Virtual Dosen Menggunakan Teknologi WebRTC Berbasis Browser

Bartholomeus Bismo Baruno, Alif Subardono, Sri Lestari
Departemen Teknik Elektro dan Informatika, Sekolah Vokasi Universitas Gadjah Mada
Jl. Yacaranda, Sekip Unit IV, Yogyakarta 55281Indonesia
bartholomeus.bismo.b@mail.ugm.ac.id, masalif@gmail.com, srilestari59@gmail.com

Abstract— **My college activity was the most important things for the student. However this thing started to be a trouble since the professor was unable to be present on some classmeeting activities and as the result every material and class activities need to be postponed and as the replacement, the material will be distributed via softcopy. One of the best solution that can be done in order to have interactive school activity is by doing long distance communication through realtime system.**

Thanks to the fast development of science facility especially the Web Real Time Communication (WebRTC), the communication through inter-web media can be done through realtime and gives the advantage for its users. Everything will be easier since it doesn't need some add-ons to have the full access of it, only browser is necessary. This research will try to test the browser's compatibilities and knowing the presence of delay between the users when the communication is being held.

Keywords*; *WebRTC, realtime, delay, browser, communication

Abstrak—**Kegiatan perkuliahan merupakan hal yang penting bagi mahasiswa. Namun hal ini akan menjadi masalah tersendiri ketika dosen tidak dapat hadir di kelas dalam kurun waktu tertentu sehingga kegiatan perkuliahan terpaksa diundur atau materi yang disampaikan hanya berupa tugas maupun file *softcopy*. Salah satu solusi agar kegiatan perkuliahan masih dapat dilakukan secara interaktif adalah melalui komunikasi jarak jauh secara langsung dengan sistem *realtime*.**

Dengan perkembangan web yang semakin pesat khususnya teknologi *Web Real Time Communication* (WebRTC) maka komunikasi melalui media web akan dapat dilakukan. Melalui WebRTC, komunikasi akan dapat dilakukan secara *realtime* serta memberikan kemudahan bagi pengguna karena tidak perlu ada *plugin* tambahan untuk mengakses sistem sehingga cukup menggunakan *browser*. Penelitian ini akan menguji kompabilitas dari *browser* yang digunakan serta mengetahui tingkat *delay* yang terjadi pada saat komunikasi dilakukan.

Kata-kunci*; WebRTC, realtime, delay, browser,* komunikasi

I. PENDAHULUAN

Sistem perkuliahan saat ini masih terbatas pada tempat dan waktu yang digunakan. Yang artinya kegiatan kuliah akan dapat dilakukan ketika dosen dapat hadir sesuai jadwal dalam lokasi yang sama. Namun seringkali perkuliahan tidak dapat dilakukan karena berbagai hal sehingga kegiatan kuliah harus ditunda untuk sementara waktu.

Berdasarkan masalah yang terjadi tersebut diperlukan adanya sistem baru yang dapat menjadi solusi komunikasi tanpa mengenal jarak dan waktu. Sistem ini akan menjembatani komunikasi antara dosen dengan mahasiswa di ruang kelas dengan melalui sebuah jaringan internet. Dengan munculnya teknologi WebRTC, komunikasi akan dapat dilakukan secara *realtime*.

Sedikit berbeda dengan sistem komunikasi yang lain seperti *Skype*, WebRTC akan berjalan pada aplikasi web menggunakan *browser*. WebRTC dirancang untuk mendukung video *conference* hanya dengan menggunakan sebuah *browser* tanpa perlu adanya *plugin* tambahan. Untuk saat ini, *browser* yang mendukung dengan teknologi ini adalah Google Chrome, Mozilla Firefox dan Opera baik untuk *desktop* maupun *mobile*.

