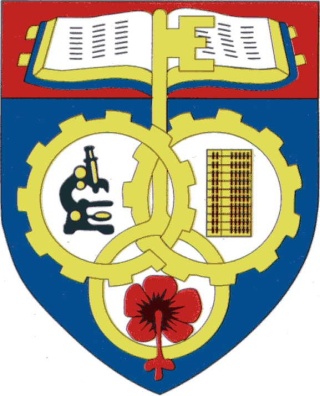
Push Technology

By

Ong Hui Theng



SCHOOL OF ARTS AND SCIENCE

TUNKU ABDUL RAHMAN COLLEGE

KUALA LUMPUR

2012/2013

Push Technology

By

Ong Hui Theng

A seminar report submitted to the School of Arts and Science

in partial fulfillment of the requirement for the

Bachelor of Science, Campbell University, U.S.A.

and

Advanced Diploma in Science,

Tunku Abdul Rahman College

2012/2013

Table of Contents

[Problem Statement 1](#_Toc334581466)

[1.0 Introduction to Push Technology 2](#_Toc334581467)

[2.0 Approaches to implement Push Technology 3](#_Toc334581468)

[2.1 HTTP Polling 3](#_Toc334581469)

[2.2 Long Polling 4](#_Toc334581470)

[2.3 HTML5 WebSocket 5](#_Toc334581471)

[2.3.1 WebSocket Protocol 5](#_Toc334581472)

[2.3.2 WebSocket API 6](#_Toc334581473)

[3.0 Technical Comparison 8](#_Toc334581474)

[3.1Network bandwidth 8](#_Toc334581475)

[3.2 Data overhead 8](#_Toc334581476)

[3.3 Real-time data 9](#_Toc334581477)

[3.4 Network traffic 9](#_Toc334581478)

[4.0 Application to Final Year Project 10](#_Toc334581479)

[4.1 Selected Push Technology 10](#_Toc334581480)

[4.2 System Architecture 11](#_Toc334581481)

[4.2.1 Contents Updating Server 12](#_Toc334581482)

[4.2.2 Notification Server 12](#_Toc334581483)

[4.2.3 Web Clients 12](#_Toc334581484)

[4.3 Implementation of the WebSocket-enabled server 12](#_Toc334581485)

[4.3.1 Node.JS and Socket.IO 12](#_Toc334581486)

[4.0 References 14](#_Toc334581487)

[4.0 Appendices 15](#_Toc334581488)

# Problem Statement

Today, more and more people are pursuing fashion and new trends. Therefore, IKnowU fashion ecommerce website provides a notification feature to notify the customers when current trends are updated, new messages have been received and subscribers are notified when the new clothing are just uploaded by the specific subscribed sellers

In order to implement the notification service, this study research has been carried out to study push technology which is able to use to implement the notification feature for the system. Three types of push technologies have been studied and compared which are HTTP polling, HTTP long polling and HTML5 WebSocket.

Since IKnowU fashion ecommerce website may has a lot of concurrent users, network bandwidth, network traffic, network latency, notification transmission speed, and availability of real-time notification issues needs to be concerned. Therefore, those three types of push technologies will be evaluated to find the best suited way to implement the notification service for IKnowU fashion ecommerce system.

# Introduction to Push Technology

Push Technology or server-push technology is opposite with the typical pull technology. Pull technology is constrained by the traditional HTTP request/response model which web client request web page from the server by pulling information to the server. Nowadays, push technology have been widely used over the internet where web server pushes information or data to the clients such as notification, stocks quotes and etc.

There are various approaches of push technologies have been adopted. Firstly, HTTP polling is used; client repeatedly request for new data from the server and server may response with or without data. This cause latency and performance problems and therefore long polling was introduced. In Long polling, the client send request to the server and the responds will be reply by the server only when the new data is available in a set period of time. Comet, which is an umbrella term was a more advanced technique that used long polling. (Lewis, 2011)

Now, HTML5 WebSocket was introduced and it was defined as the future of the web which goes beyond the past push technologies. HTML5 WebSocket is a bidirectional, full-duplex communication channel that allows asynchronous of message. It is able to achieve for real-time application which the server will respond as soon as possible to the client when new data is available without continuously request in a long-lived connection.

