黎明學報 第二十九卷 第一期 中華民國一〇七年一月三十一日
JOURNAL OF LEE-MING INSTITUTE OF TECHNOLOGY
VOL. 29, NO.1, JAN. 2018

雲端協同作業平台之設計-以WebRTC為基礎

朱仁貴[1]、洪儀竣[2]
[1]黎明技術學院電機工程系
[2]台北科技大學電機工程系

*Email:

摘要

本論文以WebRTC實作雲端協同作業平台系統，功能包含會議室建立與管理、即時文字傳輸、圖片檔案分享和訊息紀錄中心、多人音視訊與電腦畫面分享、允許團隊成員在任何地點同時登入使用，是一個方便成員溝通、討論，展開會議的多媒體平台。本平台採用MVC (Model-View-Controller) 設計架構，便於各項功能製作與使用，更能簡化程式的開發與維護。分為伺服器端和用戶端，伺服器主要負責文字、圖片和檔案的傳輸，並將訊息存放於MongoDB，同時以WebRTC建構多人視訊會議室功能。用戶端採用Express Web Framework，並在Node.js撰寫事件驅動I/O伺服器，讓用戶端擁有順暢的使用介面以達到更好的協作傳輸共享與溝通功能。

關鍵字： MVC、WebRTC、MongoDB、Node.js事件驅動I/O

1. 簡介

由於網際網路的蓬勃發展，引發多樣性的應用與服務，其中以即時通訊軟體最為盛行，目前知名的即時通訊軟體為Line[1]和Skype[2]，且不斷的開發新功能。手機通話時代已被通訊軟體取代。即時通訊軟體從單人對單人的文字訊息，變成多人對多人的文字、圖片、檔案傳輸，聲音、影像，直接視訊影像溝通，即時通訊軟體的功能也日趨強大。

對一般使用者而言，即時通訊軟體最主要的功能，就是進行聊天，文字的即時傳輸。但對現在的公司，很多工作團隊在不相同環境工作，由於需要長時間的視訊會議[3]，會出現溝通不良而造成損失。所以，對即時視訊的影像解析度、語音清晰度、頻寬的需求量和簡易方便的使用者介面[4]，種種嚴峻需求都開始被提出。

隨著Web時代不斷發展，許多功能(包括即時通訊軟體)都被開發進瀏覽器使用，不必再安裝軟體，可減輕硬體的負擔，智慧型手機[5]只需使用瀏覽器就能達到即時視訊的會議功能。基於上述的原因，本論文提出協同作業平臺觀念，提供研討室與會議室的系統，使不同地點的電腦、手機和平板只需瀏覽器支援都能使用此服務。

協同作業平臺系統，分為兩大主要功能，分別是會議室與研討室。會議室以WebRTC（Web Real-Time Communication）[6]實現，透過網頁瀏覽器進行協同作業平臺各項功能，展開點對點(P2P)[7]的音、視訊和電腦指定畫面的即時串流[8]分享，在會議室中提供多人多畫面的功能服務，使用者能依需求選取分享單音軌和多個畫面展開會議。研討室以Node.js[9]網頁應用程式語言，配合多種的第三方模組包實現研究討論的各種功能，研討室功能包含傳送研討訊息、檔案分享、訊息中心查詢和房間管理等功能。WebRTC只需要簡易的溝通即能穿越防火牆(Solving the Firewall and NAT Traversal Issues for Multimedia over IP Service)[10]認證，提供點對點資料通訊通道，透過瀏覽器展開即時會議。使用Node.js的語言特性，系統採非同步處理，使用者能即時觸發事件請求，當伺服器將事件處理完成後回應使用者，此特性既能減輕系統伺服器的負擔，也避免無限等待或死結程序的發生，在事件觸發後使用者還能繼續使用其他功能服務。

本論文設計的協同作業平臺系統特性如下所列：

1. 設計和實作Web 架構的多人網路協同作業平臺，提供研討室和會議室系統。
2. 使用Node.js事件驅動機制減少資料傳輸量，降低伺服器端負荷。
3. 伺服器端採用MVC設計模式，增加系統的擴充性和開發效率。
4. 使用WebRTC技術，實現即時視訊影像與電腦畫面的多人即時會議系統。
5. 實現研究討論功能，文字溝通、檔案共享和簡便管理的多人研討系統。
6. 實現訊息中心，紀錄每次研討訊息內容，並能查詢研討訊息。