Munculnya konsep WebRTC ini didasari ide pada penyediaan layanan komunikasi yang berkualitas tinggi namun dengan metode yang sederhana. WebRTC juga merupakan solusi untuk menjembatani komunikasi yang dilakukan antar *platform*. Karena *browser* bersifat universal dimana semua perangkat baik *mobile* maupun *desktop* dengan berbagai macam *platform* yang berbeda mempunyai aplikasi tersebut. Hal ini secara tidak langsung akan memberikan kemudahan akses bagi pengguna.

Dengan melihat keunggulan yang dimiliki WebRTC tersebut maka dalam makalah ini akan diangkat penelitian mengenai implementasi WebRTC untuk mendukung sistem perkuliahan secara jarak jauh.

II. KAJIAN PUSTAKA

A. Komunikasi Virtual

Komunikasi virtual atau komunikasi dalam jaringan adalah cara berkomunikasi di mana penyampaian dan penerimaan informasi atau pesan dilakukan dengan menggunakan media internet atau melalui dunia maya (*cyberspace*). Internet merupakan media komunikasi yang cukup efektif dan efisien dengan tersedianya berbagai layanan seperti web, *chatting*, email, sosial media dan lainnya. Komunikasi virtual mempunyai beberapa keunggulan, diantaranya adaalah dapat dilakukan kapan saja dan dimana saja, efisiensi waktu biaya, terintegrasi dengan layanan teknologi informasi dan komunikasi

lainnya, meningkatkan intensitas berkomunikasi dan meningkatkan partisipasi. [1]

B. *WebRTC*

WebRTC merupakan sebuah upaya untuk menambahkan kemampuan komunikasi *realtime* ke dalam semua *browser* dan membuat kemampuan ini dapat diakses pengembang web melalui *tag* standar HTML5 dan API JavaScript. Dengan mengikuti standar-standar yang ditentukan dalam WebRTC, *browser* menjadi berkemampuan untuk berinteraksi secara langsung dengan *browser* yang lain. Untuk saat ini *browser* yang mendukung penggunaan teknologi WebRTC adalah Google Chrome, Mozilla Firefox dan Opera baik untuk *desktop* maupun *mobile*. [2]

C. *HTML5*

HTML5 adalah generasi selanjutnya dari HTML, menggantikan HTML 4.01, XHTML 1.0, dan XHTML 1.1. HTML5 menyediakan berbagai macam fitur baru yang diperlukan untuk aplikasi web modern. HTML5 juga memberikan standar untuk banyak fitur pada *platform* web yang telah digunakan oleh pengembang web selama beberapa tahun dimana fitur tersebut tidak pernah diperiksa atau didokumentasikan oleh standar komite. [3]

Fitur HTML5 yang digunakan untuk komunikasi menggunakan teknologi WebRTC adalah fitur *tag* video sebagai media untuk menampilkan video dan audio dari lawan bicara secara *realtime*.

D. *PHP*

PHP adalah sebuah bahasa pemrograman *scripting* untuk membuat halaman web yang dinamis. Walaupun dikenal sebagai bahasa untuk membuat halaman web, tapi PHP sebenarnya juga dapat digunakan untuk membuat aplikasi *command line* dan juga GUI. Cara kerja PHP adalah dengan menyelipkannya di antara kode HTML (*hypertext markup language*). [4]

PHP dapat berperan sebagai sistem informasi untuk mengatur data yang di*input*kan pengguna ke *database* atau menangani permintaan pengguna untuk menampilkan data dari *database*.

E. *MySQL*

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan DBMS (*database management system*), *database* MySQL ini sifatnya *multithread, multi-user*. [5] MySQL berfungsi untuk menyimpan data yang di*input*kan oleh pengguna. Data yang disimpan bersifat permanen sehingga memerlukan MySQL sebagai media penyimpanan data. Selain menangani penyimpanan data, MySQL juga melayani permintaan pengguna untuk melihat data yang ada.

