JITEL 2017
XIII Jornadas de Ingeniería Telemática
27-29 Septiembre 2017, Valencia (España)

Actas de las XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia (España), 27-29 de Septiembre de 2017

ISBN: 978-84-9048-595-8
DOI: http://dx.doi.org/10.4995/JITEL2017.2017.6569

Aplicación Web para comunicación multimedia en tiempo real y en movilidad

Miguel Gil, Elsa Macías, Alvaro Suárez
Departamento de Ingeniería Telemática
Instituto de Ciencias y Tecnologías Cibernéticas

Universidad de Las Palmas de Gran Canaria

Dirección Postal: Edificio de Electrónica y Telecomunicación – Campus universitario de Tafira – 35017 Las Palmas de G.C.

miguelgilbr@gmail.com, alvaro.suarez@ulpgc.es, elsa.macias@ulpgc.es

***Resumen*- En este documento se presenta una aplicación Web para comunicación multimedia en tiempo real y compatible con dispositivos móviles, combinando varias tecnologías que son tendencia actualmente en el mundo del desarrollo: HTML5, WebRTC y Node.js. A partir de las tecnologías anteriores se ha desarrollado una plataforma que desde un navegador compatible permite: realizar videoconferencias en grupo, enviar y recibir archivos, conversaciones en texto privadas entre usuarios, conversaciones de texto en grupo y grabación de las emisiones multimedia de cualquiera de los usuarios de la sala. A nivel visual, se ha diseñado una interfaz simple de usar y completamente responsiva, es decir, compatible con todos los tamaños de pantallas, incluidas las de cualquier teléfono móvil. Se ha hecho comparativas con aplicaciones actuales comerciales y se observa un rendimiento similar o mejor que algunas de ellas. En cambio, el consumo de energía es muy considerable (al igual que en el resto de aplicaciones analizadas).**

***Palabras Clave*- HTML5, WebRTC, Node.js**

I. Introducción

En los últimos años, los servicios de videoconferencia a través de Internet han evolucionado notablemente, pasando de ser una tecnología cara y que requería de buenas conexiones de datos, a ser servicios habituales, con un consumo moderado de datos y al alcance de todo el mundo.

A día de hoy es posible realizar conferencias simultáneas entre varias personas de una manera simple y gratuita con aplicaciones como *Skype* [1] o *Google Hangouts* [2], aplicaciones ampliamente usadas para realizar este tipo de comunicaciones. El servicio de videoconferencia se suele integrar con otros servicios básicos, por ejemplo, la aplicación de mensajería instantánea más usada (*WhatsApp* [3]) o la red social con más usuarios (*Facebook* [4]).

En el marco de la Tele-enseñanza, los sistemas de videoconferencia todavía no se usan masivamente, a pesar de que en determinados escenarios son imprescindibles. Un ejemplo de esto es el aprendizaje de idiomas, en el que una parte consiste en hablar y escuchar, lo que implica mantener una conversación fluida en persona. Esto solo es posible usando sistemas de videoconferencia. En la práctica se suelen usar, no masivamente: a) sistemas de pago (de calidad negociable con el operador de Telecomunicación), como es el caso de *Polycom* [5], b) sistemas de pago de menor calidad como las de *ISL Group* [6], c) herramientas corporativas de elevadísima calidad como *WebEx* [7], entre otras muchas corporativas, y d) herramientas con versiones gratuitas que dan un servicio básico (como, por ejemplo, *TeamViewer* [8]). Sin embargo, las opciones para usar sistemas de videoconferencia en Tele-Enseñanza, están cambiando gracias a que a día de hoy es más sencillo acceder a herramientas y funcionalidades gratuitas, que se programan haciendo uso de la tecnología *Web Real Time Communications* (*WebRTC*) [9] [10] e *Hiper Text Markup Language versión 5* (*HTML5*) [11] [12] [13], debido a que en teoría es extremadamente fácil programarlas.

En este artículo presentamos el desarrollo integral de un servicio de videoconferencia que hace uso de WebRTC y HTML5. En la práctica se demuestra que la programación de este tipo de servicios eficientes, seguros y responsivos no es una tarea tan sencilla como indica la teoría. Este servicio se puede usar en las plataformas móviles más usadas en la actualidad (Android e iOS), además hace uso bidireccional de la comunicación entre usuarios, maneja flujos de comunicación de texto (*chat*), permite grabar las sesiones de video y enviar archivos. Se ha esmerado el diseño e implementación para que el uso del servicio sea lo más simple posible, que maximice la *Quality of Experience* (*QoE*) y aproveche al máximo los mecanismos de *Quality of Service* (*QoS*) disponibles en la red (en especial en *Wireless Fidelity* (*WiFi*), tecnología para la que hemos hecho pruebas de funcionamiento mostrando un comportamiento similar a servicios como *Skype* y *Hangout*.