2. WebRTC 技術

2011年以前瀏覽器之間要實現即時通信需要私有技術，大部分透過外掛程式和用戶端安裝使用。對於許多用戶來說，外掛程式的下載、安裝和更新是複雜、繁瑣和容易出錯的操作。對於開發人員來說，外掛程式的整合、測試、部署、錯誤修復和維護同樣困難，同時在技術上還涉及到版權的問題。Google在

2010年收購了GIPS（Global IP Solutions）公司獲得WebRTC 技術，在2011年依照BSD協議公開WebRTC的程式碼，同年w3c [11]啟動WebRTC計劃，使WebRTC成為HTML5 [12]標準的一部分。WebRTC實現基於網頁的視頻會議，其採用標準是WHATWG (Web Hypertext Application Technology Working Group) [13]協議，目的是通過瀏覽器提供簡單的JavaScript就可以達到即時通訊能力。WebRTC的主要目的是讓Web開發者能夠基於瀏覽器（例: Chrome、FireFox...）輕易快捷開發豐富的即時多媒體應用，而無需下載安裝任何套件，也無需關注多媒體的信號處理過程，只需編寫簡單的JavaScript程序即可實現；另外WebRTC還能夠建立互聯網瀏覽器之間的即時通信的平台，形成開發者與瀏覽器廠商良好的生態環境。

WebRTC提供視頻會議的核心技術，包括音視訊的採集、編解碼、網絡傳輸、顯示等功能，還支持跨平台，Windows、Linux、Mac和Android皆可運行，WebRTC以瀏覽器作為服務的方式，因此也受到瀏覽器的支援度影響，WebRTC技術架構如圖2.1所示:

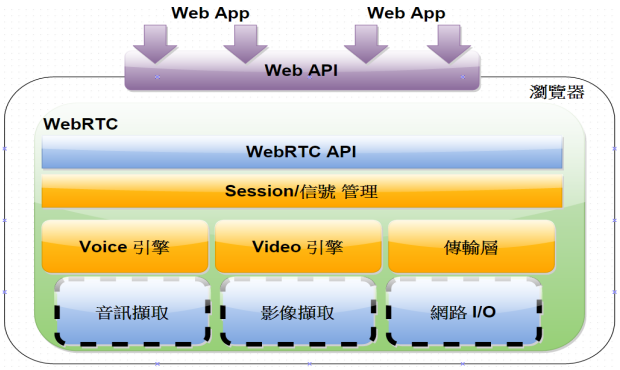

圖2.1　WebRTC技術架構圖

Web API層:
Web開發者透過JavaScript，即能使用WebRTC技術核心所提供的功能服務，能夠容易地開發多媒體即時的Web應用。

WebRTC API層:
瀏覽器廠商的API層，多瀏覽器的溝通與傳遞，此層可分為Network Stream API、 RTCPeerConnection、Peer-to-peer Data API三類。
Network Stream API :
1. MediaStream媒體數據流，一個媒體流包含多個的媒體數據源，媒體流的數據源必須同步呈現。
2. MediaStreamTrack媒體數據源，例如音頻數據源和視頻數據源，多個相互之間有關聯的媒體數據源（比如有同步關係的音頻視頻媒體數據源）構成一個媒體流。

RTCPeerConnection :
1. RTCPeerConnection，允許用戶在兩個瀏覽器之間直接通訊。
2. RTCIceServer, ICE (Interactive Connectivity Establishment) Server，結合STUN和TURN的NAT穿越防火牆技術。如圖2.2所示。
3. RTCIceCandidate，ICE協議的候選者。

Peer-to-peer Data API :
DataChannel，數據通道接口為在兩個節點之間的雙向數據通道。

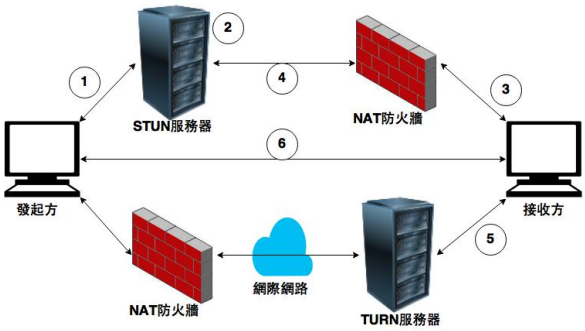

圖2.2　ICEServer穿透防火牆流程

Session/信號管理和傳輸層：
瀏覽器提供會話建立和管理功能，組件採用Google的語音通話和Libjingle [14]的部分套件，無須使用xmpp/Jingle協議。