# 2.0 Approaches to implement Push Technology

## 2.1 HTTP Polling

With HTTP polling, the client sends http requests to the web server periodically in a fixed interval of time to request for new data. Server will responds to the client’s request immediately with or without any new data. If the new data is available, server will respond by sending new data to the client. In contrast, server will respond by sending the empty message to the client when new data does not exist for the request.

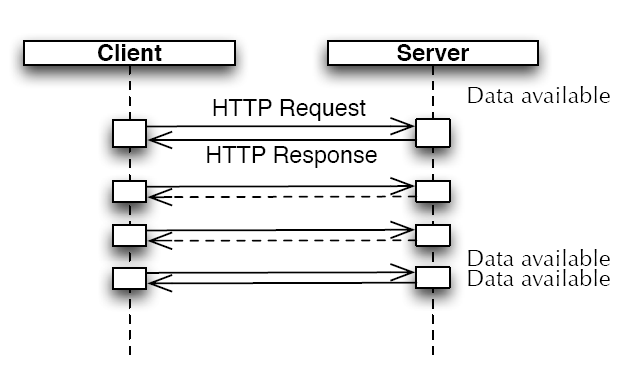


Figure 1 Communication Pattern with polling (Harri Hämäläinen, 2011)

(JSONP) Polling is one type of HTTP polling. From the sample code below, the JSONP polling is implemented in JavaScipt. setInterval is used to send requests to the server periodically.



Figure 2 JavaSript Polling (Mathieu, 2011)

JSONP Polling, which is similar to the HTTP polling, will respond to the server at once when it receives the request. From the sample JSONP polling output below, it can clearly shown that the server respond by sending event to the client when the event is available or sending no event to the client if the event is not available.

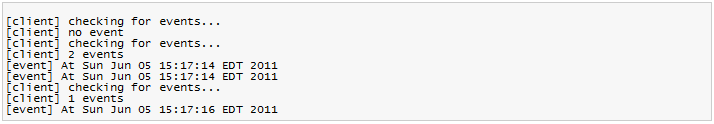


Figure 3: Sample polling demo output (Mathieu, 2011)

## 2.2 Long Polling

With long polling, the client sends HTTP requests to the web server to request for new data in a fixed interval of time. But, the server would not respond to the request if the new data is not available. It will hold the request until there is new data is available to be sent to the client or terminate the connection when the time interval is expired without any data available.

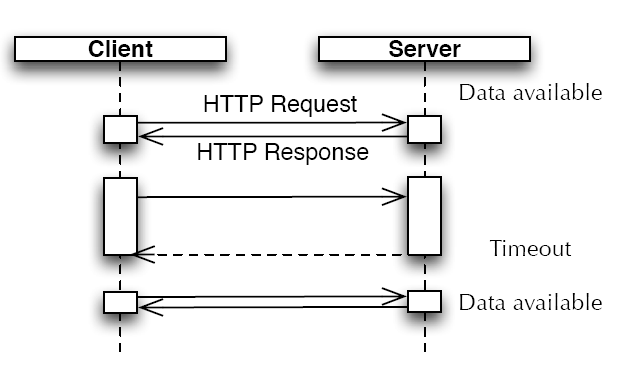


Figure 4: Communication pattern with long polling (Harri Hämäläinen, 2011)

(David, 2011) Comet which is also named as “Ajax Push”, “Server Push”, and “HTTP streaming” is one type of HTTP long polling which use the reverse of Ajax to push data to the client. In comet, the web server initiates the communication in asynchronous way. When the client such as web application needs to request data from the server, Ajax technique will be used which client pulls data from the server by asking for new data. In comet, the client needs to establish and re-establish the connection to the server. As similar to the HTTP long polling.

technique, every time the client establish a connection and make request to the server, the server will keep the connection and hold the request until new data is available to be sent or close the connection after the time interval expired. The server will close the connection instantly after sending the data. Then, the clients will processing the received data and re-establish the connection again to the server to make future request.

