

**FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**

**ESCUELA ACADÉMICA DE INGIENERIA DE SISTEMAS**

**TEMA**

**WEB SOCKETS**

**CURSO DE:**

**Programación II**

**Propósito del trabajo: Primera Parte**

Mg. Coronel Castillo Eric Gustavo

AUTOR(ES):

Aguilar Ventura Kenneth Jesús código 15201044

Cortés Baldovino Angelo Doménico código 15101022

Herrera Fiestas Jocelyne código 15201014

Los Olivos – Perú

2017 I

Este trabajo está elaborado gracias a todo el apoyo de nuestros padres para lograr nuestros objetivos y seguir con nuestras metas.

**Índice**

[**Índice** 3](#_Toc464752520)

[**Introducción** 4](#_Toc464752521)

[**CAPÍTULO I. WEB SOCKETS** 5](#_Toc464752523)

[**1.1.** **¿Qué es Web Sockets?** 5](#_Toc464752524)

[**1.2.** **Tipo de comunicación de Web Sockets** 5](#_Toc464752524)

[**1.3.** **Cuando usar Web Sockets** 6](#_Toc464752524)

[**1.4.** **Soporte de Web Sockets en navegadores Web** 6](#_Toc464752524)

[**CAPÍTULO II. FUNCIONAMIENTO DE WEBSOCKET** 7](#_Toc464752526)

[**2.1.** **Implementacion de Web Sockets** 7](#_Toc464752528)

[**2.2.** **Eventos de Web Sockets** 7](#_Toc464752528)

**2.2.1. Abierto……………………………………………………………………...**8

**2.2.2. Mensaje…………………………………………………………………….**8

**2.2.3. Cerca………………………………………………………………………..**8

**2.2.4. Error………………………………………………………………………...**8

[**2.3.** **Acciones de WebSocket**7](#_Toc464752528)…………………………………………………………9

**2.3.1. send()……………………………………………………………………….**9

**2.3.2. cerca()………………………………………………………………………**9

[**2.4.** **Ejemplo** 10](#_Toc464752528)

[**CAPÍTULO III. VENTAJAS Y DESVENTAJAS** 12](#_Toc464752529)

[**3.1.** **Identificación del problema** 12](#_Toc464752528)

[**3.2.** **Planteamiento de Solucion** 12](#_Toc464752528)

[**3.3.** **Ventajas** 13](#_Toc464752528)

[**3.4.** **Desventajas** 13](#_Toc464752528)

[**3.5.** **Caso Real** 14](#_Toc464752528)

[**Conclusiones** 15](#_Toc464752534)

[**Recomendaciones** 16](#_Toc464752534)

[**Referencias Bibliográficas** 17](#_Toc464752534)

**Introducción**

La investigación realizada busca hacer un análisis de la tecnología “Web Socket”.

El trabajo está dividido en tres capítulos. El primero abordará la implementación de esta tecnología, el tipo de comunicación que tiene y cuando usarlo, saber si facilita o perjudica nuestros próximos proyectos; en el segundo trataremos sobre el funcionamiento de la tecnología WebSocket, identificando su funcionamiento, implementación, así mismo qué eventos y acciones utiliza, dando un breve ejemplo; en el tercero nos centraremos en el planteamiento de la solución, finalizando con nuestras conclusiones.

En esta investigación, se realizarán los análisis correspondientes que permitan desentrañar la tecnología de los “Web Sockets”.

Agradeceremos las sugerencias, interrogantes y observaciones que permitan enriquecer este trabajo.

Los autores.

**CAPÍTULO I. WEBSOCKET**

* 1. **¿Qué es Web Socket?**

Es una tecnología que proporciona un canal de comunicación bidireccional y full dúplex sobre un único socket TCP (Protocolo de Control de Transmisión). Nos permite interactuar con un servidor y un cliente. Está diseñada para ser implementada en navegadores y servidores web, pero puede aplicarse por cualquier aplicación cliente/servidor.