Voice引擎：
iSAC (Internet Speech Audio Codec):針對VoIP和音頻流的音頻編解碼器，是WebRTC音頻引擎認可的編解碼器:
- 採樣頻率：16 khz，24 khz，32 khz
- 自適應速率為10 kbit/s ~ 52 kbit/s
- 自適應包大小：30~60 ms
- 算法延時：frame + 3 ms

iLBC (Internet Low Bitrate Codec):
- VoIP音頻流的語音編解碼器
- 採樣頻率：8 khz
- 20 ms位元率為15.2 kbps
- 30 ms位元率為13.33 kbps
- 標準由IETF RFC3951和RFC3952定義

NetEQ for Voice:實現音頻軟體的語音信號處理元件，其含有NetEQ算法、Acoustic Echo Canceler (AEC)和Noise Reduction (NR)三個主要單元。
NetEQ算法：自適應抖動控制算法和語音包丟失隱藏算法，使其能夠快速且高解析度地適應不斷變化的網絡環境，確保音質優美且緩衝延遲最小，能夠有效的處理由於網路抖動和語音包丟失時候對語音質量產生的影響。
Acoustic Echo Canceler (AEC):
迴聲消除器是基於軟體的信號處理元件，能即時的去除mic採集到的迴聲。
Noise Reduction (NR):
噪聲抑制是基於軟體的信號處理元件，用於消除與相關VoIP的背景噪聲（嘶嘶聲，風扇噪音等等）。

Video引擎：
VP8視頻圖像編解碼器是WebRTC視頻引擎的編解碼器，VP8適合即時通信應用場景，因為它主要是針對低延時而設計的編解碼器。視頻抖動緩衝器(Video Jitter Buffer) [15]可以降低由於視頻抖動和視頻信息包丟失帶來的不良影響。:圖像質量增強模塊(Image enhancements)對網絡攝像頭採集到的圖像進行處理，包括明暗度檢測、顏色增強、降噪處理等功能，用來提升視頻質量。

3. 系統設計

本系統分為Meeting Server、Chat Server兩個部分。Meeting Server主要功能為WebRTC多媒體視訊和桌面即時分享。使用者透過平板、電腦、手機和筆電的瀏覽器選擇分享的視訊，進行點對點的多向分享而展開即時視訊會議。Chat Server則透過Node.js多項的模組完成文字的通訊傳輸、圖片上傳分享、訊息中心紀錄等功能，訊息中心會將文字和圖片資訊存在MongoDB，方便使用者瀏覽，使不在相同環境的工作團隊或臨時展開的小會議能利用此系統的協同功能，達到有效率的溝通與會議討論。協同作業平臺系統架構如圖3.1所示，本章根據此架構進一步的功能介紹。

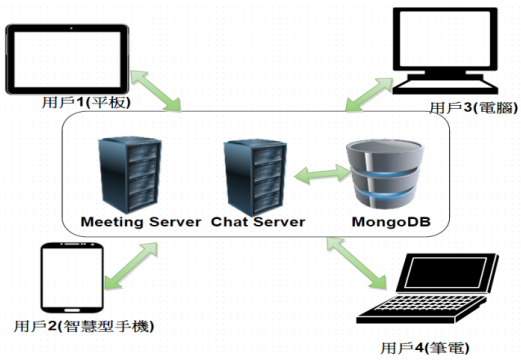

圖3.1　協同作業平臺系統架構圖

Meeting Server是由WebRTC建立的點對點溝通傳輸系統平台，透過WebRTC建立多人的點對點連線，各使用者進入會議室，同時列為RTCIceCandidate，以RTCIceServer穿透防火牆，完成會議的連線傳輸設定。Meeting Server整體系統架構，如圖3.2所示。