## 2.3 HTML5 WebSocket

HTML5 is a push technology which enables full-duplex and bi-directional communication between web server and web client. It allows data to be transmitted in real-time. In order to communicate, client needs to establish a long-lived connection to the server. After the connection has been established, data exchange can be happened at any time and server is able to push data to the client as long as the new data is available. This makes the data is able to received in the real-time.

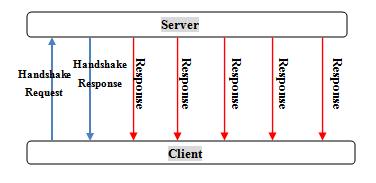


Figure 5: Cummucation pattern with WebSocket

## 2.3.1 WebSocket Protocol

WebSocket protocol is proposed as IETF’s RFC 6455 (Fette et al. 2011 ). WebSocket uses its own WS (unsecure) and WSS (secure) protocols for communication. As default, WebSocket.org states that HTML5 WebSocket use same port and same host name as the HTTP and HTTPS (WebSocket.org, 2012). Therefore, WebSocket upgrade handshake is carried out to upgrade the http:// or https:// protocols to ws:// or wss:// protocols. The process starts by the client who sends a HTTP GET request to the server to request for upgrading the protocol and this HTTP request is able to traverse proxy server and firewall. After the server has received the request, it will upgrade the HTTP protocol and send switching protocol response to the client. Server and client are able to exchange data at any time after the upgrade handshake has been completed. It is a long-lived connection as long as the connection is not terminating by the server or client.

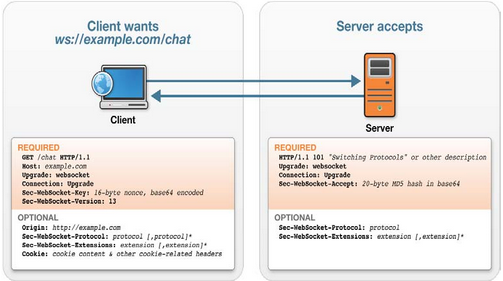


Figure 6: WebSocket Upgrade handshake (Peter Lubbers)

## 2.3.2 WebSocket API

The WebSocket API is specified by W3C and its working draft was last published in August 2012 (Ian, 2012). It defines an interface for the browser and application to allow webSocket protocol to be used by the web application.

In order to communicate, a WebSocket objects needs to be created to open the connection. Then, the browser client needs to register for the event handlers in order to inform the web application by performing specific function on it. Basically, there are four types of event handlers which are shown in the table below.

|  |  |
| --- | --- |
| Event Handlers | Description |
| onopen | Once the connection is established, client browser will fire this event handler to inform the web application and performs specific function on it. |
| onclose | After the connection has been closed, client browser will fire this event handler to inform the web application and performs specific function on it. |
| onerror | Once the errors have been occurred, client browser will fire this event to inform the web application and performs specific function on it. |
| onmessage | Once a new message has been arrived, client browser will fire this event handler to inform the web application for the incoming message. |

Table 1: Event Handlers for WebSocket

# Technical Comparison

## 3.1Network bandwidth

In HTTP pooling, in order to try its best effort to get the real-time data, the time interval between the requests needs to be reduced. Client will make requests more frequently to increase the chances of getting data as soon as possible. But, due to the requests have been increased, the chances of the clients to get the empty request become higher. This will make the consuming bandwidth become higher by processing the additional and unnecessary requests.

HTTP long polling make request in a set period of time and it does not make request periodically as HTTP polling. Therefore, HTTP long polling will consume lesser bandwidth as compare to HTTP polling.

HTML5 WebSocket does not need to make request for the response. The client is able to receive response from the server without sending any request. WebSocket does not send request as frequently as HTTP polling and HTTP long polling and therefore it consumes the lowest bandwidth as compares to HTTP polling and long polling.

## 3.2 Data overhead

HTTP polling and long polling are both go through the HTTP request/response behavior. Request header and response header will be generated through the communication and header information in these headers will cause unnecessary data overhead. Therefore, more data overhead will be caused if request and response happened more frequently. HTTP polling make request more frequently as compare to the HTTP long polling and therefore HTTP polling caused more data overhead over the HTTP long polling.