Debido a que las conexiones TCP comunes sobre puertos diferentes al 80 son habitualmente bloqueadas por los administradores de redes, el uso de esta tecnología proporcionaría una solución a este tipo de limitaciones proveyendo una funcionalidad similar a la apertura de varias conexiones en distintos puertos, pero multiplexado, es decir, diferentes servicios Web Socket sobe un único puerto TCP(a costa de una pequeña sobrecarga de protocolo).

En el lado de cliente, Web Socket esta ya implementado en Mozilla Firefox 8, Google Chrome 4, y Safari 5 así como en su versión móvil.

* 1. **Tipo de Comunicación de Web Sockets**

La comunicación que los Web Sockets tiene es mediante un servidor que nos va a permitir conectarnos en algún lenguaje de programación de navegadores en la cual también se puede comunicar dispositivos y otras plataformas.

Un ejemplo de servidor es WebSocket.org

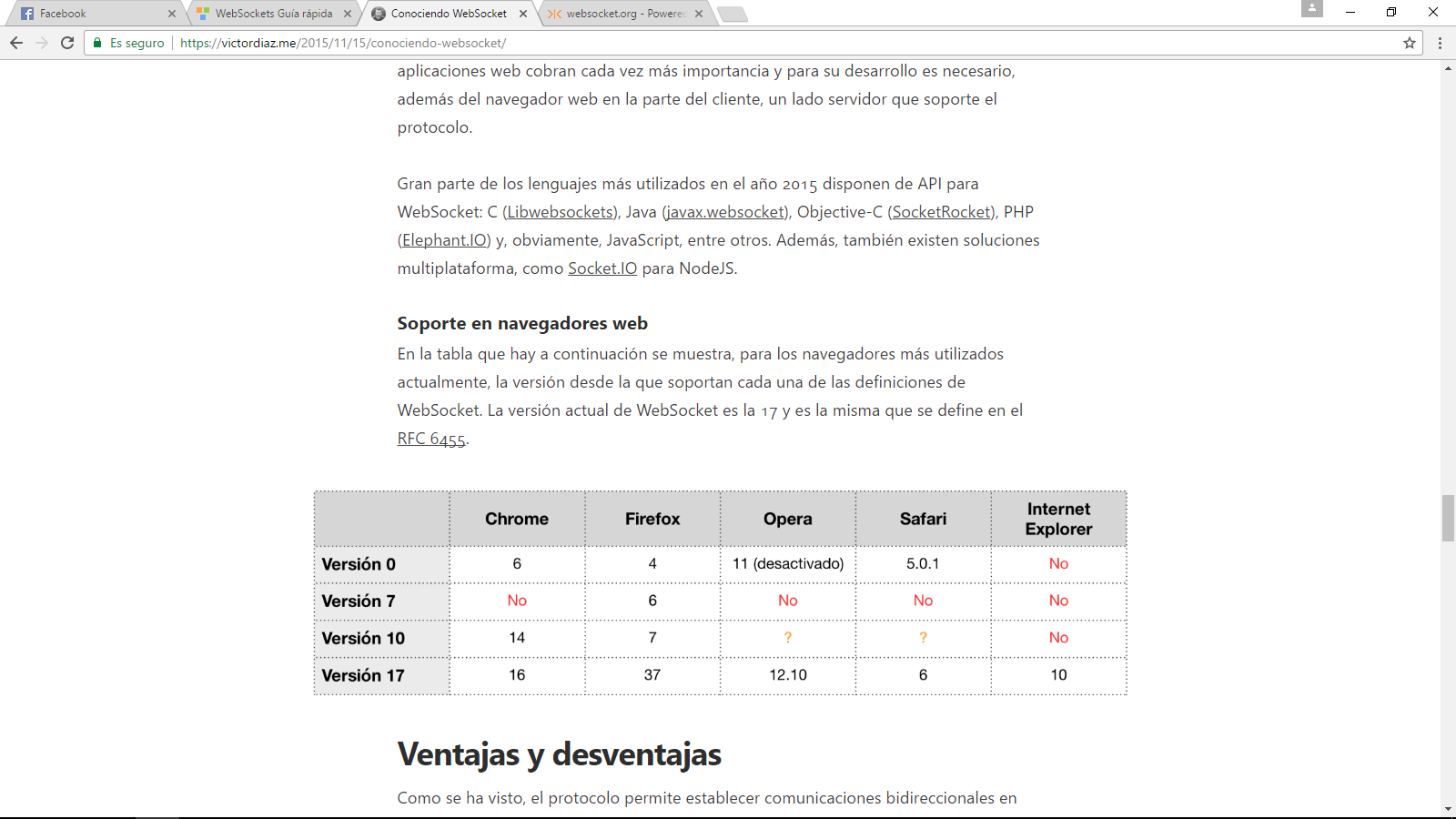
* 1. **Cuándo usar Web Socket**

Se utiliza Web Socket siempre que se necesita una conexión casi a tiempo real y de latencia baja entre el cliente y el servidor.

* Juegos Multijugadores Online.
* Aplicaciones de Chat.
* Rotativos de información deportiva, etc.

* 1. **Soporte de Web Socket en navegadores Web**

En la tabla que hay a continuación se muestra, para los navegadores más utilizados actualmente, la versión desde la que soportan cada una de las definiciones de WebSocket. La versión actual de WebSocket es la 17.



**CAPÍTULO II. FUNCIONAMIENTO DE WEB SOCKETS**

* 1. **Implementación de Web Sockets**

Los Web sockets ocupan un papel fundamental no sólo en la web, sino también en la industria móvil. La importancia de los WebSockets es la siguiente:

* Como su nombre lo indica, esta relacionado con la web. La web se compone de un montón de técnicas para algunos navegadores; es una amplia plataforma de comunicación para un gran número de dispositivos, incluyendo ordenadores de sobremesa, ordenadores portátiles, tabletas y teléfonos inteligentes.