F. *NodeJS*

NodeJS pertama kali dikembangkan oleh Ryan Lienhart Dahl pada tahun 2009. NodeJS ini dibuat untuk memperoleh performa aplikasi web yang tinggi dan mengoptimalisasikan *concurrent* yang tinggi pula. NodeJS memungkinkan bekerja dengan keandalan tinggi dan dengan metode I/O *non-blocking*. NodeJS ditulis dalam bahasa Javascript dan berbasis pada *event* (*event driven*). Tidak seperti kebanyakan bahasa Javascript yang dijalankan di *client*, NodeJS dieksekusi sebagai aplikasi server. NodeJS memiliki efisiensi memori yang lebih baik apabila bekerja dalam muatan yang banyak. Pengguna tidak perlu khawatir dengan terjadinya *deadlock* karena tidak ada *lock* dalam NodeJS. NodeJS terdiri dari V8 Javascript *engine* buatan Google dan beberapa modul bawaan yang terintegrasi. Di dalam NodeJS terdapat operasi yang sifatnya *synchronous* dan *asynchronous*, tetapi keunggulan NodeJS ini terdapat pada *asynchronous*nya atau bisa juga disebut dengan *non-blocking* I/O. [6]

Dengan kehandalan ini, NodeJS cocok digunakan untuk pertukaran data yang sifatnya *realtime*. Karena NodeJS merupakan model yang efisien untuk membangun sebuah aplikasi yang harus bersinggungan dengan *concurrency* (dua hal yang bekerja secara bersamaan). Hal ini membuat NodeJS mampu memberikan layanan jaringan yang cepat dan *scalable*.

G. *PeerJS*

PeerJS merupakan *library* yang ditulis menggunakan bahasa pemrograman NodeJS untuk membuat sistem *realtime communication*. Sistem yang dimaksud meliputi sistem *live chat*, *file transfer* maupun *video chat*. PeerJS merupakan *library* yang bersifat *open source* sehingga bebas digunakan dan dikembangkan. PeerJS juga menyediakan sebuah server jika para *developer* tidak ingin menggunakan server sendiri untuk aplikasi yang dibuat. [7]

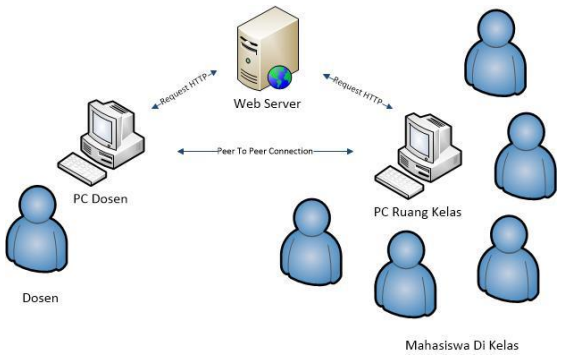
H. *STUN,TURN,ICE*

Merupakan suatu server yang dibutuhkan oleh WebRTC ketika melakukan *peer to peer communication*. Karena klien dari WebRTC membutuhkan pertukaran *metadata* untuk mengkoordinasikan suatu komunikasi yang disebut *signaling*. Selain itu, STUN,TURN,ICE dibutuhkan untuk mengatasi masalah NAT dan *firewalls* pada tiap PC yang akan saling berkomunikasi agar komunikasi *peer to peer* dapat dilakukan. [8]

III. METODOLOGI DAN PERCOBAAN

A. *Rancangan Topologi Sistem*

Pada penelitian ini akan dikembangkan suatu sistem komunikasi *realtime* berbasis web untuk kegiatan perkuliahan. Komunikasi yang terjadi antar pengguna berlangsung secara *peer to peer* dengan terlebih dahulu melakukan *request* HTTP ke web server untuk proses *signaling*. Setelah proses *signaling* selesai dilakukan, maka koneksi *peer to peer* akan terbentuk dan antar pengguna (dalam hal ini dosen dengan ruang kelas) akan dapat saling berkomunikasi. Topologi sistem dapat dilihat pada Gambar 1 berikut ini.



