出國但仍希望可以觀看實驗室的研究生報告。連線的模式與範例一幾乎相同，差別只在於接上投影機的是要進行簡報的研究生所使用的電腦(也就是 Rpusher)。在研究生向現場聽眾簡報的同時，簡報畫面與聲音也一併推播到 MServer。指導教授只要使用 CPlayer 即可連向 MServer，同步觀看簡報，同時可以聽得到現場聲音。當然，如果這時有其他實驗室的同學因故無法到達現場聽報告，他們也同樣可以利用 CPlayer 在遠端連上 MServer，與大家一同觀看簡報。

MServer 除了可以動態提供即時的簡報影音串流之外，也可以提供事先錄製好的影片。若使用者要求 MServer 播放預先錄製好的影片的時候，MServer 可以因應終端機的形式透過動態減少或增加影像的 bit-rate，讓計算能力較差的機器仍然可以順暢地觀看影像，不至於因為要影片資料解壓縮使得播放延遲，甚至無法正常播放影像。當使用者換成較好的機器時，MServer 自動恢復較高品質的影像傳輸。

第三個例子便是這樣的使用情況。使用者原本在實驗室中使用個人電腦直接透過乙太網路（Ethernet）連上網際網路，這個時候使用者是用較高的頻寬與運算能力較好的機器。觀看到一半時，使用者有事必須離開座位，但使用者並不想中斷影片。因此使用者帶著一台比較差的筆記型電腦，透過無線網路連上 MServer 並執行跨終端機換手（inter-device handoff）動作，在換手的同時，CPlayer 也一併將機器形式通知 MServer，MServer 自動減少影像的資料量。這時，使用者仍然可以順利地在這台筆記型電腦上觀看，但得到的影像品質會比較差。並且在進行換手的時候，使用者並不需要記憶剛剛看的是那個影片，看到哪裡…等資訊，使用者只要在新的 CPlayer 上再次登入 MServer，並選擇將先前觀看的串流轉換到新的 CPlayer 上即可，MServer 與 CPlayer 會負責後續工作，完成換手動作。

三、系統實作

(一) RPusher

(1) 實作架構

RPusher是給演講者在遠端進行演講時使用的。RPusher會取得PowerPoint的事件，並且透過DirectX〔2〕取得螢幕上的像素[2]（pixel）以及麥克風的聲音。這些媒體資料會被推播到MServer上。在RPusher上我們需要執行一個程式，也叫做RPusher。程式的架構如圖 3 所示。

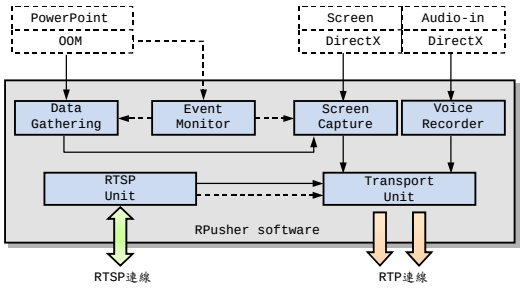

圖 3 RPusher 方塊圖

圖中 RPusher software 是我們的程式，OOM 是 Office Object Models。實線箭號表示資料傳遞方向，虛線箭號則表示訊息通知。程式方塊圖說明如下：

- RTSP unit：負責與 MServer 建立 RTSP session。建立 session 的過程當中，會將運送多媒體資料的底層傳輸埠設定傳遞給 Transport unit。包括 MServer 預備在哪些傳輸埠接收多媒體資料，使用 UDP 或者 TCP，以及使用 RTP 等等。

[2] 顯示在螢幕上的一個點。

此外，也會通知 Transport unit 是否開始傳送資料以及是否暫停或終止傳送資料。

- Transport unit：負責接收媒體擷取單元（Screen capture、Voice capture）取得的多媒體資料並且將這些資料傳送給 MServer。
- Screen capture：利用 DirectX 提供的介面取得螢幕上的像素然後加以處理，將需要傳送給 MServer 的部分轉遞給 Transport unit。至於哪些是需要傳送的部分，會在下一節中詳細說明。
- Voice recorder：透過 DirectX 錄製麥克風的聲音，並將之轉遞給 Transport unit。
- Event monitor：過濾 PowerPoint 所產生的事件，如果發生「投影片換頁」或者「投影片動畫」的事件，就會通知 Data gathering 取得投影片上的資訊，以及通知 Screen capture 擷取螢幕上的像素並處理像素。
- Data gathering：會取得目前顯示在螢幕上的投影片上的文字、圖片的位置與大小，這些資訊可以幫助 Screen capture 判斷螢幕上哪些資料是需要傳送給 MServer 的。