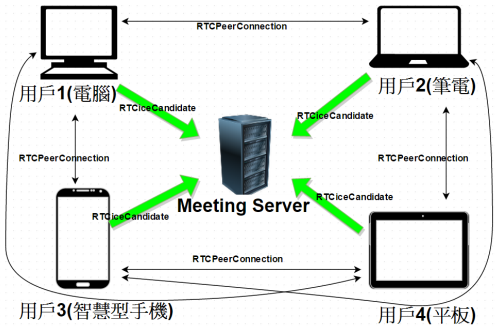

圖3.2　Meeting Server整體系統架構

Meeting會議室功能分為三個區塊:操作介面、分享音訊、視訊和分享指定桌面，如圖3.3所示。

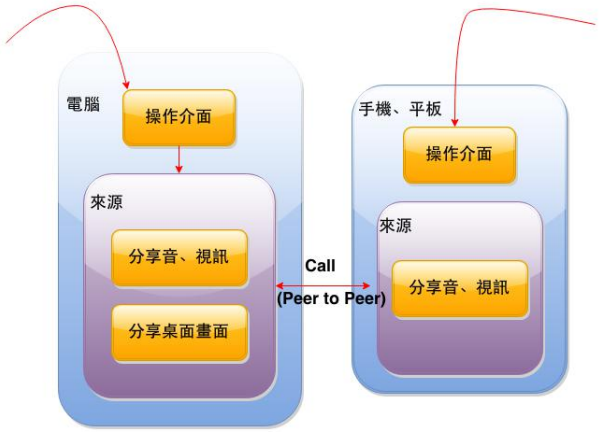

圖3.3　Meeting會議室功能

- 操作介面：使用者進入會議室，呼叫連線對象，建立點對點分享通道。
- 分享音訊、視訊: 使用者指定分享的視訊鏡頭與麥克風，進行通話分享。
- 分享指定桌面：使用者指定分享桌面來源，只能由電腦作業系統做多來源分享。

Chat Server整體架構透過以上模組，實現系統平台的角色系統、房間管理、文字傳遞、圖片傳輸和訊息中心的紀錄，以實現研討室的協作平台。

Chat研討室功能分為六個區塊: 操作介面、角色介面、房間管理、文字訊息傳遞、圖片傳輸和訊息中心:

1. 操作介面:登入、註冊、設定個資和帳號，使用者能透過手機、平板和電腦經由瀏覽器操作各項功能。
2. 角色介面：角色的設定，能同時加入多個房間，進行不同工作性質的研討。
3. 房間管理：創建、修改、刪除和驗證，各房間的資訊和啟用。
4. 文字訊息傳遞：訊息框能夠即時的傳遞文字訊息。
5. 圖片傳輸：上傳圖片，圖片格式支援gif、jpg、jpeg和png，輔助研討的資訊。
6. 訊息中心：各房間文字與圖片紀錄中心，擁有時間和區段的搜尋訊息資料。

Chat Server初始化系統後，即會進入等待事件發生，當使用者進入使用操作介面，對各項功能產生觸發事件，登入後會進房間列表的選擇，選擇要進入的房間研討室，房間觸發想要的事件，分別為文字訊息傳遞、上傳圖片和前往訊息中心，文字訊息傳遞能夠即時的發布到所有在此研討室的使用者訊息框，上傳圖片也能夠上傳至此研討室檔案列表和訊息框中，訊息框在更新後都會往返房間繼續下一事件觸發，信息中心能瀏覽此研討室歷史的訊息，也能進一步的搜尋關鍵字，以有效率的查詢內容。

4. 系統實作

會議事運作流程，如圖4.1所示：

1. 伺服器端監聽有使用者進入會議室，將使用者建立RTCIceCandidate。
2. 成為RTCIceCandidate，請求伺服器端建立穿越防火牆技術。
3. 伺服器開始建立STUN與TRUN的NAT穿越防火牆技術，並回應請求。
4. RTCIceCandidate接收成功穿越防火牆信息，並回應請求。
5. 使用者即建立即時傳輸的RTCPeerConnection通道。