* HTML5 aplicación que utiliza sockets de Internet funcionará desde cualquier navegador web HTML5 habilitado.
* Web Sockets es compatible con los sistemas operativos convencionales. Todos los jugadores claves de la industria móvil proporcionan Web Socket API en las aplicaciones nativas propias.
  1. **Eventos de Web Sockets**

Hay cuatro principales API WebSocket **events (abierto, mensaje, cerca, error).**

Cada uno de los eventos son manejados por la aplicación de las funciones como **onopen, enmessage, onclose, y onerror** respectivamente y se describen de la siguiente manera.

* + 1. **Abierto**

Una vez que se ha establecido la conexión entre el cliente y el servidor, el evento abierto es despedido de ejemplo WebSockets. Se llama como el apretón de manos inicial entre el cliente y el servidor. El evento, que se eleva una vez establecida la conexión, se llama **onopen.**

* + 1. **Mensaje**

Este evento de aviso ocurre normalmente cuando el servidor envía algunos datos. Los mensajes enviados por el servidor al cliente pueden incluir mensajes de texto sin formato, datos binarios o imágenes. Cada vez que se envían datos, la **función onmessage se dispara.**

* + 1. **Cerca**

La función cerca marca el final de la comunicación entre el servidor y el cliente. El cierre de la conexión es posible con la ayuda del evento **onclose.** Después de marcar el final de la conversación con la ayuda de este**, no hay mensajes que se puedan transferir adicionalmente entre el servidor y el cliente.**

* + 1. **Error**

Muestra un error, lo que ocurre durante la comunicación. Está marcado con la ayuda del evento **onerror. Siempre va seguido de la terminación de la conexión(onclose).**

* 1. **Acciones de Web Sockets**

Los eventos se desencadenan por lo general cuando algo sucede. Por otra parte, las acciones se toman cuando un usuario quiere que suceda algo. Las acciones se realizan por llamadas explicitas utilizando las funciones de los usuarios.