Gambar 1. Rancangan topologi sistem

B. *Prosedur Penelitian*

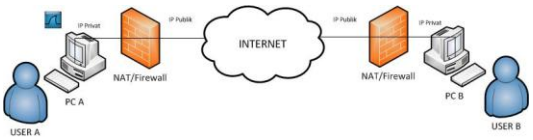
Gambar 2. *Flowchart* pengujian kompabilitas *browser*

Pada pengujian yang pertama adalah pengujian tentang kompabilitas *browser*. Karena sistem ini menggunakan *browser* sebagai media utama komunikasi, maka kompabilitas *browser* menjadi hal yang penting agar dapat diketahui sistem berjalan pada *browser* apa saja. *Browser* yang diuji meliputi *browser* untuk *desktop* maupun *mobile*.

Pengujian dilakukan dengan cara membuka sistem menggunakan *browser* tertentu. Jika terdapat notifikasi bahwa *browser* tidak support WebRTC hal itu berarti *browser* tidak memungkinkan untuk digunakan sebagai media komunikasi dengan teknologi WebRTC.

Gambar 3. *Flowchart* pengujian *QoS* sistem

Setelah pengujian kompabilitas selesai dilakukan, pengujian yang berikutnya adalah pengujian *Quality of Service* (QoS) meliputi *Throughput*, *Delay*, *Jitter* dan *Loss Packet*. Pengujian tersebut dibantu dengan *software* bernama Wireshark. Wireshark akan dipasang pada salah satu PC yang akan berkomunikasi. Setelah komunikasi dilakukan, pengamatan dengan Wireshark akan dimulai. Wireshark akan melakukan *monitoring* paket data yang dikirim maupun diterima oleh PC. Pergerakan paket data akan terekam baik untuk informasi besar paket, protokol yang digunakan maupun IP dan Port yang dipakai untuk transimisi data. Rancangan topologi pengujian QoS dapat dilihat pada Gambar 4 berikut ini.



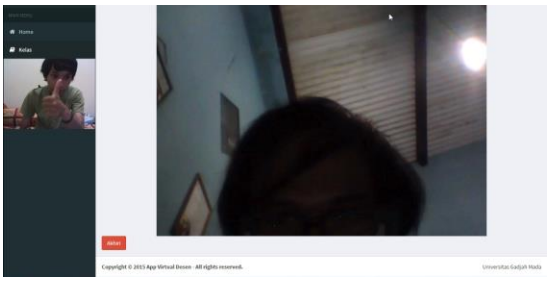
Gambar 4. Rancangan topologi pengujian QoS

Sedangkan untuk pengujian kualitas video dan audio dilakukan dengan melakukan pengamatan secara langsung. Kualitas video maupun audio akan diamati apakah data video dan audio yang diterima cukup lancar dan tidak ada gangguan.

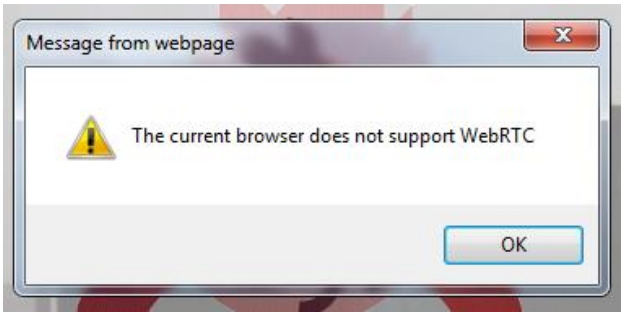
IV. HASIL DAN PEMBAHASAN

A. *Hasil Pengujian Kompabilitas Browser*

Hasil pengujian kompabilitas *browser* ini akan menunjukkan hasil berhasil jika dapat berkomunikasi seperti pada Gambar 5. Dan jika *browser* tidak mendukung penggunaan teknologi WebRTC maka akan muncul notifikasi seperti pada Gambar 6.