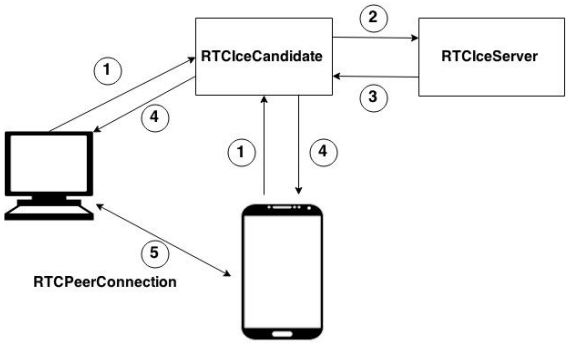

圖4.1　會議事運作流程

RTCPeerConnection通道建立後，使用者採用WebRTC技術分享音、視訊和桌面畫面的即時傳輸流程，如圖4.2所示：

1. RTCPeerConnection通道建立完成，使用者需完成上述的連線流程。
2. 使用者透過WebRTC技術中的MediaStreamTrack，取得Camera和Mic的信號來源，或是電腦桌面畫面的訊號。
3. 成功取得信號來源，將來源轉成能夠傳輸的媒體串流。
4. 回應媒體串流給使用者，使用者能夠看見自己分享的媒體串流畫面。
5. 最後透過RTCPeerConnection傳輸即時的媒體串流，實現會議室開會功能。

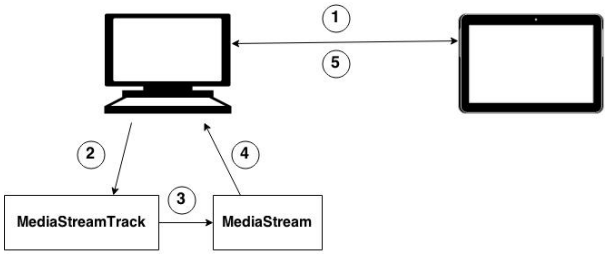

圖4.2　RTCPeerConnection即時傳輸流程

本論文實作的網路會議系統，分為七個控制器來組織實作，如表4.3系統控制器負責組織功能說明所示。

表4.3　系統控制器負責組織功能說明

控制器	負責功能說明
account	帳號管理，註冊使用者、使用者更新個人資料和帳號設定等。
rooms	房間管理，新增和刪除房間、使用者更新房間資料。
messages	訊息管理，實現研討室功能，傳送和廣播訊息等
files	檔案管理，實現上傳和廣播檔案功能。
transcript	紀錄中心，紀錄研討室訊息和檔案資料訊息。
users	使用者列表，研討室上線使用者列表。
connections	連線狀態，使用者與伺服器端連線狀態。

各項控制器分別負責不同的功能，當使用者請求送至伺服器時，控制器會根據被觸發的事件，呼叫實際處理的控制器，處理使用者請求，並透過模型和檢視回應相對的請求事件。以下將介紹各控制器所負責的事件工作：

account 控制器

account控制器用來註冊、登入和修改使用者資料，使用者可以使用此功能向系統請求使用者帳號相關功能，此功能流程如圖4.4所示。

1. 使用者觸發功能事件的請求，轉交給account控制器處理。
2. account控制器使用users 模型，處理與MongoDB資料溝通的工作。
3. users 模型將使用者各項資料寫入或修改資料庫，users 模型會判斷資料是否符合資料庫型態定義，並返回處理結果。
4. account控制器根據users 模型回傳的處理結果，回應使用者完成事件功能與否。

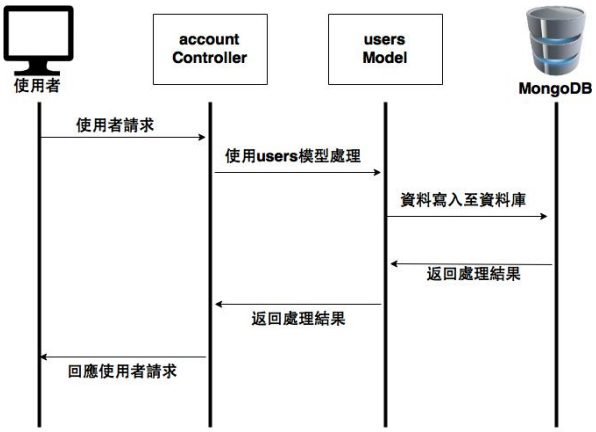

圖4.4　account控制器的流程

rooms 控制器

rooms控制器用來建立、修改和開起研討室房間資料，使用者可以使用此功能向系統請求研討室房間相關功能。此功能流程如圖4.5所示。

1. 使用者觸發功能事件的請求，轉交給rooms控制器處理。
2. rooms控制器使用room 模型，處理與MongoDB資料溝通的工作。
3. room模型將研討室房間各項資料寫入或修改資料庫，room模型判斷資料是否符合資料庫型態定義，並返回處理結果。
4. rooms控制器根據room模型回傳處理結果，回應使用者完成事件功能與否的狀況。