(2) Screen capture 機制

當演講者使用 PowerPoint 進行簡報的時候，畫面的變化有一些特性，這些特性可以用來減輕 CPU 在擷取畫面時的負擔並且減少傳送出去的影像資料量。

演講者使用 PowerPoint 進行簡報教學的時候，PowerPoint 會將投影片一頁一頁地顯示在螢幕上。當投影片換頁的時候，螢幕上並非所有的像素都會發生變化。我們只要找出變化的像素，而且只傳送發生變化的資料給 MServer。此外，通常演講進行的時候，顯示投影片的螢幕並非一直都在變化，通常是一個投影片會停留在螢幕上一段時沒有任何變化。如果使用傳統的 screen capture 方式（也就是不斷地擷取螢幕）會浪費 CPU 的處理時間。因此我們希望只有在顯示投影片的螢幕發生變化的時候才去檢查螢幕上哪些地方確實有變動。

在這篇論文中，RPusher 程式會等待 PowerPoint發出事件通知。目前最在意的有兩個事件，分別是「投影片換頁」或者「投影片動畫」[3]。當演講者讓投影片換頁之後，PowerPoint就會發出「投影片換頁」事件，而當一張投影片中有一個動畫發生過後，則會發出「投影片動畫」事件。這兩個事件都代表了螢幕上的像素發生了變化。所以我們會在這兩個事件發生時，執行Screen capture動作以及比較螢幕像素變動的情形。

如果我們以像素為單位來比較差異，並且傳送變化的像素，這樣會造成較多的資料量以及增加 CPU 的負擔。因此我們要先將播放投影片的螢幕分成許多區塊，然後以此區塊為單位來比較一個區塊是否有變化，若有變化則傳送這個區塊。

當演講者以 PowerPoint 開啟一份投影片之後，每一頁投影片上的文字、圖形都會有一個區域範圍（如圖 4）。程式可以透過 OOM 取得每一頁投影片裡頭的所有文字、圖形區塊的範圍。我們將這些範圍執行聯集運算（如圖 5），便會形成一些區域，而這些區域就是在簡報的過程中就是可能發生變化的區域，我們稱之為 Possible Variable Area（簡稱 PVA），如圖 5 右下角方塊中的深灰色區域。

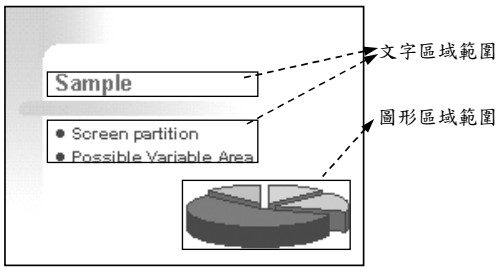

圖 4 投影片上的文字與圖形區域

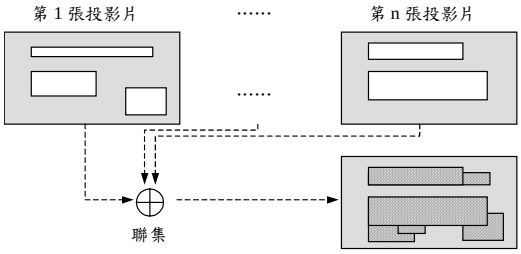

圖 5 計算出 Possible Variable Area

程式就以此 PVA 為基準，將螢幕切成多個區塊，如圖 6 之中，灰色區域為 PVA，白色區域則是預測可能不會變動的區域。也就是說，當演講者在進行簡報教學的時候，播放投影片的過程中只有 PVA 中的像素可能產生變動。

[3] 在這裡，這兩個事件的名稱是我們自行命名的，以方便說明。在OOM之中，每個事件都只是一個數值而已。

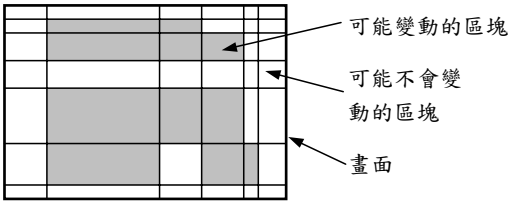

圖 6 以 PVA 為基準將螢幕切割成多個區塊

由於以 PVA 切割出來的區塊可能會太大，因此我們要把這些區塊再切割成較小的區塊。但是區塊若切得太小，則區塊太多，每個區塊傳送的時候都需要封包標頭（header），會有較多的 overhead。區塊若切得太大，則比較差異的時候就不夠精確，可能一個區塊中變化的像素只佔有少部分而大部分像素都是不變的，卻仍然要傳送此區塊。因此，我們以系統上的 MTU（Maximum transmission unit）為上限，把區塊的像素壓縮成 JPEG 格式後，其資料量與必要的封包 headers 加起來盡量小於 MTU。