Gambar 5. Hasil uji coba kompabilitas *browser*



Gambar 6. *Browser* tidak mendukung teknologi WebRTC

Untuk hasil uji coba kompabilitas *browser* selengkapnya dapat dilihat pada Tabel 1.

TABEL1. HASIL PENGUJIAN KOMPABILITAS *BROWSER*

Browser	*Support* Teknologi WebRTC
Firefox	√
Chrome	√
Opera	√
Firefox *Mobile*	√
Chrome *Mobile*	√
Opera *Mobile*	√
I.E.	x
Safari	x

Dari hasil pengujian Tabel 1 dapat dilihat bahwa untuk saat ini *browser* yang mendukung teknologi WebRTC meliputi Mozilla Firefox, Google Chrome, Opera, baik untuk *desktop* maupun *mobile* dengan ditunjukkan tanda √. Sedangkan untuk *browser* Internet Explorer dan Safari masih belum mendukung penggunaan teknologi tersebut dengan ditunjukkan tanda x.

B. Hasil Pengujian QoS

TABEL2. HASIL PENGUJIAN KOMUNIKASI

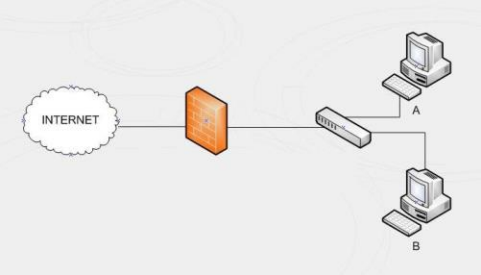
Kecepatan	Throughput	Delay	Jitter	Loss Packet	Kualitas Video	Kualitas Audio
20.63 / 19.29 Mb	3,156 Mbit/s	0,001783 s	0,0 s	1,46%	Lancar, tidak ada gangguan	Lancar, tidak ada gangguan
10.61 / 1.61 Mb	4,429 Mbit/s	0,001578 s	0,0 s	0,00%	Lancar, tidak ada gangguan	Lancar, tidak ada gangguan
5.13. / 1.51 Mb	1,538 Mbit/s	0,003001 s	0,0 s	0,00%	Lancar, tidak ada gangguan	Lancar, tidak ada gangguan
1.49 / 6.22 Mb	2,684 Mbit/s	0,002232 s	0,0 s	7,10%	Lancar, tidak ada gangguan	Lancar, tidak ada gangguan
0.11 / 0.68 Mb	0,681 Mbit/s	0,004684 s	0,0 s	11,00%	Kadang terputus, gambar kadang hilang	Lancar, tidak ada gangguan

Hasil pengujian pada Tabel 2 merupakan hasil QoS meliputi *Throughput, Delay, Jitter* dan *Loss Packet* serta kualitas dari audio dan video yang diterima. Jika semakin besar *Throughput* dan semakin kecil nilai *Delay, Jitter* dan *Loss Packet* tentu saja ini kondisi yang ideal bagi pengguna sistem. Begitu pula dengan kualitas audio dan video yang diterima. Jika kondisinya lancar dan tidak ada gangguan maka kondisi ini merupakan kondisi yang ideal. ini tentunya dipengaruhi oleh berbagai macam faktor yang ada. Contohnya adalah kecepatan koneksi yang digunakan.