In HTML5 WebSocket, once the connection is established, only 2 bytes overhead was produces per frame in a transmission but HTTP polling produces 871 bytes of overhead as stated in (WebSocket.org, n.d.). Therefore, HTML5 WebSocket produces the least overhead than HTTP polling and long polling.

## 3.3 Real-time data

HTTP polling and long polling provide half-duplex communication. Request and response can be made through this communication but they are unable to happen in the same time. Once the request has been sent, it needs to wait for the response otherwise it is unable to make another request. Because of this, HTTP polling and long polling are only able to produce near real-time data and not suitable for real-time application.

HTM5 webScoket provides full-duplex communication. Request and response can be made through this communication and they are able to happen in the same time. Once the connection has been establish, data exchange can happen any time between the server and the client simultaneously (Gerard et al. 2011). Besides that, based on the results of the testing from Carl Gutwin, Michael Lippold and T. C. Nicholas Graham, WebSocket’s message receive rate from the server to the browser in LAN, MAN and WAN is the higher than long polling and others techniques used (Carl et al. 2011). This cause most real-time data is able to receive in HTML5 WebSocket web application.

## 3.4 Network traffic

Minimal network traffic is important in order to reduce the network latency in a communication over the network. If the network traffic is high, it will cause network latency and therefore the data will be sent in a longer time to reach the destination.

Matthias Heinrich and Martin Gaedke show that network traffic generated in HTTP polling is 1,049,400 bytes, HTTP long polling is 34, 980 bytes and WebSocket is 1, 005 bytes. This can be clearly seen that HTML5 WebSocket produces lowest network traffic as compare to HTTP polling and long polling. (Matthias and Martin, 2012)

Besides that, Victoria Pimentel and Bradford G. Nickerson have carry out a testing to measure the latency of real-time wind sensor data transmission in four countries. The results shows that the WebSocket protocol has produces lowest average latency followed by HTTP long polling and HTTP polling has produced the highest latency in the test (Victoria and BradFord, 2012).

Therefore, HTML5 WebSocket produces the lowest network traffic and the lowest latency as compared to HTTP polling and HTTP long polling.

# 4.0 Application to Final Year Project

## 4.1 Selected Push Technology

IKnowU fashion ecommerce system would like to use HTML5 WebSocket push technology to implement the notification feature.

From the technical comparison above, HTML5 WebSocket consumes the lowest bandwidth in a communication over the network. For IKnowU fashion ecommerce website, high bandwidth is needed in order to support a large numbers of customers. The system needs to send notification to the customers immediately when new trends are updated. Therefore, high bandwidth of HTML5 WebScoket is able to make server send notification to the customers in higher speed.

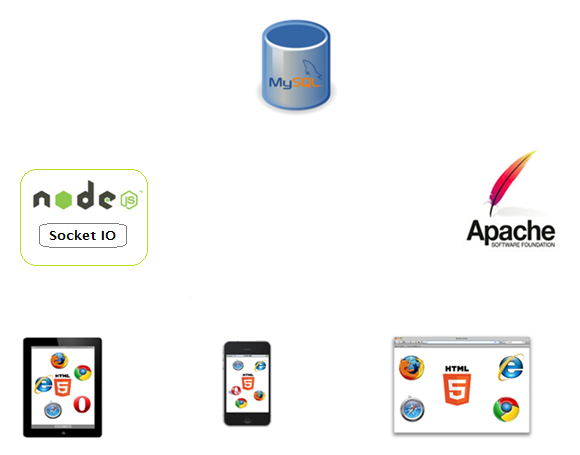
Secondly, HTML5 WebSocket produces the lowest data overhead as compared to HTTP polling and HTTP long polling. Higher data overhead will cause data delivered slower to the destination. For IKnowU fashion ecommerce system, notification needs to send to the customers as soon as possible. Therefore, HTML5 WebSocket is the best suited way to implement the notification feature in the system to provide better notification service to the customers.