由於投影片換頁時，並非螢幕上所有區塊都發生變化，所以當投影片由第 i 張換成第 j 張的時候，程式可以把這兩張投影片上所有的文字與圖形的範圍「聯集」起來，我們稱這個區域為「變動區域」。程式只需要比較與「變動區域」重疊的區塊即可。

此外，發生的事件若是「投影片動畫」的時候，螢幕上其實沒有更換投影片，只是某個動畫顯示在螢幕上而已。但這不影響演算法的正確性。因為仍然可以計算出「變動區域」，在比較與「變動區域」重疊的區塊時，只有動畫所在位置的像素會產生變化。

(二) CPlayer

使用者可以使用 CPlayer 登入系統的 MServer 之後，觀看多媒體串流，包含預先錄製好的節目或者目前正在進行的簡報教學。若是觀看預先錄製好的節目，可以進行跨終端機換手，將串流換到其他電腦上觀看。CPlayer 的程式架構如圖 7。

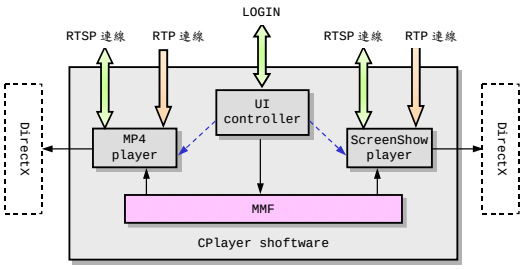

圖 7 CPlayer 方塊圖

圖 7 之中，實線箭號表示資料傳遞方向，虛線箭號則表示訊息通知。程式方塊圖說明如下：

- UI controller：執行登入到 MServer 的動作，並從 MServer 中取得多媒體串流的資料。當使用者決定播放某個多媒體串流或者進行跨終端機的換手時，UI controller 會啟動適當的 player 模組（MP4 player 或者 ScreenShow player）並將此多媒體串流的資訊通知 player 模組，讓 player 模組與 MServer 連線並取得串流，然後開始播放。
- ScreenShow player：用來播放即時簡報的多媒體串流。使用者從 UI controller 中選取的多媒體串流若是一個即時簡報教學時，UI controller 會啟動 ScreenShow player，然後由 ScreenShow player 負責建立 RTSP session 與 RTP session。ScreenShow player 會將接收到的多媒體串流加以解碼，然後透過 DirectX 播放到螢幕上。若此多媒體串流也同時含有聲音的話，也同樣透過 DirectX 播放出來。
- MP4 player：用來播放 MPEG-4 格式的多媒體串流。其動作與 ScreenShow player 都相同，差別只在於播放那一種串流資料。除此之外，使用者也可以透過 MP4 player 控制串流的播放，像是暫停播放、跳到指定時間點播放或者停止播放…等等。
- MMF：這是由 Windows 系統所提供的一種機制，全名為 Memory-Mapped File。用來作為 UI controller 與其他 2 個 player 模組之間溝通的橋樑。

(三) MServer

MServer 是系統的中心，要處理來自 CPlayer 以及 RPusher 的要求。架構如下：

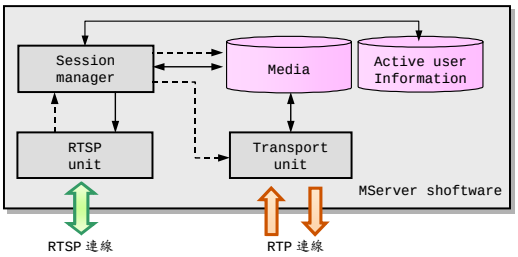

圖 8 MServer 方塊圖

- RTSP unit：負責接受 CPlayer 或者 RPusher（以下合併簡稱客戶端）的連線要求。處理 RTSP request 之後，將該 request 所需要執行的動作傳遞給 Session manager。若 RTSP request 有錯誤時，直接回應相關錯誤訊息給發出 request 的客戶端。