El protocolo WebSocket es compatible con dos acciones principales, a saber:

* send()
* close()
  + 1. **send()**

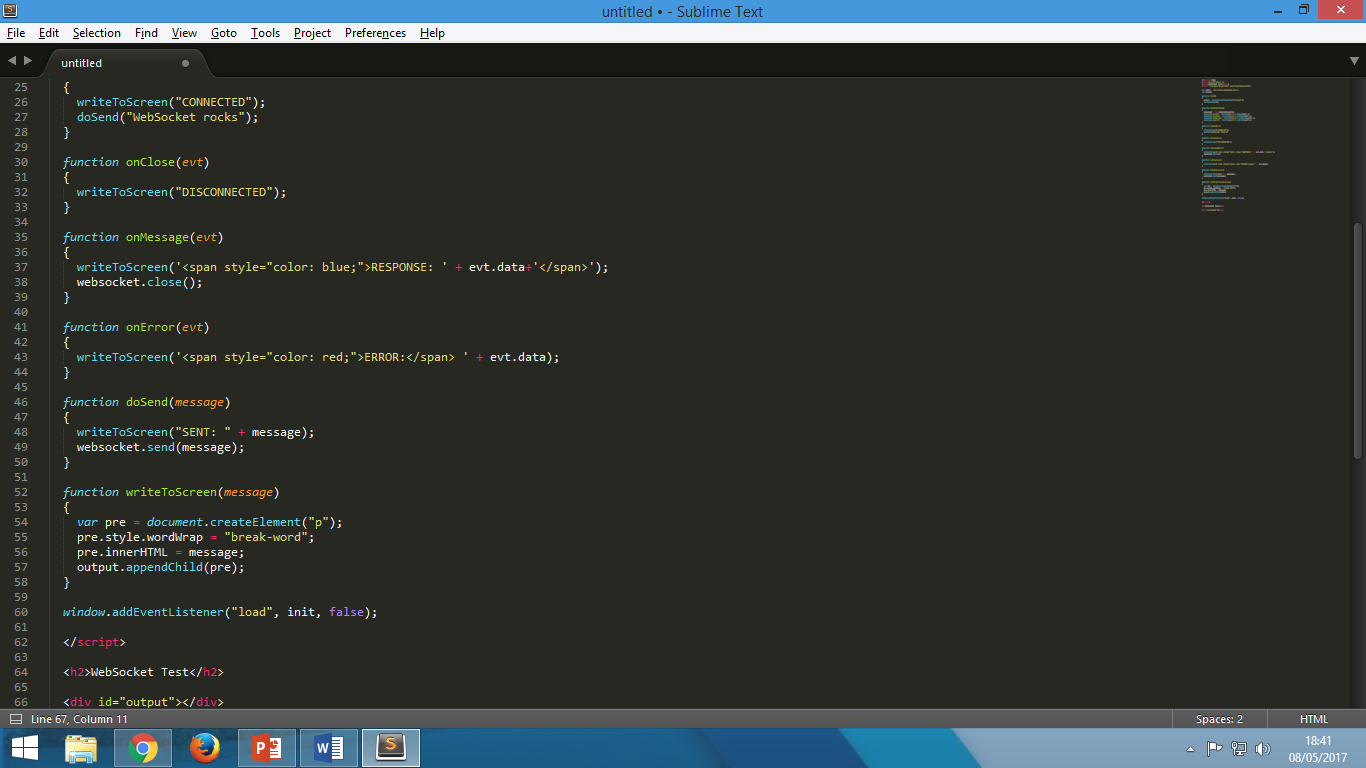
Esta acción es generalmente preferida por alguna comunicación con el servidor, que incluye el envío de mensajes, que son basicamente archivos de texto, datos binarios o imágenes.