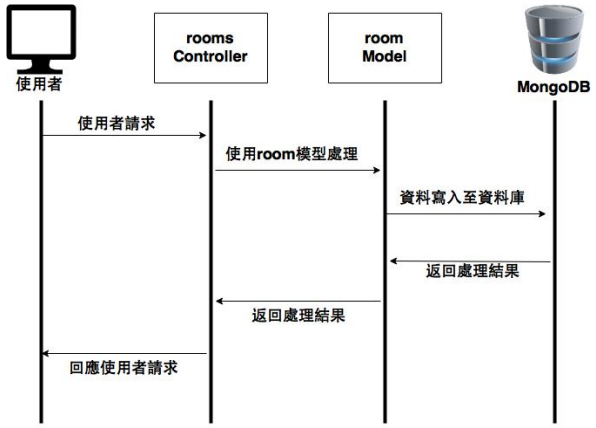

圖4.5　rooms控制器的流程

rooms控制器同時監聽研討訊息，使用者進入研討室房間時，可以取得其他使用者發送至伺服器的研討訊息，透過與伺服器建立的Socket.io即時性傳輸資料，如圖4.6所示。

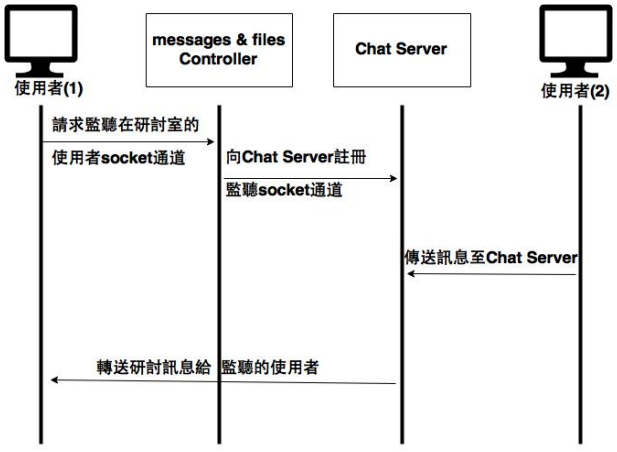

圖4.6　監聽研討訊息rooms 控制器的流程

此功能流程如下所列：

1. 將使用者的socket通道監聽請求，轉交給rooms控制器處理。
2. rooms控制器向Chat Server註冊該使用者監聽的socket通道。
3. 研討室房間內其他的使用者傳送研討訊息至Chat Server。
4. Chat Server將該研討訊息送給在此研討室房間所有監聽的socket通道使用者。

messages和files控制器

當使用者於進入研討室房間時，使用者採用messages和files控制器傳送研討訊息和上傳檔案至伺服器，並建立研討訊息內容或上傳檔案的檔案資訊。此功能流程如圖4.7所示。

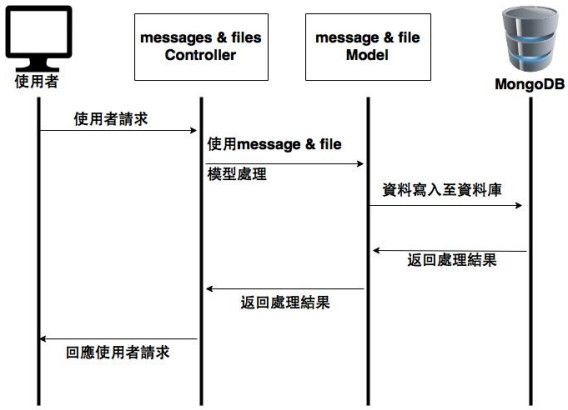

圖4.7　messages和files控制器的流程

此功能流程如下所列：

1. 使用者觸發功能事件的請求，轉交給messages或files控制器處理。
2. messages或files控制器使用message或file模型，處理與MongoDB資料溝通的工作。
3. message或file模型將研討訊息或上傳檔案各項資料寫入或修改資料庫，message或file模型會判斷資料是否符合資料庫型態定義，並回傳處理結果。
4. messages或files控制器根據message或file模型回傳的處理結果，回應使用者完成事件功能成功與否的狀況。

transcript控制器

transcript控制器用來查詢研討訊息的歷史紀錄，使用者可以依據近期訊息或日期期間進行有效率且詳細的瀏覽研討訊息，應用room Model向MongoDB查詢研討訊息和上傳檔案的研討室資料，利用message和file Model向MongoDB取得資料訊息。如圖4.8所示。