Pengujian dilakukan sebanyak lima kali dengan kecepatan yang berbeda. Hal ini untuk mengetahui apakah sistem layak dan mampu digunakan pada kecepatan tertentu. Pengujian yang pertama mempunyai QoS yang bagus. Dengan tidak adanya *Jitter* hal ini berarti tidak ada perbedaan waktu antar paket yang datang. Sehingga informasi yang diterima utuh dan tidak rusak. Sedangkan nilai *Packet Loss* adalah nilai dari paket yang hilang selama perjalanan dari sumber ke tujuan. Untuk nilai *Packet Loss* pada kecepatan ini adalah 1,46%. Nilai ini bisa disebabkan oleh beberapa faktor seperti *signal* yang menurun, paket yang *corrupt*, kesalahan *hardware* jaringan, memori yang kurang dan lain sebagainya. Untuk nilai *packet loss* ini masih dapat ditolerir oleh pengguna sehingga tidak mengganggu komunikasi. Kemudian untuk *Delay* yang didapatkan adalah 0,001783 detik. Nilai ini merupakan nilai yang kecil karena *Delay* tidak sampai 1 detik. Nilai ini dapat ditolerir dan tidak mengganggu komunikasi. Yang terakhir adalah *Throughput* didapat. *Throughput* merupakan kemampuan sebenarnya dari suatu jaringan untuk melakukan pengiriman data. Dilihat dari hasil yang didapat, *Throughput* pada kecepatan ini adalah 3,156 Mbit / s dimana artinya paket yang mampur dikirimkan adalah sebanyak 3,156 Mbit dalam setiap detik.

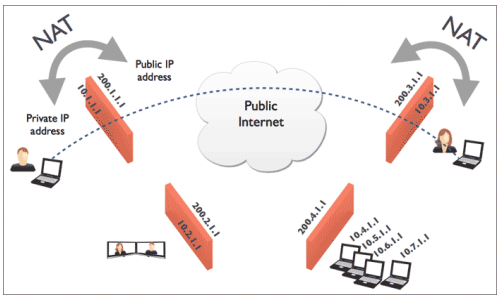
Pengujian kedua dapat dilihat jika hasil yang didapat lebih bagus daripada pengujian pertama walaupun kecepatan jaringan yang digunakan lebih lambat, *Throughput* lebih besar dan *Delay* lebih kecil. Hal ini disebabkan oleh penggunaan jaringan antara kedua *user*. Kedua *user* berada pada satu jaringan yang sama dengan kata lain mengakses pada jaringan internet yang sama. Sehingga koneksi yang terjadi masih dalam satu *segmen* jaringan. Pengguna yang berkomunikas pada jaringan yang sama relatif mempunyai waktu terhubung dan waktu tunda yang lebih kecil. Hal ini dikarenakan ketika pengguna berada dalam satu jaringan yang sama, mereka tidak mempunyai masalah dengan NAT / *Firewall*. Sedangkan jika berbeda jaringan maka masing – masing pengguna sudah terlindungi oleh NAT / *Firewall*.

Konsep WebRTC adalah komunikasi secara *peer to peer*. WebRTC mempunyai konsep komunikasi dimana data yang dikirim akan langsung sampai ke pengguna. Hal ini sulit dilakukan mengingat identitas tiap pengguna yang berada di jaringan internet terlindungi oleh *firewall* dengan menggunakan NAT. Kondisi topologi NAT dapat dilihat pada Gambar 7 berikut.



Gambar 7. Topologi NAT

Karena masalah ini WebRTC perlu sebuah server untuk menangani masalah *firewall* dengan cara melakukan *by-pass firewall*. Server ini disebut server STUN,TURN dan ICE. STUN dan TURN adalah sebuah protokol yang membantu mengarahkan paket data pada komunikasi *peer to peer*. Sedangkan ICE lebih ke teknik yang menggunakan STUN dan TURN pada komunikasi *peer to peer*. Cara kerja server seperti Gambar 8 berikut.



Gambar 8. *By-pass Firewall* NAT

Dengan gambaran cara kerja diatas juga menjelaskan alasan mengapa kondisi pengujian yang menggunakan jaringan yang sama mempunyai hasil yang lebih bagus daripada jaringan yang berbeda. Karena jika *user* berkomunikasi dengan jaringan yang berbeda, teknologi WebRTC akan mempunyai waktu yang lebih lama untuk mencari identitas *user* yang dihubungi. Selain itu *user* yang dihubungi juga terlindungi oleh *firewall* sehingga membutuhkan bantuan dari STUN,TURN,ICE server untuk melakukan *by-pass*.