II. Arquitectura del Software

El servicio propuesto se compone de dos sistemas principales:

A. Servidor

Desarrollado usando la tecnología *Node.js* [14] [15], ya que, es muy adecuada para escribir de manera simple y rápida servicios Web. Se encarga de las siguientes tareas: almacenar y hacer accesible a los usuarios la aplicación Web, guardar los datos de cada uno de los usuarios que se conecte al sistema, controlar en tiempo real la entrada y salida de usuario e informar al resto de usuarios, encargarse de la señalización a la hora de crear canales de comunicación *Peer 2 Peer* (*P2P*), ya sea para las transmisiones multimedia, o para los canales de datos, gestionar el estado de las conexiones entre los clientes, de manera que el sistema asegure las conexiones entre todos los usuarios, toda la lógica del chat de grupo.

B. Cliente Web responsivo

Desarrollado con HTML5, combinado con algunas bibliotecas *JavaScript* y *Cascade Style Sheets versión 3* (*CSS3*) [16]. Es el código que ejecuta el usuario en su navegador y se encarga de realizar las siguientes tareas: punto de entrada del usuario al sistema, gestión de la conexión con el servidor central mediante WebSockets, acceso a los medios de captura multimedia usando WebRTC, conexión entre los usuarios conectados al sistema, acceso al sistema de archivos del usuario para poder enviarlos, control de ventanas del Chat.

A continuación (Fig. 1), se describen los módulos funcionales de la solución de forma independiente al sistema que esté implicado en su funcionamiento. Esto implica que existen módulos que dependen solo del servidor, de la aplicación Web, o en el caso más común, de ambos.

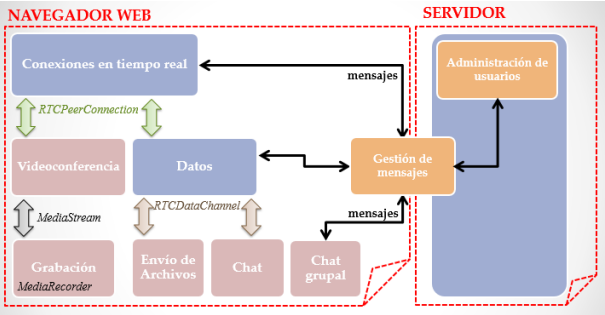

Fig. 1. Módulos funcionales de la aplicación.

El módulo *Conexiones en Tiempo Real*, se encarga de la configuración y gestión de los objetos *RTCPeerConnection*, que son los encargados de gestionar las conexiones P2P entre los usuarios. Una vez que ya tenemos establecido un canal P2P entre dos usuarios, es muy sencillo establecer un canal de comunicaciones extra usando el módulo *Datos*, que se encarga de generar los canales de datos *RTCDataChannel* usando la función *createDataChannel* de los objetos *RTCPeerConnection* generados con el primer módulo.

Ambos módulos requieren de un proceso de sincronización y señalización entre los usuarios implicados en la comunicación, de los que se encarga el módulo *Gestión de mensajes*, para el intercambio de información entre usuarios a través del servidor y el módulo *Administración de usuarios* que se encarga de almacenar y actualizar en tiempo real los datos y el estado de los usuarios del sistema.

Para finalizar, se observan los módulos de aplicación, que se corresponden con funcionalidades concretas del sistema. El módulo *Videoconferencia* se encarga de la transmisión y recepción de las transmisiones audiovisuales de los usuarios, mientras que el módulo *Grabación* graba las transmisiones recibidas, usando el objeto *MediaRecorder*. Los módulos *Envío de Archivos* y *Chat* se apoyan en el módulo *Datos* para funcionar y por último el módulo *Chat grupal* funciona gracias al servidor y a los mismos módulos encargados de la señalización y sincronización y que se comentaron en el párrafo anterior.

III. Rendimiento y Usabilidad

En este apartado presentamos en primer lugar la ventana principal del servicio desarrollado (para que se observe el diseño minimalista presentado) y en segundo lugar unas ideas mínimas sobre su rendimiento.