* + 1. **close()**

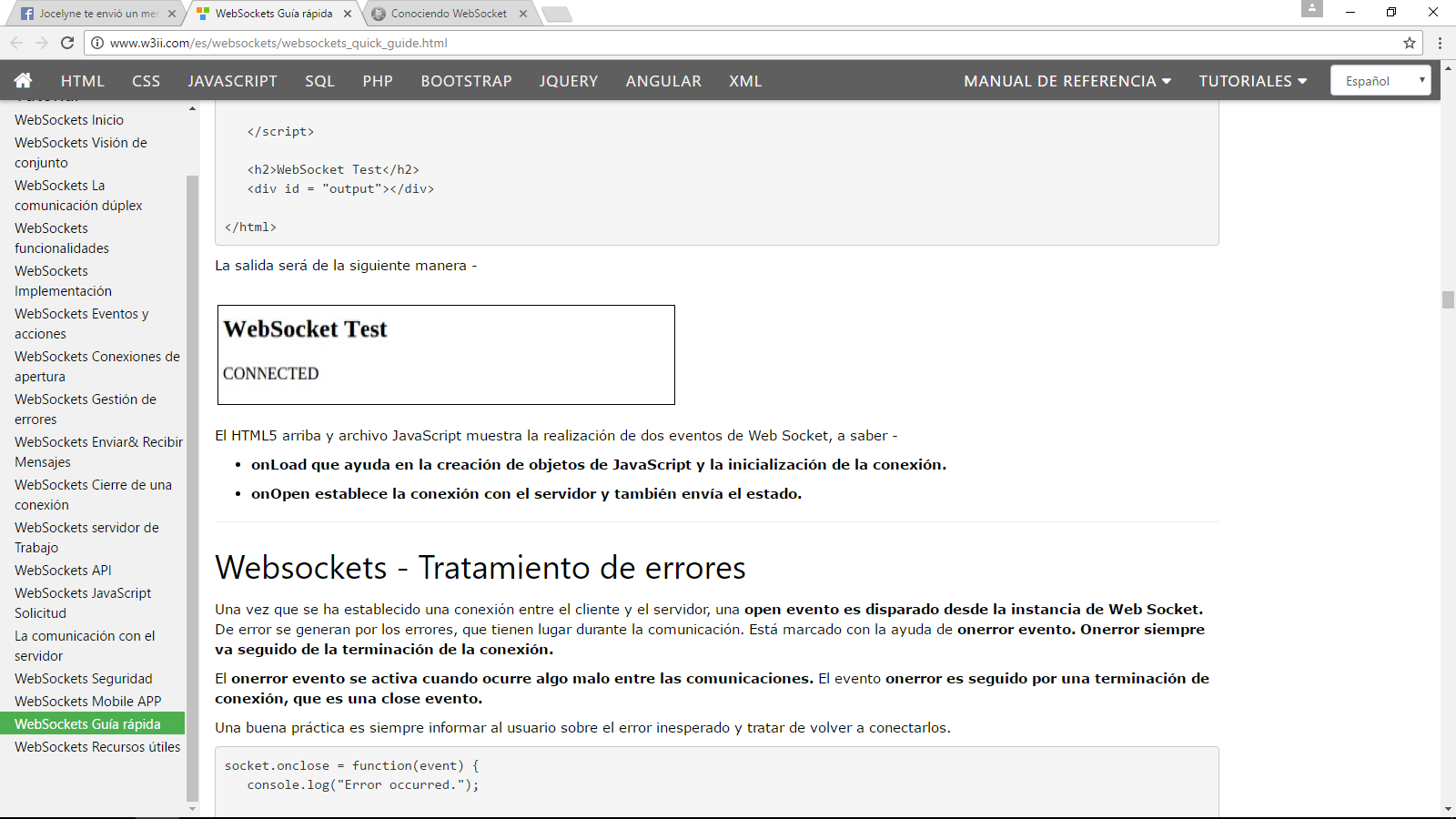
Este método es sinónimo de apretón de manos de despedida. Se termina la conexión completamente y no hay datos que puedan ser transferidos hasta que se reestablece la conexión.

* 1. **Ejemplo**

La construcción de la ficha de cliente en HTML5.



La salida será de la siguiente manera:



El HTML5 arriba y archivo JavaScript muestra la realización de dos eventos de Web Socket, a saber:

* **onLoad** que ayuda en la creación de objetos de JavaScript y la inicialización de la conexión.
* **onOpen** establece la conexión con el servidor y también envía el estado.

**CAPÍTULO III. VENTAJAS Y DESVENTAJAS**

* 1. **Identificación del problema**

Bajo el paraguas de HTML han ido apareciendo, en los últimos meses, una gran cantidad de tecnologías que prometen dotar de una potencia a las aplicaciones web nunca vista antes. El fin del presente trabajo de investigación consiste en demostrar como las aplicaciones web puedan mantener una comunicación bidireccional con procesos en el lado del servidor, es decir, que puedan iniciarse en el lado del servido; todo ello con una mejora en la latencia que actualmente soportan esta clase de comunicaciones.