Pengujian ketiga masih mempunyai nilai yang bagus. Karena *Loss Packet* yang tidak ada sehingga informasi tidak rusak. Namun demikian *Delay* terjadi lebih besar daripada pengujian sebelumnya. Dan *Throughput* yang dihasilkan tidak sebesar pengujian sebelumnya. Hal ini disebabkan oleh kecepatan jaringan yang tidak secepat pengujian sebelumnya. Sehingga QoS yang dihasilkanpun berbeda. Dan pengujian ini menggunaan jaringan data nirkabel yang secara umum tidak lebih stabil dari yang menggunakan kabel.

Dua pengujian terakhir menunjukkan nilai yang cukup. Karena dilihat dari *Packet Loss* yang terjadi masih cukup tinggi. Dengan nilai demikian informasi yang terkirim bisa tidak utuh sehingga tidak dapat disampaikan dengan baik. Dan *Throughput* yang dihasilkan tidak sebesar tiga percobaan sebelumnya. Hal ini mengakibatkan pengiriman data tidak bisa banyak dalam kurun waktu tertentu. Untuk pengujian kelima pengaruhnya ada pada kualitas video yang tidak begitu lancar. Sehingga hal ini akan menganggu kenyamanan dalam berkomunikasi. Untuk penggunaan sistem dengan kecepatan seperti ini perlu dihindari agar komunikasi berjalan dengan nyaman.

V. KESIMPULAN

1. Sistem ini mampu menghubungkan dua pengguna yang ingin berkomunikasi dari jaringan yang berbeda secara *realtime*.
2. Pengguna yang menggunakan satu sumber jaringan mempunyai QoS yang lebih baik daripada yang menggunakan jaringan yang berbeda dilihat dari *Throughput, Delay, Jitter* dan *Packet Loss*.
3. Hindari penggunaan jaringan internet dengan kecepatan dibawah 5 Mbps untuk *download* dan 1 Mbps untuk *upload* dan yang menggunakan media nirkabel karena komunikasi yang terjadi tidak stabil, seringkali terputus dan paket yang hilang cukup banyak.
4. Sistem dapat berjalan di *browser* Mozilla Firefox, Google Chrome dan Opera baik untuk *desktop* maupun *mobile*.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan YME, orangtua dan keluarga, dosen pembimbing, dosen dan kepala jurusan Teknologi Jaringan, teman dekat, serta berbagai pihak yang tidak bisa penulis sebutkan.

DAFTAR PUSTAKA

[1] Langkat, Imam. 2014. *Pengertian Komunikasi Dalam Jaringan*. http://www.slideshare.net/imamlangkat/pengertian-komunikasi-daring. Diakses pada tanggal: 23 Mei 2015.

[2] Johnston, A. B., & Burnett, D. C. 2012. *WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web (1st Edition)*. St. Louis, MO: Digital Codex LLC.

[3] Pilgrim, Mark. 2010. *HTML5: Up and Running*. O'Reilly Media, Inc.

[4] Zaki, Ali & SmitDev Community. 2008. *36 Menit Belajar Komputer Php Dan Mysql*. PT. Elex Media Komputindo, Jakarta.

[5] Huda, Miftakhul. 2010. *Membuat Aplikasi Database Dengan JAVA, MySQL, dan Netbeans*. PT. Elex Media Komputindo, Jakarta.

[6] Teixeira, Pedro. 2013. *Professional Node.js: Building Javascript Based Scalable Software*. John Wiley & Sons, Inc.

[7] PeerJS enables WebRTC browser-to-browser. http://www.h-online.com/open/news/item/PeerJS-enables-WebRTC-browser-to-browser-banter-1804427.html. Diakses pada tanggal: 20 September 2014.

[8] Ahson, Syed A., Mohammad Ilyas. 2009. *SIP Handbook: Services, Technologies, and Security of Session Initiation Protocol*. Taylor & Francis Group.