Thirdly, HTML5 WebSocket is able to provide most real-time data in a communication as compared to HTTP polling and HTTP long polling. IKnowU fashion ecommerce system needs to provide real-time notification to the customers as long as the new fashion and clothing have been updated. Real-time notification is important for the system to compete with others fashion ecommerce system to provide a better service for the sellers and customers to pursue latest fashion trends in the fastest time. Therefore, HTML5 WebSocket is the best way to implement for the real-time notification service.

Lastly, HTML5 WebSocket produces the lowest network traffic and lowest network latency in a communication as compared to HTTP polling and long polling. IKnowU fashion ecommerce system needs to send real-time notification to a huge numbers of users. Therefore, numerous notifications need to be sent in the same for different customers. In order to send notification to the customers without delay, HTML5 WebSocket is chosen to implement for the notification feature of the system.

In the conclusion, IKnowU fashion system will choose HTML5 WebSocket to implement the notification feature for the system as it is the best suited way as compared to HTTP polling and HTTP long polling.

## 4.2 System Architecture



**Multi-platforms and multi-browsers**

**WebSocket Protocol**

**6th**: Send notification to web application.

**5th**: Send response to request.

**4th**: Request updated content

**2nd**: Store content and check content whether it has stored successfully.

**1st**: upload content.

**3rd**: Notify for the updated and changes.

**Node.js Server**

**Client 2**

**Client 1**

**Client 3**

**MySQL relational database**

**Apache HTTP Server**

Figure 7: System Architecture

System architecture of IKnowU fashion ecommerce website is shown in the diagram above. Basically, it is separated in three parts:

## 4.2.1 Contents Updating Server

Apache HTTP server will manage the updating contents. When the sellers update the content to the PHP, Apache Web Server will upload and store content to MySQL database and check whether the content has been stored successfully. After verified that content has been updated, update message will be sent to the Node.js server by using zeroMQ method as zeroMQ is defined in Wikipedia as a high performance, asynchronous library for concurrent and large scalable applications (“ØMQ”).

## 4.2.2 Notification Server

Node.js is responsible for retrieve updating content and sending notification to the client. When the updating message has been received, Node.JS will retrieve the updating content from the database. After the updating content has been retrieved successfully, Node.JS server will send the notification to the customers.

## 4.2.3 Web Clients

Customers are using various types of browsers to connect to IKnowU e-commerce website. The customers are able to receive notifications which are sent from the Node.JS server. After receiving the notification, customers are able to read and view the updated contents from the website.

## 4.3 Implementation of the WebSocket-enabled server

## 4.3.1 Node.JS and Socket.IO

From the diagram above, it shows that Node.JS server and Socket.IO is installed to support HTML5 push technology for the notification feature of the system.

* Install Node.js server from <http://nodejs.org/download/>.
* Install Socket.IO into the Node.js server.