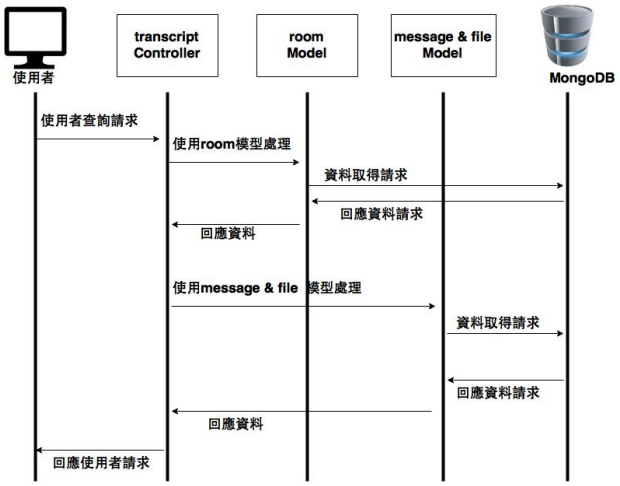

圖4.8　transcript控制器的流程

此功能流程如下所列：

1. 使用者觸發功能事件的請求，轉交給transcript控制器處理。
2. transcript控制器使用room模型，處理與MongoDB資料溝通的工作。
3. room模型回應資料給transcript控制器，查詢資料請求並回傳處理結果。
4. transcript控制器取得room資料所有的message和file陣列資料，再使用message和file模型，處理與MongoDB資料溝通的工作。
5. message和file模型回應資料給transcript控制器，回應處理資料，並回傳處理結果給使用者。

connections控制器

connectons控制器管理伺服伺連線狀態，當使用者登入時觸發事件，系統會紀錄使用者線上狀態，使用者可知道與伺服器連線的狀態。

1. 使用者觸發登入功能事件的請求，轉交給connectons控制器處理。
2. connectons控制器向Chat Server註冊該使用者監聽與伺服器連線狀態。
3. 當伺服器錯誤或是網路斷線，使用者監聽即斷線，並回應處理結果。

5. 系統分析與評估

現今即時通訊軟體透過網際網路進行服務，用戶能夠便利的透過3C電子產品於任何時間任何地點快速通信，普遍的功能有訊息傳送、視訊通話、資料分享和創立群組，本小節將進行即時通訊軟體各項功能分析，本系統所提供的協同作業平臺與現今市面上兩個即時通訊軟體Skype與LINE，進行優缺點的比較分析，如表5.1所示。

表5.1　協同作業平臺與Skype和LINE優缺點

功能	協同作業平臺	Skype	LINE
安裝	於瀏覽器使用，無需任何安裝。	專屬應用程式。	專屬應用程式。
訊息傳送	使用者能指定對象傳輸訊息。	使用者能指定對象傳輸訊息。	使用者能指定對象傳輸訊息。
視訊通話	會議室提供多來源視訊進行分享通話。	單一視訊進行通話。	單一視訊進行通話。
群組	建立專屬研討室，進行訊息傳送。	能夠建立群組，需將使用者依依加入。	建立專屬房間，進行訊息傳送。
視訊中訊息傳送	在會議室內能即時傳輸訊息給指定對象。	無法。	進行視訊通話時，傳輸訊息給指定對象。
圖片存取	能同時傳輸多筆圖片。	能同時傳輸多筆圖片，提供貼圖。	能同時傳輸多筆圖片。
歷史紀錄	提供完整的訊息中心，依據時間進行訊息與圖片查詢，系統不會刪除任何記錄。	訊息依據持有裝置而存在，在更換持有裝置時，訊息與圖片將會刪除，圖片在一段時間後會失去有效期限。	訊息與圖片隨著身分移動，在指定電腦的訊息與圖片只能於此電腦中取得。