**3.2. Planteamiento de Solución**

Teniendo como conocimiento la utilidad de Web Sockets aprovecharemos sus funcionalidades con el fin de analizar una página web que contenga características de un servicio de chat; de esta manera demostraremos como es que Web Sockets nos ayuda a interactuar tanto servidor como cliente en tiempo real.

**3.3. Ventajas**

Como se ha visto, el protocolo permite establecer comunicaciones bidireccionales en tiempo real en la Web, posibilidad que antes solo existía de forma simulada y bastante costosa mediante técnicas como long polling. Optar por este protocolo permite reducir la saturación de cabeceras que ocurriría si se utilizase HTTP en su lugar, especialmente para aplicaciones que requieren un gran volumen de comunicaciones. Además, evita que cada aplicación utilice una solución de integración diferente, con los problemas de compatibilidad que ello conllevaría. Por otra parte, se ha visto que su funcionamiento es extremadamente sencillo: se establece una conexión, se envían/reciben mensajes y se cierra la conexión. Por último, al funcionar bajo los mismos puertos que HTTP evita problemas relacionados con cortafuegos, facilitando así el funcionamiento de productos basados en arquitecturas orientadas a servicios (SOA), entre otros.

**3.4. Desventajas**

Por otra parte, es necesario gestionar y mantener un gran número de conexiones que han de permanecer abiertas mientras ambas partes sigan interactuando. Esto puede llegar a ser un problema en determinados casos, teniendo en cuenta que el número máximo de conexiones simultáneas que admite un puerto TCP es de 64.000 y que, además, mantener las conexiones abiertas requiere memoria del servidor.

**3.5. Caso Real**

Después de ver el funcionamiento de WebSocket y todo lo que puede aportar como tecnología de comunicaciones a la integración de aplicaciones, resulta interesante destacar un caso real en el que se utiliza el protocolo: WhatsApp Web.

El servicio de mensajería WhatsApp es un servicio distribuido. Su cliente web utiliza WebSockets para comunicarse con los servidores, que se encargan de almacenar información y retransmitir a los clientes web la información que solicitan indirectamente a los dispositivos móviles.

Utilizando las herramientas para desarrolladores que proporciona Google Chrome es posible analizar las comunicaciones entre un cliente WhatsApp Web y los servidores del servicio.

**Conclusiones**

Web Sockets permite que dos aplicaciones establezcan una comunicación bidireccional dependientemente de la plataforma en la que estén ejecutándose y del lenguaje en el que hayan sido escritas. Además, existen multitud de implementaciones para prácticamente cualquier lenguaje que permiten a los desarrolladores centrarse en sus aplicaciones olvidándose de implementar comunicaciones.

Esto abre un abanico de posibilidades para la integración de aplicaciones, permitiendo a estas intercambiar información en tiempo real y de forma sencilla, contribuyendo además a la estandarización de los mecanismos de comunicación.

**Recomendaciones**

* Seguir investigando acerca de diversos casos de implementación de la tecnología Web Sockets.
* Investigar de cómo podemos aprender a desarrollar este tipo de tecnología y saber más acerca de su lenguajes de programación de navergador.

**Referencias Bibliográficas**

* Fette, I. y Melnikov, A. (2011). *The WebSocket Protocol*. Internet Engineering Task Force. <<https://tools.ietf.org/html/rfc6455>> [consulta: 02 de abril de 2017].
* Hickson, I. (2012). *The WebSocket API*. W3C. <[http://www.w3.org/TR/websockets/](https://www.w3.org/TR/websockets/)> [consulta: 05 de abril de 2017].
* Engine Yard. *WebSocket: 5 Advantages of Using WebSockets*. <<https://www.engineyard.com/articles/websocket>> [consulta: 12 de abril de 2017].
* Tiobe Software (2015). *TIOBE Index for November 2015*. <<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>> [consulta: 25 de mayo de 2017].