A. Interfaz gráfica

En la Fig. 2 se muestra la interfaz de usuario de la aplicación que consta de 3 partes fundamentales:

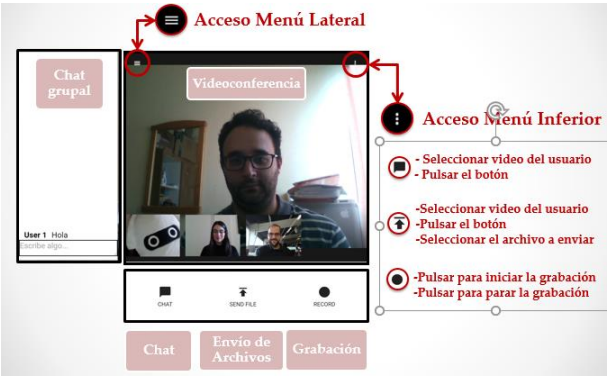

Fig. 2. Interfaz Gráfica de la aplicación

- La ventana principal en la que se observa, en primer plano, la videoconferencia principal y en ventanas más pequeñas sobrepuestas a la principal el resto de videoconferencias. Al pinchar en la videoconferencia de una ventana pequeña, esta pasa a ser la principal.

- El menú lateral, fácilmente desplegable usando el botón superior izquierdo y que se encarga de mostrar la interfaz del chat en grupo.

- El menú inferior, que se encarga del acceso directo al resto de aplicaciones del sistema: el chat privado, el envío de archivos y la grabación. Todas estas funcionalidades afectarán al usuario de la videoconferencia contenida en la ventana principal.

Comentar adicionalmente que la interfaz desarrollada es compatible con todo tipo de tamaños de pantalla (computadores de sobremesa, tabletas y teléfonos móviles), y está optimizada para su correcta visualización en dispositivos móviles.

B. Rendimiento

Para medir el rendimiento del servicio, en primer lugar, se ha analizado el consumo de *Random Access Memory* (*RAM*) del servidor ya que, en teoría, una de las mejores prestaciones del Node.js es su reducido uso de RAM. Para realizar las pruebas, se ha instalado el servidor Node.js en varios sistemas operativos distintos y se ha medido el consumo de RAM.

Los resultados de la prueba se muestran en la Fig. 2 y se han realizado en un computador *Mac* (*Apple*) con el sistema operativo *OS X El Capitan* versión 10.11.6, una configuración de 12 GB de RAM DDR3 a 1600 MHz y un procesador Intel Core i7 de 2,3 GHz. El consumo de memoria RAM es muy bajo, unos 25 MB de RAM.

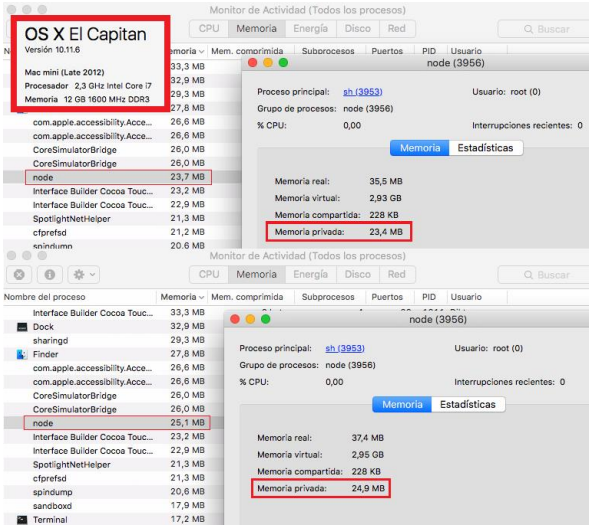

Fig. 3. Consumo de RAM de la aplicación

A continuación, se ha analizado el consumo de datos de una videoconferencia de 1 minuto de duración. Se han analizado los paquetes de la conversación, capturándolos usando el software analizador de redes *Wireshark*. En la Fig. 3 aparece el consumo de datos en bruto de todo el proceso y se distinguen 4 fases.

La primera de ellas se corresponde con el acceso al sistema y la descarga de toda la aplicación Web, cuyo procedimiento desemboca en la pantalla de acceso al sistema. La segunda fase es la entrada al sistema y la conexión con el Servidor. En este momento en el sistema no hay nadie y se está a la espera de que entre alguien con el que conectar. El tráfico en este punto sigue siendo *Transmission Control Protocol* (*TCP*).

La tercera y cuarta fase ocurren casi simultáneamente: cuando un nuevo usuario entra al sistema, el sistema lo comunica. Posteriormente comienza la negociación entre los usuarios, como paso previo a iniciar la conexión multimedia efectiva. Todo este procedimiento ocurre bajo TCP. La cuarta fase es el intercambio de datos en tiempo real y se efectúa en su mayoría con tráfico *User Datagram Protocol* (*UDP*). Una vez estabilizada la conexión, el ancho de banda usado oscila entre los l70 y los 220 paquetes por segundo y un ancho de banda equivalente en torno a los 100 KBps de media.