- Transport unit：負責與客戶端建立 RTP session。如果是 CPlayer 的話，則此模組會從 Media 中取得多媒體串流，然後傳送給 CPlayer。如果是 RPusher，則此模組會接受來自 RPusher 的多媒體串流，然後將接收到的媒體資料放到 Media 中。
- Session manager：管理目前正在播放中的 session，例如建立與結束一個 session、session 的跨終端機換手。此外，當 CPlayer 要求登入 MServer 的時候，Session manager 必須向 Active user information 取得登入的使用者正在播放的串流資訊，以及向 Media 取得 MServer 目前可以提供的多媒體串流的資料。然後將這兩部分資料傳遞給 RTSP unit，由 RTSP unit 傳回給 CPlayer。
- Media：管理 MPEG-4 檔案，以及提供緩衝（buffering）機制給即時的多媒體串流。此模組會提供一個讀取 MPEG-4 的介面供 Transport unit 呼叫以便取得 MPEG-4 檔案中的一個影像或者聲音的 frame。所以這個模組必須能解析 MPEG-4 檔案的格式。
- Active user information：記錄使用者名稱與正在播放串流的對應。也就是說，當觀看者以一個使用者名稱登入 MServer 並且開始播放一個多媒體串流時，此模組之中就會加以記錄。之後如果有相同使用者名稱登入時，此模組會把該使用者正在播放的串流資訊提供給 Session manager。

四、成果

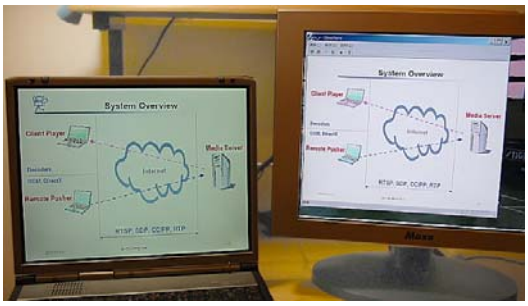

圖 9 在 RPusher 所呈現的畫面

上圖的範例展示演講者使用 RPusher 進行演講時，在 RPusher 所呈現的畫面，這些影像資料送到 MServer 之後，由 MServer 轉送給所有需要的 CPlayer。圖的左方為 RPusher 正在播放的畫面，圖的右方則是另一端的 CPlayer 所收到的畫面。在資料傳輸量的方面，這樣的一個畫面需要約 135 KBytes 的影像資料。這些影像只有 RPusher 投影片換頁的那一個時間點才會產生。

圖 10[4] Inter-device handoff

上圖的範例則展示 Inter-device handoff 的時候，MServer 可以針對各種不同的設備，傳送不同的影像品質。左圖的影像品質較差，但其影像資料量也較少。以這個影片來說，此影片畫面寬為 176 pixels，高為 144 pixels。我們取其前 3000 個 frames 加以統計，左圖產生的資料量比右圖少了 8578 KBytes 左右。

五、結論與未來研究

(一) 結論

在這一篇論文中，我們嘗試提出了一個系統架構，在這個架構之下，觀看者使用 CPlayer 除了可以觀看伺服器（MServer）上的多媒體影音檔案之外，也可以看到由遠端演講者使用 RPusher 推播到伺服器上的多媒體簡報串流。不管是觀看者或者演講者，都可以在任何地方透過網路連上 MServer 參與教學活動，並且具備可移動的能力。由於演講者的 RPusher 和觀看者的 CPlayer 之間存在一個 MServer，所以觀看者並不需要知道演講者在什麼地方，觀看者只需要連上 MServer 即可看到演講者正在進行的 session。

觀看者除了可移動之外，還可以進行跨終端機的換手。進行跨終端機換手的時候，MServer 可以根據觀看者使用的設備來調整影像的 bit-rate。當觀看者使用一般桌上型電腦時，MServer 可以傳送較佳的影像畫質（相對也會有較多的影像串流資料量）。如果觀看者將 session 切換到其他計算能力較差的設備時，MServer 可以即時地減少影像串流資料量。讓計算能力較差的機器仍然可以即時地解壓縮影像資料，不至於因為計算能力不足而造成影像播放的不連續。此外，系統也提供一個登入機制讓使用者可以很容易地進行跨終端機換手。