6. 結論

隨著網際網路的進步，研發團隊通常跨多企業合作，在多個團隊下進行的開發工作，往往會產生分歧，需要便利且清楚溝通與研討環境。此時會議和商談顯得相當重要。本論文實作系統會議室與研討室功能，會議室提供簡易會議室和多點分享會議室，能依需求提供簡便快速或多視頻的分享，研討室提供即時文字和檔案的傳輸與共享。使用者能透過個人電腦、筆電、智慧型手機和平板於任何時間任何地點，展開研討與會議的功能服務，本系統不須安裝任何額外軟體，只需要開啟支援WebRTC技術的瀏覽器即可使用此協同作業平臺所提供的功能服務，本平臺服務為解決工作團隊和商業會議在視訊會議上的不足，支援電腦畫面多來源的即時分享，使得視訊會議能夠更清楚

展開，研討室備有訊息中心，會紀錄所有在線的訊息內容和檔案資料並提供查詢的服務，使研討與會議更清楚且便利的帶來科技生活。

7. 參考文獻

1. Line, http://line.me/zh-hant/, Jun.2015
2. Skype, http://www.skype.com/zh-Hant/home/, Jun. 2015
3. W Hyun, S.G Kang, "Interoperable Telepresence Services: Beyond HD-Videoconferences and Towards Telepresence",*APSCC (Asia-Pacific Conference on Services Computing)*, pp. 327-329,2006
4. 李文宏，以**SIP** 為基礎之網路視訊電話實作，碩士論文，國立臺北科技大學電機工程系碩士班，台北，2006。
5. 智慧型手機, https://en.wikipedia.org/wiki/Smartphone, Jun. 2015
6. S Loreto, S.P Romano, "Real-Time Communications in the Web: Issues, Achievements, and Ongoing Standardization Efforts", *IEEE Internet Computing*, vol.16, no. 5, pp. 68-73, Sept.-Oct. 2012
7. M. J Neely, "Optimal Peer-to-Peer Schedulingfor Mobile Wireless Networkswith Redundantly Distributed Data", *IEEE Transactions on Mobile Computing*, vol.13, no. 9, pp. 2086-2099, Sept. 2014
8. J Kim, "Topology and Architecture Design for Peer to Peer Video Live Streaming System on Mobile Broadcasting Social Media",*ICISA*, 2014, pp. 1-4
9. Node.js, https://nodejs.org/, Jun. 2015
10. Solving the Firewall and NAT Traversal Issues for Multimedia over IP Service, http://www.newport-networks.com/
11. w3c, http://www.w3.org/, Jun. 2015
12. HTML5, http://www.w3schools.com/html/html5_intro.asp, Jun. 2015
13. WHATWG, https://whatwg.org/, Jun. 2015
14. Libjingle, https://developers.google.com/talk/libjingle/developer_guide, Jun. 2015
15. H.Dahmouni, " The Impact of Jitter on Traffic Flow Optimization in Communication Networks", *IEEE Transactions on Network and Sevice Management*, vol. 9, no. 3, Sep. 2012

Design of WebRTC Based Cloud Collaboration Platform

Ren-Guey Chu[1], Yi - Chun Houng[2]

[1]Department of Electrical Engineering, Lee-Ming Institute of Technology

[2]Department of Electrical Engineering, National Taipei University of

*Email:

Abstract

In this paper, we implement the WebRTC-based Cloud Collaboration Platform for team cooperation. The features of the proposed system are establishing and managing meeting rooms, instant text transmission, image and file sharing, messaging records center and multiplayer video, audio and computer screen sharing. The collaboration team members can apply the proposed system to communicate, discuss and even open conference with each other over Internet. The Platform system is implemented with MVC (Model-View-Controller) design framework. The framework separates the representation of information from the users' interaction with it. MVC can reduce the development complexity and improve the system maintainability.The proposed platform include server and client. The function of the server is to transport text, image file and deposit message into MongoDB transcript message center. The other function is to share multiple video and computer screen. The user can specify a Webcam and computer screen to share Multiplayer video conference room. The client is implemented over the Event-drivenInput and Outputserver of Web Express Framework of Node.js.

Keywords: MVC、WebRTC、MongoDB、Node.js、Event-drivenInput and Output