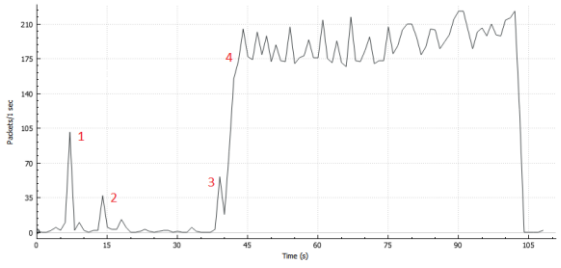

Fig. 4. Consumo de datos de la aplicación

Por último, se ha comparado el consumo de datos con los de Skype y Google Hangout. Los resultados aparecen en la tabla I, en la que se observa que la aplicación tiene un consumo moderado y sobretodo, es muy rápida a la hora de realizar una conexión efectiva.

Tabla I
COMPARATIVA

	Datos	Velocidad Tx	Tiempo conexión
Aplicación	5,19 MB	83 KB/s	**2,62 s**
Skype	**2,79 MB**	**42 KB/s**	6.5 s
Hangouts	14 MB	197 KB/s	11 s

Se hicieron otros tipos de pruebas (recogidas en [17]), en especial se hicieron pruebas de análisis de la QoE y de respuesta de la QoS en WiFi. Después de esos estudios llegamos a la conclusión que se debe dejar al programador (la herramienta que define el servicio es de código abierta) el poder variar el valor de una variable que limita el número máximo de usuarios que pueden hacer videoconferencia simultáneamente. Nosotros encontramos que el mejor número de usuarios para la mayoría de las instalaciones WiFi es 4.

IV. CONCLUSIONES

En este artículo hemos presentado un servicio de videoconferencia muy simple de usar y adecuado especialmente para tele-enseñanza, por cuanto permite flujos de video, chat archivos y grabación de sesiones. A partir de los resultados experimentales, podemos asegurar que la aplicación desarrollada tiene un rendimiento destacable y es fácil de usar.

Como trabajo futuro debemos desarrollar mecanismos de consumo inteligente de sesiones de videoconferencia adaptando inteligentemente el número de usuarios según la QoS de la Red y por otro lado optimizar el consumo de energía en los teléfonos móviles. También comparar con otras herramientas como Facebook Messenger.

AGRADECIMIENTOS

This work has been funded by the Spanish Ministry of Economy and Competitiveness/FEDER under project TEC2015-67387- C4-4-R.

REFERENCIAS

[1] Página oficial de Skype: https://www.skype.com/es/

[2] Página oficial de Google Hangouts: https://hangouts.google.com/?hl=es

[3] Página oficial de WhatsApp: https://www.whatsapp.com/?l=es

[4] Página oficial de Facebook: https://www.facebook.com/

[5] Página oficial de Polycom: http://www.polycom.es/

[6] Página oficial de ISLOnline: http://www.islonline.com/?hl=es

[7] Página oficial de WebEx: https://www.webex.es/

[8] Página oficial de TeamViewer: https://www.teamviewer.com/es/

[9] Salvatore Loreto, Simon Pietro Romano, Real-Time Communication with WebRTC. Peer-To-Peer in the browser. Editorial O'Reilly. Mayo 2014. ISBN 978-1-78216-630-6

[10] Rob Manson, Getting Started with WebRTC. Explore WebRTC for real-time peer-to-peer communication. Editorial Packt Publishing. Septiembre 2013. ISBN 978-1-449-37187-6

[11] Peter Lubbers, Brian Albers, Frank Salim, Pro HTML5 Programming. Powerful APIs for Richer Internet Application Development. Use HTML5 to create cutting-edge Web applications. Editorial Apress. 2010. ISBN 978-1-4302-2791-5

[12] Dale Cruse, Lee Jordan, HTML5 Multimedia Development Cookbook. Recipes for practical, real-world HTML5 multimedia-driven development. Editorial Packt Publishing. Mayo 2011. ISBN 978-1-849691-04-8

[13] Silvia Pfeiffer, The Definitive Guide to HTML5 Video. Everything you need to know about the new HTML5 video element. Editorial Apress. 2010. ISBN 978-1-4302-3091-2

[14] Tom Hughes Croucher, Mike Wilson, Node. Up and Running. Editorial O'Reilly. Mayo 2012. ISBN 978-1-449-39858-3

[15] Pedro Teixeira, Professional Node.js. Building Javascript-Based Scalable Software. Editorial John Wiley & Sons, Inc. 2013. ISBN 978-1-118-18546-9

[16] Página oficial de la librería NativeDroid2: http://nativedroid.godesign.ch/material/

[17] Miguel Gil, Aplicación Web para comunicación multimedia en tiempo real y en movilidad, Proyecto final de Carrera, Escuela de Electrónica y Telecomunicación, Universidad de Las Palmas de Gran Canaria, Directores: Elsa Macías y Alvaro Suárez, 2017.