在演講者這一端，透過 RPusher 將演講的

[4]此圖為歌手孫燕姿MTV的影像畫面，其版權屬於該畫面製作公司或個人所有。

投影片畫面與聲音資料推播到 MServer 上。MServer 適當地將推播來的媒體串流加以緩衝儲存後，可以等待觀看者來觀看。由於投影片影像畫面的一些特性，例如在更換投影片畫面時並非影像中所有地方都會發生變化以及每一個畫面都會停留一段時間，所以一般的影像壓縮方式或者螢幕擷取（screen capture）方式並不太適合用來處理這樣的影像串流。在這裡我們針對這些特性提出 Possible Variable Area Prediction 演算法，它可以減輕 RPusher 計算的負擔並且減少傳送出去的影像資料量。

(二) 未來研究

目前系統中的 RPusher 和 MServer 之間並無任何資料傳輸量的調整機制，也就是說不管這一段的網路狀況為何，RPusher 總是送出相同的資料量給 MServer。未來如果可以根據網路狀況來調整資料量的話，可以讓這一段的網路傳輸更加順暢。以目前的系統實做方式來說，可以調整的部分包括改變影像的顏色數和影像壓縮的品質等等。由於 RPusher 端在擷取畫面的時候使用 32 位元的顏色數會產生較多的資料，若是將顏色降為 256 色灰階的話則可以減少一些資料量。目前 RPusher 將畫面擷取下來之後，會將畫面壓縮成 JPEG 格式，而 JPEG 在壓縮的時候可以選擇壓縮率，越大的壓縮率可以產生越少的影像資料。

因為 RPusher 或者 MServer 都是使用 UDP 傳送投影片影像資料，而投影片影像被切割成小區塊之後，可能在傳輸的過程中遺失。因此不管是 MServer 和 RPusher 之間，或者 MServer 和 CPlayer 之間都需要增加重傳影像區塊的機制。否則就必須改用 TCP 來傳送影像區塊。這一點和一般的 real-time streaming 特性是不太一樣的。

當真正的演講或者教學在進行的時候，必定會需要一個可以讓觀看者與演講者互動的機制，最常見的例子就是觀看者可以發問問題。由於觀看者並非集中在一個地方，而是分散在網路上，所以當有觀看者要發問的時候，應該要抑制其他觀看者發問的權力，等到發問完畢之後，才可以進行回答或者下一個發問動作。

為了增加觀看者端的效果，我們也可以把演講者端的滑鼠動作傳送出去，讓觀看者可以看到演講者滑鼠的位置。此外，經常在演講的時候，演講者會在投影片畫面上畫線畫者畫上其他東西。如可以將這些變化一併傳遞給觀看者的話，也可以增加演講時的效果。

六、參考文獻

[1] Audio-Video Transport Working Group, et al. "RTP: A transport protocol for real-time applications", *RFC 1889*, January 1996

[2] Bradley Bargen, Terence Peter Donnelly. *Inside DirectX*, *Microsoft press*, Washington, 1998

[3] Kraig Brockschmidt. *Inside OLE – 2nd edition*, *Microsoft press*, Washington, 1995

[4] Perkins, C.E. "Mobile IP", *IEEE Communications Magazine*, Vol. 35, pp. 84-99, May 1997

[5] Yih-Farn Chen, et al. "Personalized Multimedia Services Using A Mobile Service Platform", *WCNC2002*, pp. 918 –925, USA, March 2002

[6] Ross Cutler, et al. "Distributed Meetings: A Meeting Capture and Broadcasting System", *10th ACM international conference on Multimedia*, pp. 503-512, France, December 2002

[7] R. Fielding, et al. "Hypertext Transfer Protocol -- HTTP/1.1", *RFC2616*, June 1999

[8] M. Handley, V. Jacobson. "SDP: Session Description Protocol", *RFC2327*, April 1998

[9] Weiping Li. "Overview of Fine Granularity Scalability in MPEG-4 Video Standard", *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 11, No. 3, March 2001

[10] Mikael Nilsson, Johan Hjelm, Hidetaka Ohto. "Composite Capabilities/Preference Profiles: Requirements and Architecture", *W3C*, July 21, 2000

[11] Hidetaka Ohto, Johan Hjelm. "CC/PP exchange protocol based on HTTP Extension Framwork", *W3C*, June 1999

[12] H. Schulzrinne, A. Rao, and R. Lanphier. "Real Time Streaming Protocol (RTSP)", *RFC 2326*, April 1998.

[13] Audio-Video Transport Working Group, H. Schulzrinne. "RTP Profile for Audio and Video Conferences with Minimal Control", *RFC 1890*, January 1996

[14] Lori Turner. "Automating Microsoft Office 97 and Microsoft Office 2000", *MSDN Library*, July 2001