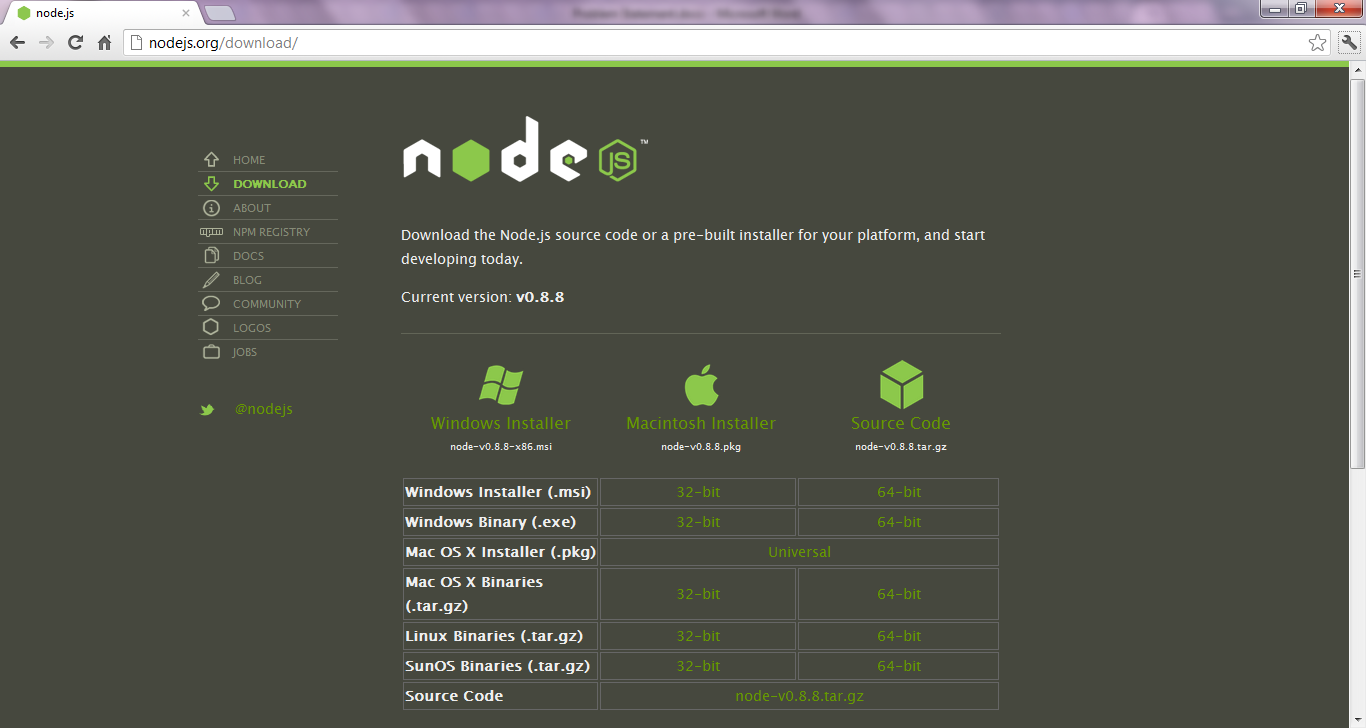
There are a lot of WebSocket-enabled servers nowadays but Node.js has been chosen for the implementation of notification feature on IKnowU fashion ecommerce system. This is because node.js server is high performance and suitable for large scalable web server. Node.js is also able to support a lot of users concurrently without degrading of the performance (Paudyal, 2011). Therefore, Node.js server is suitable for IKnowU fashion ecommerce system to send notification to a large numbers of users simultaneously.

Socket.IO is a JavaScript library that runs on both client and server side which is designed to enable Node.js to cross-browser (Tasha et al. n.d.). Since HTML5 Web Socket does not support quite a lot of browsers which can be seen from <http://caniuse.com/websockets>. Therefore, Socket.IO is a good choice to allow cross-browser. Socket.IO will use fallback function to use others methods if WebSocket is not supported for the client’s browser.

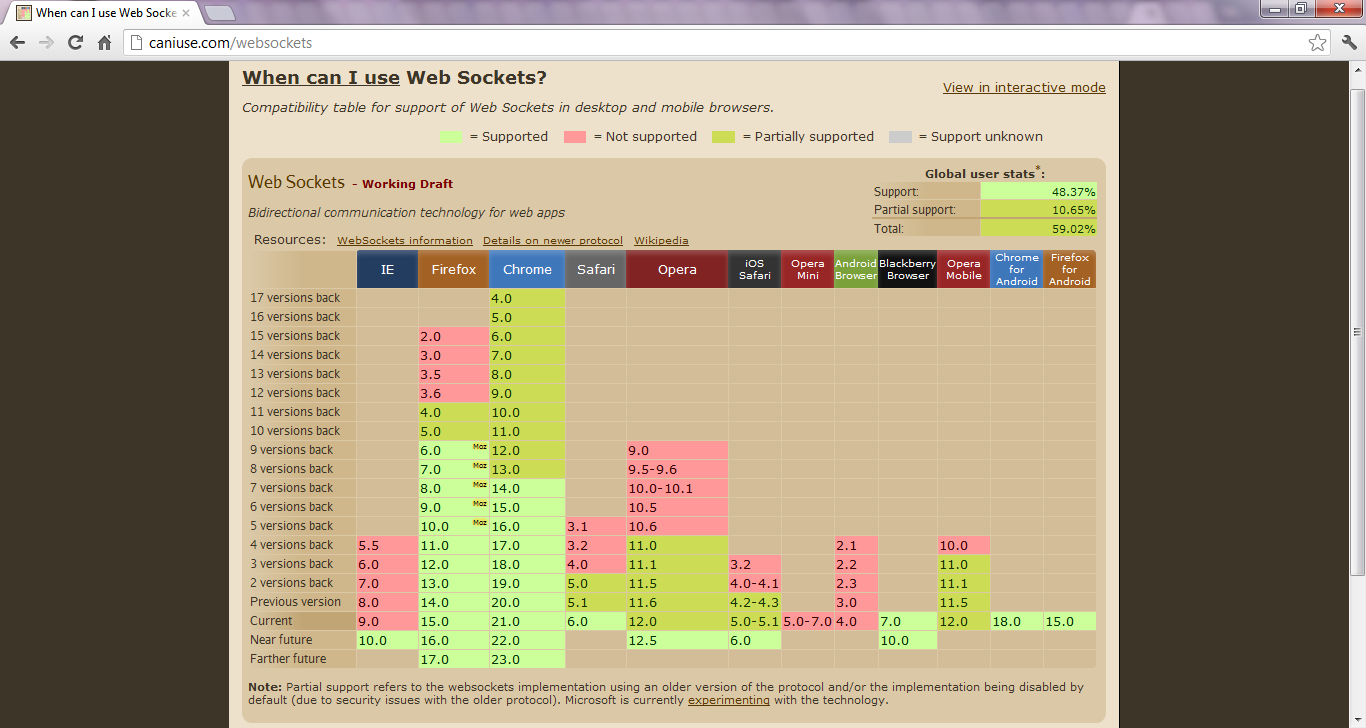
# 4.0 References

1. Benge, L. (2011) HTML 5 Web Sockets: The Real Time Web. *Lewis Benge*, [blog] April 25, 2011 , Available from: http://www.lewisbenge.net/index.php/2011/04/html-5-web-sockets-the-real-time-web/ [Accessed: 27/08/12].
2. Harri Hämäläinen (2011) *HTML5: WebSocket*, pp. 1.
3. Fette, I. et al. (2011) The WebSocket Protocol. *Internet Engineering Task Force (IETF),*(December), p.1.
4. IBM (2012) *Reverse Ajax, Part 1: Introduction to Comet*. [online] Available from: http://www.ibm.com/developerworks/web/library/wa-reverseajax1/index.html#download [Accessed: 30/08/12].
5. Flanagan, D. (2012) *JavaScript: The Definitive Guide*. California: O'REILLY, p.491-493.
6. WebSocket.org (2012) *About HTML5 WebSockets*. [online] Available from: http://www.websocket.org/aboutwebsocket.html [Accessed: 01/09/12].
7. WebSocket.org (n.d.) *About HTML5 WebSockets*. [online] Available from: http://www.websocket.org/aboutwebsocket.html [Accessed: 31/08/12].
8. W3C (2012) *The WebSocket API*. [online] Available from: http://www.w3.org/TR/websockets/ [Accessed: 03/09/12].
9. Pimentel, V. and G. Nickerson, B. (2012) Communicating and Displaying Real-Time Data with WebSocket. *Programmatic Web Interfaces*, (July/August) p.47-52.
10. Heinrich, M. and Gaedke, M. (2012) Data Binding for Standard-based Web Applications. p.4-6.
11. Paudyal, U. (2011) Scalable web application using node.JS and CouchDB. p.1-6.
12. Thiebaut, T. et al. (n.d.) The Impact of William, a mind mapping writing application. p.6.
13. Nicolas, G. et al. (2011) Architecting end-to-end convergence of Web and Telco services. p.1-7.
14. Gutwin, C. et al. (2011) Real-Time Groupware in the Browser: Testing the Performance of Web-Based Networking. p.172.
15. Wikipedia (2012) *ØMQ*. [online] Available from: http://en.wikipedia.org/wiki/ØMQ [Accessed: 04/09/12].

# Appendices



<http://nodejs.org/download/>



<http://caniuse.com/websockets>